Degree Project in Computer Science and Engineering

Second cycle, 30 credits

Enhancing Generative User Interfaces with LLMs

A User-driven Iterative Refinement Process

YUTONG FANG

Enhancing Generative User Interfaces with LLMs

A User-driven Iterative Refinement Process

YUTONG FANG

Master's Programme, ICT Innovation, 120 credits
Date: September 22, 2025

Supervisors: Amir H. Payberah, Rikaz Nismi
Examiner: Cristian Bogdan
School of Electrical Engineering and Computer Science
Host company: Bontouch (Framna)
Swedish title: Förbättring av generativa användargränssnitt med stora språkmodeller
Swedish subtitle: En användardriven iterativ förfiningsprocess

Abstract

Generative user interfaces can offer a personalized experience by adapting content and layout to individual preferences in real time. Recent advancements in large language models (LLMs) have demonstrated significant capabilities for dynamic and real-time user interface (UI) generation based on natural language prompts. However, existing solutions have primarily focused on user interface code generation for developers using large language models, while their practical usability and personalization capabilities for non-technical end users remain underexplored. This study investigates how users interact with a UI personalization system driven by OpenAI's GPT-4.1-nano model, integrated into a custom-built Android application, AdaptFit. This research aims to understand the user experience, the effectiveness of user involvement, the ease of user interface personalization, and the challenges users face in this UI personalization process.

This study combines both quantitative and qualitative methods, including questionnaires, usability testing with six participants, and semi-structured interviews. Thematic analysis was applied to better understand user experiences, and user iteration behaviors were recorded to examiner user satisfaction, prompt specificity, and outcome quality. Results show that users found the concept of UI personalization intriguing and engaging, but the performance of the system was inconsistent, often limited by vague prompts, LLM hallucination, and fixed system parsing structures. Specifically, well-articulated, detailed prompts yielded better outcomes, which shows the importance of prompt quality in LLM-driven design.

This thesis offers insights into the design of LLM-powered UI personalization system. Future work could explore better integration between generated outputs and UI framework to enhance real-world deployment.

Keywords

Generative User Interfaces, Large Language Models, User-Centered Design, Prompt Engineering, Android Development

Sammanfattning

Generativa användargränssnitt kan erbjuda en personlig upplevelse genom att anpassa innehåll och layout till individuella preferenser i realtid. Nya framsteg inom stora språkmodeller (LLM:er) har visat betydande förmågor att dynamiskt och i realtid generera användargränssnitt (UI) baserat på naturliga språkanvisningar. Befintliga lösningar har dock främst fokuserat på kodgenerering av användargränssnitt för utvecklare med hjälp av stora språkmodeller, medan deras praktiska användbarhet och anpassningsmöjligheter för icketekniska slutanvändare fortfarande är otillräckligt utforskade. Denna studie undersöker hur användare interagerar med ett UI-personaliseringssystem som drivs av OpenAIs GPT-4.1-nano-modell, integrerat i en specialbyggd Android-applikation, AdaptFit. Forskningen syftar till att förstå användarupplevelsen, effektiviteten i användarnas deltagande, enkelheten i personaliseringen av användargränssnittet samt vilka utmaningar användarna möter i processen för UI-personalisering.

Studien kombinerar både kvantitativa och kvalitativa metoder, inklusive enkäter, användbarhetstester med sex deltagare samt semi-strukturerade intervjuer. Tematisk analys tillämpades för att bättre förstå användarupplevelser, och användarnas iterationsbeteenden registrerades för att undersöka användartillfredsställelse, promptens specificitet och resultatets kvalitet. Resultaten visar att användarna fann konceptet med UI-personalisering intressant och engagerande, men att systemets prestanda var ojämn och ofta begränsad av vaga prompts, hallucinationer från LLM och fasta strukturer för systemets tolkning. Specifikt visade det sig att väldetaljerade och tydliga prompts gav bättre resultat, vilket betonar vikten av promptens kvalitet i design som drivs av LLM.

Denna avhandling erbjuder insikter i utformningen av LLM-drivna UI-personaliseringssystem. Framtida forskning kan undersöka bättre integration mellan genererade resultat och UI-ramverk för att förbättra verklig implementering.

Generativa användargränssnitt, Stora språkmodeller, Användarcentrerad design, Promptteknik, Androidutveckling

Acknowledgments

I would like to thank Rikaz Nismi, my supervisor at Bontouch, who gave me huge support through this whole journey. I also would like to express my gratitude to Amir H. Payberah and Cristian Bogdan, my supervisor and examiner at KTH, who offered valuable insights and guidance on my project. Also my sincere thank you to all participants who joined my usability testings, offering their time, valuable feedback, and thoughtful insights.
Moreover, I would like to thank Bontouch, who provided an amazing platform for me to conduct my research. It has been such a pleasant adventure!
Finally, I am grateful to my beloved friends and family who always offer their unconditional support.

Stockholm, September 2025
Yutong Fang

Contents

List of Figures

List of Tables

Listings

Chapter 1

Introduction

This chapter describes the specific problem that this thesis addresses, the context of the problem, the goals of this thesis project, and outlines the structure of the thesis.

1.1 Background

A generative user interface (gen UI) is a user interface (UI) that is created in real time by artificial intelligence (AI) to offer a personalized experience based on the context and requirements of the user [1]. It aims to create dynamic and adaptive UIs for end users in real time by understanding user intents, needs, and context. Gen UIs allow user experiences to become more personalized, efficient, and delightful. Instead of static one-size-fits-all designs, gen UIs enable true user-centric personalization calibrated to each individual.

Gen UIs represent a rapidly emerging research area at the intersection of human-computer interaction (HCI), AI, and user experience (UX) design. These interfaces leverage Large Language Models (LLMs) to dynamically create, adapt, and personalize user interfaces based on real-time user behavior data and contextual cues [2]. These approaches are grounded in a broader trend towards adaptive and user-centered design, wherein interfaces evolve to better align with individual preferences, goals, and usage patterns.

Recent advancements in LLMs (e.g., GPT-4) have demonstrated unprecedented capabilities in natural language understanding, reasoning, and content generation. These developments have stimulated new research and development efforts aimed at harnessing generative models to create more intuitive, personalized digital environments [1]. For instance, current work

in personalized recommender systems and adaptive dashboards has shown that tailoring digital experiences to user behavior can lead to improved engagement, satisfaction, and overall usability [3]. However, existing solutions often focus on manual customization or rule-based adaptation [4] rather than the fully generative approaches now made possible by advanced LLMs.

1.2 Problem

Current interfaces are designed following a "one-size-fits-all" approach, aiming to satisfy the broadest possible range of users. However, no single design can adequately meet the diverse preferences, goals, and contexts of all individuals. Users differ significantly in their needs, behaviors, and expectations, and static interfaces often fail to provide optimal experiences for everyone.

This gap highlights the necessity for solutions that enable highly personalized, tailor-made interfaces that dynamically adapt to each individual's needs. Recent advances in LLMs, offer a promising avenue for achieving this vision by allowing real-time, data-driven personalization at an individual level.

Moreover, it is necessary to investigate the effectiveness of user-involved refinement loops, which allows users to iteratively adjust and improve their personalized interfaces, and how such involvement influences overall satisfaction and usability outcomes.

Thus, this study aim to answer the following research questions:

- **RQ1** How does a LLM-generated personalized UI affect user satisfaction, perceived usability, and engagement compared to the baseline UI (standard, non-personalized UI)?
- **RQ2** How does user involvement affect the personalization of LLM-driven interfaces?
- **RQ3** How easily and effectively can users personalize the UIs to adapt to their needs?
- **RQ4** What challenges do users encounter when generating and personalizing the UIs to suit their needs?

Scientific and engineering issues

The engineering challenges lie primarily in:

- Prompt engineering: Designing effective prompts for LLMs to produce consistent, high-quality, and functional response that is syntactically valid, renders correctly, and supports interactive behaviors as intended.
- Automation and error handling: Implementing mechanisms to detect, correct, and iterate on LLM-generated outputs.
- Performance and scalability: Managing the computational cost associated with real-time UI personalization and iteration, especially given the resource-intensive nature of LLM inference.
- Data privacy and ethical concerns: Ensuring transparency and privacy in how user data is collected, used, and protected and maintaining user control over personalization processes.

1.3 Purpose

The purpose of this thesis is to explore and validate the use of LLMs for dynamically generating personalized UIs. This project will bridge LLM technology with UI personalization, introducing a user-involved refinement loop that enhances adaptability and usability. By exploring how generative UIs can dynamically respond to user interactions, this work will benefit end-users by providing UIs that adapt to their individual needs, enhancing usability, satisfaction, and engagement. Additionally, companies like Bontouch, involved in this research, can deliver experiences that not only meet current user expectations but also anticipate future needs, setting their clients apart in a highly competitive market. Moreover, this research intends to contribute to the broader understanding of AI-driven interface design. Researchers who study in HCI, UX, and adaptive UI design and companies who try to develop dynamic, AI-powered UI systems might find this study valuable.

1.4 Goals

The goal of this project is to develop a system that leverages LLMs to generate personalized UIs tailored to individual users' preferences and contextual needs. This has been divided into the following sub-goals:

1. Develop a mobile application that supports dynamic LLM-based UI personalization, integrating both baseline and personalized interface modes.

2. Collect and utilize user data to inform UI personalization strategies.

3. Implement iterative UI refinement, where users can modify and update their personalized UI/UX proposals.

4. Evaluate user-driven personalization effectiveness by conducting usability testing, user interviews, and satisfaction surveys, measuring both quantitative and qualitative outcomes.

1.5 Research Methodology

This research adopts a mixed-methods research strategy, combining both quantitative and qualitative approaches to evaluate the effectiveness of LLM-driven UI personalization. The methodology involves three phases:

1. Prototype Development
   A mobile application is developed using Kotlin and integrated with OpenAI's GPT-4.1-nano model. The application enables users to enter personal workout habits, UI preferences, and accessibility needs. Users then receive customized proposals calibrated to their profiles and dynamically generated UI layouts according to the proposals from the model. A custom JSON-to-UI rendering engine is implemented to support layout generation in real time without requiring recompilation.

2. Usability Testing
   A usability testing study is designed to evaluate the prototype from several perspectives including system usability, user satisfaction, and user engagement. During the session, participants are required to complete questionnaires, perform tasks, and participate in post-interviews. Observations, screen recordings, system interaction logs, and semi-structured interviews are collected. Pre- and post-test questionnaires, including the System Usability Scale (SUS) and Net Promoter Score (NPS), are administered to quantify user satisfaction and perceived usability.

3. Data Analysis
   Quantitative data (SUS, NPS, iteration logs) are analyzed to identify patterns and trend in user behavior and system interaction. Qualitative data (interview transcripts, observation notes) are analyzed using thematic analysis to extract meaningful themes related to user behavior,

challenges, and perceptions. Triangulation is used to strengthen the validity of the findings by integrating multiple data sources [5]. This mixed-method approach facilitates comprehensive assessment from both technical and user-experience perspectives. Alternative methods, such as purely observational studies or system-only evaluations without user feedback, are not chosen because they would not sufficiently capture the dynamic and subjective nature of user personalization experiences, which are central to this research.

See chapter 3 for a more detailed description of the specific methods.

1.6 Delimitations

This study focuses only on a single use case which is fitness management, instead of exploring multiple diverse application contexts. This delimitation was chosen to allow for a more controlled study scope within the available time frame, but it also means that the findings may not generalize to other domains.

Another delimitation is the reliance on prompt engineering rather than fine-tuning the LLM. This choice was made to prioritize feasibility and avoid the additional computational and dataset preparation costs of fine-tuning. While fine-tuning could potentially yield stronger personalization, it lies outside the scope of this project.

The participant pool for usability testing was also delimited by the use of convenience sampling. All six participants had higher education backgrounds, and four of them possessed programming experience. While this profile provided valuable insights into technically literate users, it may not represent the experiences of broader or less technically experienced populations.

Finally, the study employs GPT-4.1-nano as the model due to its affordability and computational efficiency. This selection reflects a pragmatic choice to align with project resources, even though other models might offer different capabilities.

1.7 Sustainability

In this project, several sustainability perspectives are worth considering.

The first one is environmental sustainability. Traditional methods of UI personalization, especially deep learning approaches, often require substantial computational power for training and deployment, leading to

increased energy consumption and carbon emissions [6, 7]. In contrast, this project employs a pre-trained, cloud-based LLM without any additional fine-tuning. This lightweight approach significantly reduces the need for computational resources, making the system more environmentally sustainable by minimizing energy usage [8]. However, cloud-based API calls to LLMs still consume energy especially when usage scales up [8]. And relying on OpenAI's infrastructure doesn't allow user to optimize for green energy usage or efficiency.

The second one is social sustainability. By empowering end users rather than only developers to personalize UIs through natural language interaction, this system promotes inclusivity and user autonomy. The tool lowers the technical barrier to customization, enabling individuals with limited programming knowledge to actively participate in the design of UIs. This creates a more user-centric and participatory design process, contributing to long-term system acceptance and engagement [9]. However, pre-trained LLMs may reflect social, cultural, or gender biases in their training data [10, 11]. This could lead to UI suggestions that unintentionally reinforce stereotypes or exclude minority preferences. LLMs may not always prioritize accessibility unless prompted explicitly. Without strict constraints, generated UIs may fail accessibility standards (e.g., color contrast, font size, screen reader compatibility) [12].

The last one is economic sustainability. The use of pre-existing LLM infrastructure also reduces the development cost and time compared to building and training custom models from scratch [13]. This can make the personalization system more accessible to smaller organizations, educational institutions, or low-resource environments where budget constraints limit the use of high-end AI infrastructure. However, relying on OpenAI's proprietary API may limit flexibility and increase future costs, especially if pricing changes [14]. This dependency may make the solution less sustainable or replicable for others over time.

1.8 Structure of the thesis

Chapter 2 presents relevant background information on LLMs, as well as its application in UI/UX design and code generation. It also covers the evolution of UI personalization. Lastly, the related works are presented. Chapter 3 presents the design and evaluation of the experimental process, as well as the evaluation criteria for analyzing results. Chapter 4 presents the outcomes of the user tests performed comparing the static baseline UI with the personalized

UI generated by LLM. Chapter 5 discusses the key findings, highlighting the strengths and limitations of the used methods. Chapter 6 summarizes the research findings and suggests potential areas for future work.

Chapter 2

Background

2.1 Large Language Models

LLMs are language models with massive amounts of parameters, and are trained with self-supervised learning on a vast amount of text [15]. LLMs have demonstrated incredible capabilities in addressing code-related tasks, especially in code generation which generates source code from natural language descriptions [16]. OpenAI Codex is one of the models that can be applied to essentially any programming task such as transpilation, explaining code, and refactoring code [17]. In the field of software development, increasing numbers of both academic researchers and industry specialists has adopted LLM-based code generation tool such as GitHub Copilot [18, 19] powered by Codex in their daily programming tasks since these tools can always offer a starting point and save the time of searching online [20].

Prompt Engineering

Prompts play an essential role in conversing effectively with LLMs since they are instructions and contexts for an LLM to set the rules and guarantee specific qualities of generated output [21, 22]. Prompt engineering is the means by which LLMs are programmed through prompts that guide them to generate desired responses aligning the user's intent[22, 23]. Some prompt engineering techniques might be useful to this project:

- **Expert Personas:** This technique first imagines a distinguished expert agent (e.g., You are a senior Android developer specialized in UI development) that is best suited for the context and then asks the LLMs to answer the question conditioned on such expert identity. Expert

personas are used to inform the knowledge, experience, and perspectives of experts to LLMs, which helps select what types of output to generate and what details to focus on [22]. Instructing LLMs to reply like professional experts can produce higher quality answers [24].

- **Chain-of-Thought (CoT):** CoT prompting mimics a step-by-step thought process, which encourages the LLM to generate intermediate rationales for solving a problem, by providing a series of reasoning steps in the instructions [25][26].CoT technique has been proven to improve the performance of LLMs with sufficient scale on complicated reasoning, sometimes to a large degree especially on arithmetic reasoning, multi-hop question answering, and symbolic reasoning [26].

- **Temperature and token control:** Temperature is a parameter that controls the creativity and randomness of the response generated by LLMs [27]. Lower temperature values result in more focused and deterministic outputs, which might be useful for generating syntactically correct code. While higher values lead to more diverse and creative responses which might be good to explore creative approaches to UI personalization [23, 27]. In prompt engineering, a token is the smallest text unit processed by an LLM. It is often smaller than a word, such as subwords or characters. For example, ”unauthorized” might be tokenized into [”un”, ”authorized”]. Every language model has token limits (for example, 2048 or 4096 tokens) for input user tasks or output responses, and exceeding these can result in incomplete response [28]. Token optimization is a crucial factor for prompt engineering because it directly impacts the efficiency, cost, and performance of LLMs [29]. Minimizing the number of tokens will lead to a faster response time, less computational resources, clearer, and more concise answers [30].

LLMs in UI/UX Design

LLMs has shown significant potential in supporting UI/UX design tasks, including ideation [31], persona generation [32], and prototyping [33], but their integration into later stages remains limited by biases, contextual gaps, and difficulties with high-fidelity outputs [34].

The emergence of LLMs has changed the design process and the use of AI in creative tasks, such as idea generation, has moved to the center of attention in both industries and academia. LLMs have the potential to provide large and diverse initial concepts to stimulate designers' thinking [35].

Duan et al. proposed ConceptVis [31], a system designed to enhance early-stage design ideation by leveraging LLMs. This system allows designers to intuitively manage the breadth and depth of concept exploration by turning the design space into an interactive knowledge graph. By simply prompting the LLM from graph nodes, designers can guide the ideation process more effectively. Suh et al. introduced an interactive system designed to enhance ideation process by purposing a framework that emphasizes divergent thinking, allowing users to systematically explore a wide array of ideas [36].

Designers usually conduct interviews and ethnography to build personas to help themselves better understand users' needs in User-centered Design (UCD) activities. Zhang et al. introduces an innovative system designed to automate the creation of user personas by leveraging advanced technologies, including the GPT-4 model, DALL-E 2 model, and knowledge graphs [37]. This system is able to efficiently analyze and interpret user data to inform persona creation, create visual representations of personas, adding a tangible aspect to the profiles, and offer options for automatic and customized persona creation, accommodating various user needs and preferences.

While LLMs can improve and accelerate the process of exploration and ideation, they add an entirely new capability to nontechnical individuals that do not have any software skills. LLMs can operate as a text-to-code generator empowering users to embrace early prototyping without writing code themselves [38]. Users simply describe the digital product (e.g., a website) in natural language, which the LLM then turns into programming code (e.g., HTML code). This puts nontechnical users in a position to close the gap between conceptual work (i.e., ideas and concepts) and early look-and-feel-like prototypes that can be tested with users. DIDUP is a system for code-based UI prototyping that helps users generate interactive UIs in JavaScript, CSS, and HTML [39]. This system specializes in failure prevention and dynamic planning, taking changes and iterations into account that come up during the development process.

Code and Structured Specification Generation with LLMs

Generative AI has made crucial achievements in recent years in context understanding and code generation [40, 41, 42, 43]. This introduces a new pattern of application development in which multimodal large language models (MLLMs) directly convert visual designs into code implementations. LLMs can generate accurate web and UI code using multimodal inputs (e.g.,

screenshots, textual descriptions) and iterative refinement methods [41, 43].

Si et al. introduces a benchmark designed to evaluate the capability of MLLMs in converting visual webpage designs into functional code [40]. Among the models tested, GPT-4V demonstrated the highest performance in generating accurate and visually consistent code representations.

Wan et al. introduces DCGen, a novel framework designed to enhance the automation of translating webpage screenshots into functional UI code [41]. They identified prevalent issues in using LLMs for UI code generation, such as element omission, distortion, and misarrangement. They observed that focusing on smaller visual segments could help mitigate these problems. By addressing these issues, they purpose a divide-and-conquer-based approach by dividing screenshots into manageable segments initially, generating descriptions for each segment, and then reassembling them into complete UI code for the entire screenshot. This article highlights the effectiveness of segment-aware prompting in improving the accuracy and reliability of automated UI code generation from visual inputs.

LLMs have shown impressive capabilities in generating syntactically correct code but still encouter challenges when it comes to UI code generation. Firstly, they often struggle to recall and reproduce detailed visual elements, including precise color codes, typography, and spacing [44]. Secondly, their purely text-based training limits their ability to interpret and convert visual layouts into code, leading to misaligned grids and improper component hierarchies [45]. Moreover, without task-specific fine-tuning on multimodal datasets, these models tend to hallucinate CSS properties and generate non-standard design tokens, which require extra manual post-editing [42]. Finally, challenges such as context window constraints, lack of standardized evaluation metrics for UI quality, and difficulties integrating with established design systems further underline the need for dedicated research and fine-tuning in this domain [16].

However, this thesis focuses not on generating executable source code directly, but on producing JSON-based UI specifications. JSON here serves as a structured intermediate representation that describes layout and component properties, which can then be parsed and rendered into actual interfaces. This distinction ensures safety, interpretability, and adaptability compared to direct code generation approaches.

While many recent studies in UI generation focus on code generation from designs, there is a growing body of work emphasizing JSON-based or specification-based representations for UI definition. For example, SpecifyUI by Chen et al. introduces SPEC, a hierarchical intermediate representation that

allows designers to express UI intent through structured parameters, which is then rendered into actual UI [46]. Similarly, Json-GUI dynamically generates form-based web interfaces from JSON configuration objects, including features like validation rules [47]. Another related work by Chavarriaga uses JSON-DSL to support domain-specific web applications, defining behavior both server- and client-side via JSON specifications [48]. These works support the validity of using JSON as an intermediate specification layer rather than requiring direct code generation.

OpenAI API

OpenAI offers its users an Application Programming Interface (API) to state-of-the-art AI models for text generation, natural language processing, computer vision, and more [49]. It allows developers and companies to integrate into their digital products. The model that was used for this project work is the GPT-4.1 nano, which is the fastest and cheapest OpenAI model. It manages to deliver exceptional performance at a small size with its 1 million token context. It accepts both text and image inputs, and produces text outputs (including structured outputs). It is ideal for fine-tuning, and model outputs from a larger model like GPT-4.1 can be distilled to GPT-4.1 nano to produce similar results at lower cost and latency [50].

2.2 User Interface Personalization

UI personalization aims to tailor digital interfaces to individual users' preferences, behaviors, and contexts, thereby enhancing usability, satisfaction, and task efficiency. This section explores the evolution of UI personalization methodologies, including traditional rule-based systems, model-driven frameworks, and contemporary deep learning techniques. It also addresses the core challenges in achieving effective personalization.

Traditional Approaches

Traditional approaches to UI personalization primarily involve rule-based, heuristic, and manual customization methods. These methods allow direct manipulation by users or designers through manual settings or pre-defined rules at the widget level, facilitating immediate and understandable personalization options. Toolkits such as User Interface Façades enable end-users to manually configure UI elements according to personal preferences, thereby

enhancing usability through straightforward and transparent modifications [51]. Rule-based frameworks like those described by Leonidis et al. embed adaptation logic within UI components themselves, providing designer-driven logic but typically limited in scalability and adaptability [52]. Mixed-initiative systems blend user-driven customization with adaptive system interventions, leveraging cognitive modeling techniques such as Goals, Operators, Methods, and Selection rules (GOMS) to intelligently recommend adaptations while preserving user autonomy [53, 54].

Model-based Approaches

Model-based personalization utilizes explicit user, cognitive, behavioral, semantic, and contextual models to guide interface adaptations [4, 55]. Cognitive and behavioral models, including predictive HCI models and Markov chains, aim to predict user task performance and tailor the interface accordingly [56, 57]. These methods, while enhancing explainability and targeted adaptation, often require sophisticated modeling processes and may struggle to generalize across diverse contexts [4, 58]. Ontology and semantic-based models systematically adapt interfaces based on clearly defined user, device, and context profiles, which is particularly beneficial in accessibility-focused personalization [59, 60]. Nonetheless, they often necessitate extensive initial engineering and lack the dynamic adaptability afforded by data-driven approaches [55].

Deep Learning Approaches

Deep learning approaches use advanced neural architectures for end-to-end UI adaptation, providing significant improvements in personalization effectiveness and user satisfaction. Sequential deep learning techniques, such as those employing recurrent neural networks (RNN) and gated recurrent units (GRU), effectively model user interaction sequences to deliver real-time, contextually relevant UI recommendations, surpassing traditional prediction metrics [61]. Moreover, sophisticated generative models, including variational autoencoders (VAEs) and generative adversarial networks (GANs), dynamically generate personalized UI layouts and elements based on rich interaction histories, showcasing high adaptability and real-time responsiveness [6, 62]. Recent innovations integrate collaborative filtering with deep learning to refine predictions and personalization strategies further [61].

Reinforcement Learning (RL) and particularly deep reinforcement learning (Deep RL) methods have emerged as leading approaches, framing UI personalization as a sequential decision-making problem. Techniques such as model-based RL utilizing Monte Carlo tree search and predictive HCI models have demonstrated superior planning capabilities by anticipating user interactions and balancing adaptation costs like cognitive load and surprise [58]. Multi-agent reinforcement learning (MARL), exemplified by MARLUI, introduces simultaneous learning between simulated user agents and adaptive UI agents, effectively bridging the simulation-to-reality gap and enabling robust personalization policies without extensive pre-collected user data [63]. However, most deep learning methods require large datasets and are computationally expensive, limiting their practicality in resource-constrained environments or scenarios with limited user interaction data.

Challenges

Despite significant advancements, several challenges persist in UI personalization. Data sparsity and cold-start issues are prominent, as personalization effectiveness typically depends on extensive user interaction data, which may not always be available [58, 63]. Privacy and transparency concerns also limit widespread adoption, particularly with deep learning and RL methods that inherently lack explainability [55]. Long-term user engagement and evaluation represent further critical hurdles, as real-world deployment and sustained user adoption often yield results that differ significantly from offline or simulated evaluations [64]. Moreover, discrepancies between offline accuracy metrics and actual user experience outcomes necessitate more robust, longitudinal, and contextually embedded evaluation methodologies [65]. The balance between automation and user control remains delicate; overly adaptive systems risk undermining user trust, whereas too much manual intervention can degrade the personalization benefits [51, 55]. Addressing these multifaceted challenges requires ongoing refinement of personalization techniques and deeper integration of mixed-initiative strategies.

2.3 Related Work

In this section, we focus on two major aspects: approaches that incorporate user involvement in the UI personalization process, and prompt engineering techniques that enable more structured and effective UI outputs from LLMs.

User Involvement

Recent work has realized the importance of the user involvement during the process of UI generation to better align the interfaces created by LLMs with user preferences and needs. These methods are integrated with various forms of user input and feedback to enhance the controllability and usability of the interfaces.

Integrating User Preferences

CrowdGenUI presents a novel approach for aligning LLM-based UI generation with actual user preferences through crowdsourcing techniques [66]. The system collects preferences from diverse user groups and integrates this collective intelligence into the generative model. This approach significantly improves the usability and appeal of the LLM-generated interfaces by capturing preferences across different user demographics and contexts. The results of the study showed that the UIs generated with user preferences got higher usability ratings than those created using generic UI design principles alone.

Similarly, Huang et al. introduces a framework for guiding LLM reasoning with real user preferences. Their approach captures task-specific requirements and individual user needs, enabling the generation of UI widgets that more accurately reflect how users interact with interfaces [67]. The system implements a preference learning mechanism that builds user models over time, allowing the interface to adapt to changing user behaviors and requirements. This continuous learning capability helps bridge the gap between generic UI generation and truly personalized user experiences.

Interactive Modification and Control

Low-code approaches have emerged as an effective means to give users more control over the UI generation process. Low-code LLM, a novel human-LLM interaction framework was introduced to enhance the controllability and usability of LLM-based UI generation [68]. This framework provides users with visual tools to modify LLM outputs without requiring programming expertise. Users can manipulate UI elements via a graphical interface, while the system handles the translation between visual adjustments and the underlying code. This approach maintains the generative power of LLMs while making the process more accessible to non-technical users. The authors report a 47% reduction in time required to implement design changes

compared to traditional approaches.

Taking the concept of malleability further, Cao et al. proposed a system that supports real-time modification and extension of interfaces through both natural language instructions and direct manipulation [69]. A key innovation in this work is that user interactions are mapped to modifications in the underlying data model, which then drives dynamic interface updates. This creates a truly ”malleable” interface that evolves with user needs. The system maintains a consistent mental model between the user's intent and the interface representation, allowing for intuitive customization of complex interfaces without breaking functionality.

Iterative Refinement Through User Feedback

Incorporating user feedback to iteratively improve generated interfaces is another important direction in this field. Gaspar-Figueiredo et al. introduced an adaptive User Interface adaption framework which introduces a human feedback module that allows users to provide input on their preferences and satisfaction during interaction [70]. This feedback is integrated into the reward function of a RL agent, guiding it to learn adaptation strategies that better align with user expectations. The system demonstrates how continuous user feedback can lead to interfaces that evolve and improve over time, rather than remaining static after initial generation. Their user studies showed that interfaces refined through this feedback loop achieved 32% higher user satisfaction scores compared to non-adaptive interfaces.

Prompt Engineering Techniques for UI Generation

Constraint-Based Approaches

Constraints play an important role in guiding LLMs to generate UIs that meet specific requirements and limitations. Recent work explores how to effectively encode design constraints in prompts to control various aspects of the generated UIs, such as layout hierarchy, styling, component selection and structure.

A grammar-based approach is introduced to represent the hierarchical structure inherent in UI screens to guide UI generation [71]. This approach defines a formal UI grammar that represents aspects of the UI model with predefined patterns and rules reflecting common mobile design practices. By constraining LLM outputs to follow this grammar, the system produces more consistent and usable mobile interfaces. The grammar includes

specifications for component hierarchies, layout constraints, and interaction patterns, ensuring that generated UIs maintain proper structural relationships. Their evaluation showed that grammar-guided generation reduced layout errors compared to unconstrained LLM generation.

Similarly, Cao et al. discuss UI specification generation in their data model-driven approach [69]. Their system generates UI specifications based on the task-driven data model, which guides the composition of interface components and state management. This process incorporates predefined UI design patterns and rules to ensure stability and consistency in the generated interfaces. The specifications serve as an intermediate representation between the abstract data model and the concrete UI implementation, providing a structured way to reason about interface components and their relationships.

Intermediate Representations

Rather than generating UI code directly from user prompts, some approaches incorporate intermediate representations to help LLMs better understand and execute the generation task. Huang et al. introduce the concept of personas as an intermediate step in the generation process [67]. By first generating or selecting a user persona that matches the target use case, the system can tailor the UI more effectively to specific user groups. This approach adds a layer of abstraction that helps the LLM reason about user needs and expectations before attempting to generate the interface itself.

Other forms of intermediate representations include blueprints [72], wireframes [73], component trees [74], and design specifications [75]. These representations provide structured guidance to the LLM, breaking down the complex task of UI generation into manageable sub-tasks. By generating these intermediate artifacts first and then refining them into complete interfaces, LLMs can maintain better consistency and adhere more closely to design principles.

2.4 Summary

In this chapter, several areas related to this project were discussed, including the capabilities of LLMs, prompt engineering techniques, the application of LLMs in UI/UX design and UI development, the approaches to UI personalzation. Key literature were reviewed on how to generate personalized UI more effectively. These work focused on the tasks such as integrating user inputs, preferences, and feedback as well as implementing prompt engineering

techniques like merging UI grammar into prompts, incorporating intermediate representations.

A variety of approaches to UI personalization were reviewed, including traditional rule-based methods, model-based approaches, and deep learning-based techniques. These existing works primarily focused on enabling developers to generate UI components programmatically, rather than supporting real-time, on-the-fly personalization by end users. Furthermore, they often emphasized static or predefined customization paths rather than flexible, open-ended interactions.

Compared to traditional and model-based approaches, which typically rely on rigid design schemas and limited interaction flows, this project allows for higher degrees of freedom, both for end users in expressing their preferences and for the LLM in interpreting and generating responses. While deep learning methods can offer powerful personalization capabilities, they generally require extensive labeled data, long training times, and considerable computational resources. In contrast, the approach taken in this project utilizes a pre-trained LLM accessed via the OpenAI API. This enables low-cost, scalable interaction without the need for fine-tuning, making it more practical for lightweight prototyping and exploratory personalization use cases. By shifting the focus toward natural language interaction and real-time adaptability, this work aims to offer a novel perspective on user-driven UI personalization.

Chapter 3

Methods

3.1 Prototype Development

System Architecture Overview

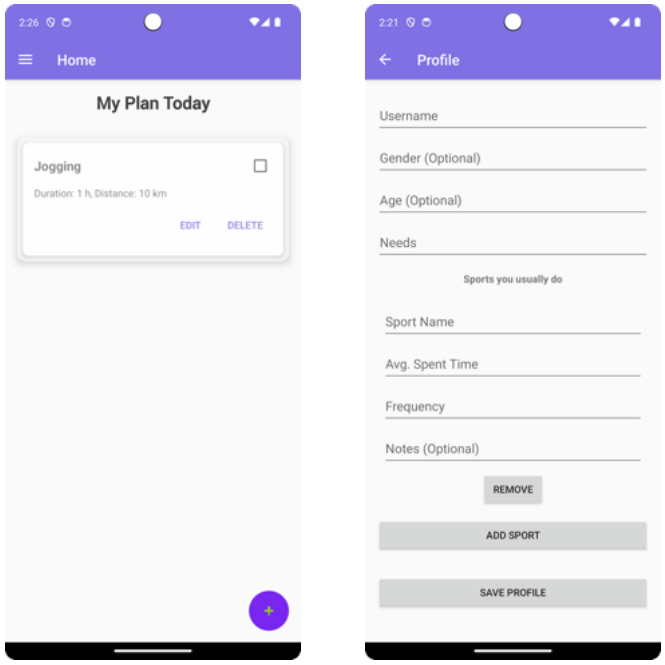
An android application named AdaptFit was developed for this project using Kotlin and integrated with OpenAI's GPT-4.1-nano model. It was specifically designed for user testing of LLM-driven UI personalization, which enables users to interact with and evaluate AI-generated UI personalization in a controlled environment.

The scenario focuses on fitness management where users can record their fitness goals. This particular scenario was selected because it offers opportunities to explore a broader range of UI interactions and use cases given the diverse fitness habits and preferences of individuals. Moreover, fitness goal management represents a common mobile application use case that users can easily understand and engage with during testing. This scenario not only aligns with the project's goals, but also remains relatively straightforward to implement. Plus, the GPT-4.1-nano model was chosen because of its lightweight size, fast inference speed, and suitability for mobile integration. It enables real-time interaction on resource-constrained environments like mobile phones.

The app comprises three main components:

- **Home Page:** As shown in Figure 3.1a, this page provides a standard, non-personalized UI that serves as the control condition in comparative studies. This enables users to perform core fitness planning tasks and establishes a benchmark for measuring personalization effectiveness.

- **Profile Page:** As shown in Figure 3.1b, this page collects user demographic information, accessibility needs, UI preferences, and behavioral data related to fitness activities. This component operationalizes the research question of how user involvement can inform UI personalization decisions.



(a) (b)

Figure 3.1: (a) Home Page. (b) Profile Page.

- **Proposal Page:** As shown in Figure 3.2 and Figure 3.3, this page implements the LLM-driven personalization process, generating customized interface proposals and rendering personalized UIs based on user profiles. This component represents the core experimental intervention being studied.

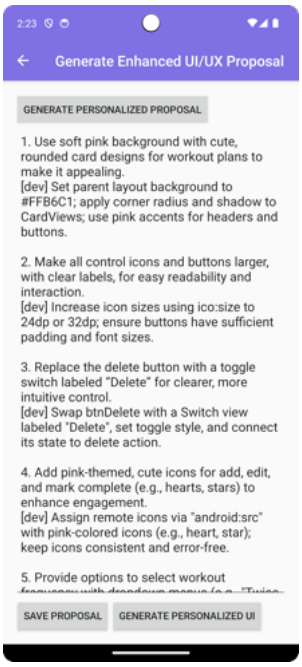


Figure 3.2: Proposal Page

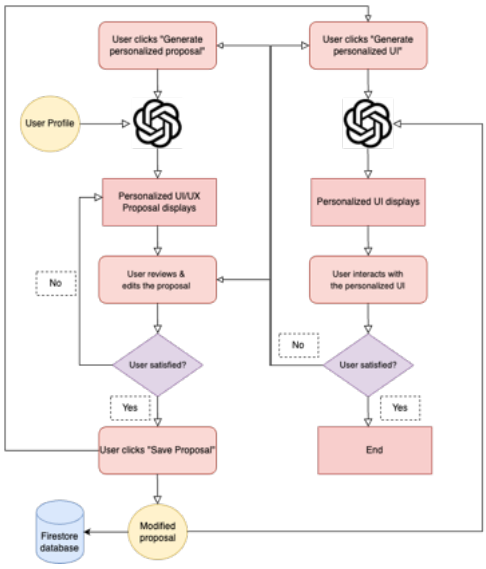


Figure 3.3: User Interaction Flow on the Proposal Page

Personalization Strategy

The core personalization strategy of the application is built around personalizing the UI to reflect each user's unique profile, workout habits, and needs. This strategy is implemented through a two-step process involving (1) proposal and UI JSON specification generation and (2) dynamic UI rendering. The proposal is served as an intermediate representations for LLMs to better understand users' needs and specify UI structures more effectively. On the other hand, it also helps users to have more control and awareness over this procedure. The overall workflow of the UI personalization process is illustrated in Figure 3.4.

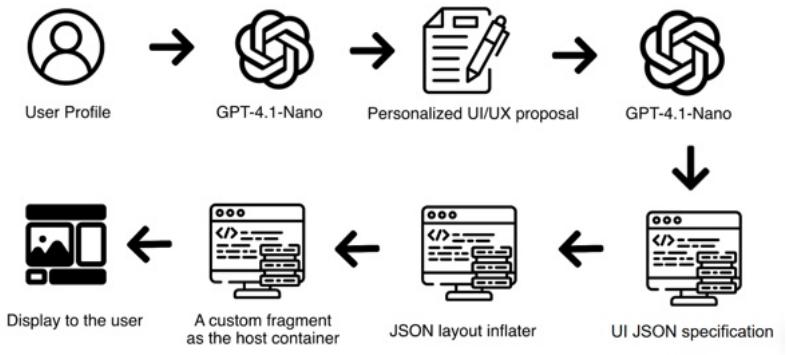


Figure 3.4: System workflow illustrating how user profiles are transformed into personalized UIs

Proposal & UI JSON specification Generation

This part includes two steps, beginning with the collection of user profile data through a profile form as shown in Figure 3.1b. This data includes the user's name, age, gender, UI preferences or specific accessibility needs, and a list of sports or fitness routines regularly performed. Each fitness routine record includes additional metadata such as frequency, duration, and special notes provided by the user.

Upon submission, this data is passed as input to the model to generate a personalized UI/UX proposal, a descriptive natural language proposal detailing suggested improvements or additions to the user's homepage layout. The prompt given to the LLM is crafted to refine and enhance the UI of AdaptFit, ensuring it aligns with user needs. (See Appendix A.2 for the full prompts.)

The second step involves transforming the generated proposal from last step into a UI JSON specification. The user initiates this step by selecting

"Generate Personalized UI," which triggers a second LLM prompt designed to convert the proposal into a JSON-based representation of Android layout components.

It is important to clarify that, in this system, the LLM does not generate executable source code (e.g., HTML, XML, or Kotlin). Instead, the model outputs a JSON-based intermediate representation that encodes layout structures, component attributes, and constraints (see examples in A.5). While JSON is not code in a strict sense, it functions here as a structured specification language that can be parsed by our `JsonLayoutInflater` into actual UI components at runtime. This distinction ensures that the generated output remains safe, lightweight, and easily interpretable, while still allowing the flexibility of dynamic UI generation. The reason why the LLM is required to return JSON instead of XML is because Android's LayoutInflater only works with precompiled layout resources which cannot inflate dynamic XML at runtime.

The chosen prompt engineering techniques are employed during this process:

- **Expert Persona:** The prompt clearly defines the model's role as *"a professional UI/UX designer specializing in fitness applications"* when generating a proposal according to the user profile and *"a professional Android developer specializing in UI development"* for UI JSON specification generation. This strategy ensures generated suggestions remain relevant and professional.

- **UI Grammar**: The prompt explicitly references the existing XML layout structures, requiring the LLM to adhere closely to predefined functional structures. This ensures compatibility and consistency in the generated UI.

- **Scope Limitation:** Suggestions generated by the model are constrained to address only layout, style, and control properties, avoiding modifications related to backend logic, navigation, or additional functionality outside the UI layer.

- **Output Format:** The model is instructed to generate proposals in a clear, numbered list format, each containing user-facing descriptions limited to approximately 15 words, followed by developer-facing implementation hints. This format is designed to bridge the gap between end users and the code generation process. The user-facing descriptions provide a concise, understandable overview of proposed UI

elements, requiring no technical background. In parallel, the developer-facing implementation hints ensure that downstream code generation is grounded in actionable, structured guidance. By separating concerns in this way, the prompt facilitates both effective communication with users and accurate, personalized UI synthesis in the next generation step.

- **Chain-of-Thought**: The prompt instructs the model to first analyze user data thoroughly, identifying and clearly articulating specific user needs before generating structured UI recommendations. This approach enhances logical coherence and relevance of the suggestions.

The resulting UI/UX proposal can then be reviewed and edited by users. Once finalized, the proposal is stored in Firebase Firestore for subsequent use and reference.

Dynamic UI Rendering

This JSON object returned from LLM follows a strict schema, mapping closely to the structure of Android XML layout files but allowing for dynamic modification. It includes definitions for all primary UI components such as TextView, CardView, RecyclerView, and TextInputLayout, along with layout constraints and styling attributes.

The application employs a custom-built `JsonLayoutInflater` to parse this JSON and render the interface at runtime. This enables the app to display tailored layouts for each user without requiring any static XML modification or app recompilation. A custom Fragment (`EnhancedHomeFragment`) is implemented as the host container to attach application-specific logic or behavior to the dynamically generated Views (e.g., setting click listeners, binding data, handling UI updates). In other words, these two files together bridge the gap between structure and behavior, turning static JSON into interactive UI components with logic. See A.3 for the pseudocode to understand the system workflow.

This strategy ensures that users not only receive personalized content but also interact with an interface that reflects their specific needs and preferences. It balances flexibility and scalability by separating data-driven customization from the underlying business logic of the application.

Example Results

To illustrate the outcomes of personalized UIs generated by the system, ten example results are shown in Figure 3.5. These examples demonstrate how

user preferences, expressed via natural language prompts, are translated into customized interface layouts. In (a), the user requested a simple and intuitive interface with icons. In (b), the user requested bold fonts and visual elements to better support their attention deficit hyperactivity disorder (ADHD). In (c), the user requested a simple UI with a visual element that can be marked as completed. In (d), the user requested both a timetable and unique icons for each exercise plan. In (e), the user requested a clean and intuitive UI with mild colors that are not strong for the eyes. In (f), the user requested a pink and cute interface with large, readable elements and no extra icons. In (g), the user requested a simple UI with icons. In (h), the user requested an intuitive interface with ground color. In (i), the user requested an interface that includes a motivational quote and a motivational picture. In (j), the user requested a simple and readable interface with reasonably sized icons and a stylish design.

More example generated results including LLM-generated proposals and UI JSON specification can also be found in A.5.

(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

Figure 3.5: Ten personalized UI examples generated during user testing.

3.2 Usability Testing

Usability testing is a research method following standard guidelines as defined by Nielsen to test the functionality of an interface by observing real users when they attempt to complete tasks on it [76]. The goals of usability testing are to recognize problems in the design of the product, discover chances for improvement, learn about the user's behavior and preferences.

The testing process was divided into the following stages:

Recruitment Phase

Six participants were recruited through convenience sampling, primarily drawn from the researcher's academic and social network. All participants

had higher education backgrounds, and four of them possessed programming experience. Convenience sampling, defined as the practice of selecting readily available participants within reach of the researcher, was chosen due to time and resource constraints [77]. While this approach enabled accessible and efficient recruitment, it also shaped the participant profile toward a more technically literate group. A more detailed analysis of participant demographics and how these characteristics influenced the study outcomes is presented in Chapter 4.1.

Pre-Test Phase

This phase involved three main components: obtaining consent, briefing participants, and administering a pre-test questionnaire. The consent form and questionnaire can be found in A.1.

Participants were first presented with a digital consent form outlining the purpose of the study, the testing procedure, the data collection process (including audio recording), and how their data would be stored. They were informed that all responses would remain confidential and would only be used for research purposes. Participants had the opportunity to ask questions before agreeing to take part in the study.

After consent was obtained, participants completed a digital pre-test questionnaire designed to collect demographic data and insights into their digital habits and preferences. The questionnaire included items on: demographic information, comfort level with using new mobile apps, accessibility needs or UI/UX preference, fitness habits, prior experience with fitness management apps, and comfort level with the use of AI.

Test Phase

During the usability testing, each participant was provided with an Android phone where the application had been pre-installed. Participants were given a task instruction (See A.1) with four tasks including interacting with initial UI, input information for personalized UI, interact with personalized UI, and iterate. As the participant performed these tasks, the researcher observed the participant's behavior and listened for feedback. The tasks were designed to let participants experience both baseline UI and LLM-generated personalized one. The goal of the test was to find out the user experience with LLM-generated personalized UI, the importance of user involvement during this process, the ease and effectiveness of UI personalization, and the capabilities

and limitations of LLMs.

Post-Test Phase

This phase involved two main steps: administering a post-test questionnaire and conducting a semi-structured interview. The post-test questionnaire and interview questions can be found in A.1. The goal of this post-test was to collect both quantitative and qualitative feedback regarding participants' experiences with the LLM-generated personalized UI, as well as to assess the system's usability and perceived level of user involvement.

The first part of the post-test questionnaire includes multiple-choice and Likert-scale questions covering the following aspects: the number of iterations (i.e., how many times participants generated a new version of the enhanced UI); participants' goals behind each iteration (e.g., exploring different styles or improving specific functionalities); perceived improvement across iterations; criteria for selecting a satisfactory version; sense of involvement in shaping the final UI; desire for more control during the design generation process. Likert-scale questions measure participants' attitudes or perceptions by asking them to indicate their level of agreement or disagreement with specific statements, typically on a 5-point or 7-point scale [78]. It aims at understanding users' intent, behavior, satisfaction and their experience. By analyzing their iteration behavior, our goal is to understand better how the users interact with the personalized UI system and what factors affect their decision during the refinement.

The second part of the questionnaire is the System Usability Scale (SUS), a series of 10 Likert-scale questions giving a comprehensive view of subjective assessments of usability [79]. By asking participants to rate their agreement with statements such as "I found the system unnecessarily complex" or "I felt very confident using the system," on a five-point Likert scale, the scale provides a quick measure of usability. The scale is validated by large amount of industry-wide data, which is available to help benchmark the score and understand it in context of the other similar systems [80].

To grasp richer qualitative insights, participants also took part in an 8–10 minute semi-structured interview. The interview aimed to get deeper understandings on the user experience, particularly the perceived advantages and limitations of the enhanced UI compared to the baseline version. Open-end questions are designed to enable participants express their thoughts, report on usability issues that are not captured by the questionnaire, and suggest improvements from the users' perspectives. Together with the questionnaire,

the phase was designed to help form a comprehensive understanding of the users' experiences and expectations when interacting with the personalized UI system.

3.3 Planned Data Analysis

The usability testings yielded both quantitative and qualitative data. The analysis was planned as follows.

Quantitative Data Analysis

The quantitative data collected from the questionnaires was analyzed to reveal general user experience and measure overall system usability. Basic descriptive statistics (e.g., mean) were used to summarize participants' responses to each question, including iteration counts, perceived improvement ratings, and involvement levels. Responses to the SUS items were scored according to standard SUS procedures. Each participant's raw score was converted to a usability score ranging from 0 to 100. These scores were used to assess the perceived usability of the enhanced UI and benchmark it against accepted usability thresholds (e.g., a SUS score of 68 is considered average usability) [81]. In addition to the SUS, this study also employed the Net Promoter Score (NPS) to evaluate user loyalty and overall satisfaction [82]. NPS was measured by simply asking a question: "How likely are you to recommend this system to others?" According to the responses, participants were categorized into three groups:

- Promoters (scores 4-5) are highly satisfied participants who are passionate about the product. They are more likely to recommend it to others.
- Passives (scores 3) are moderately satisfied participants but not enthusiastic. They are unlikely to recommend the product. While not negative, they are also not strong advocates.
- Detractors (scores 0-2) are dissatisfied participants who might have had negative experiences. They are unlikely to recommend the product and might even discourage others from using it.

Together these two metrics offered a holistic understanding of the user experience. This study also includes a structured analysis of user iteration

behaviors during the user testing. A detailed record was kept for each iteration, including the specific actions performed by the user (e.g., editing needs, generating UI/proposal), if the UI was successfully deployed, the prompt specificity, any usability issues encountered, the user's subjective satisfaction with the outcome, and observational notes regarding their interactions. The iteration patterns and user satisfaction can be identified by coding and visualizing these iteration records. It also offer the insights into the effectiveness and challenges of the system in supporting UI personalization on the fly.

Qualitative Data Analysis

The qualitative data, including the observational data (screen recordings, notes on participants' behaviors, system interaction logs) and the audio interview transcripts, were analyzed through thematic analysis using ATLAS.ti [83]. Thematic analysis is a widely used, systematic method of qualitative data analysis, which breaks down and organizes data by identifying common themes or patterns [84]. It involves careful reading and interpretation of the material to extract meaning and an inductive coding approach is used to identify the common segments [85]. Codes are iteratively grouped into higher-level themes that reflect shared experiences or divergent views [86]. Themes are interpreted in light of the research questions and contextualized using quotes from participants. Findings from the interviews are also triangulated with quantitative results to enhance validity.

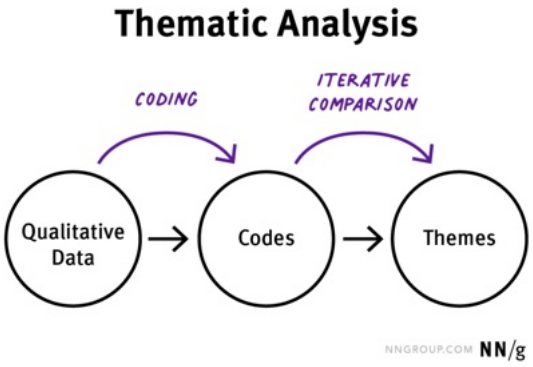


Figure 3.6: Thematic Analysis [86]

Chapter 4

Results and Analysis

In this chapter, we present and discuss the results based on data analysis.

4.1 Participant Demographics

A total of six participants (4 female, 2 male) were recruited for the study through convenience sampling. Their ages ranged from 18 to 30 years old. All sessions were conducted in English, with two participants being native English speakers and four non-native speakers. Each session lasted between 20–30 minutes.

Background Classification

To analyze potential bias from technical expertise, participants were classified using a coding system where aliases are followed by letters indicating their background: D (design/UI experience) and P (programming experience).

Participant Alice and Diana represent the general user population with no programming or design experience. BethP works in autonomous systems and has programming experience but no UI development background. CharlieP studies physics and has programming experience without UI specialization. EveDP works as a full-stack developer with both programming and UI development experience. FrankP has a data science/machine learning background with programming experience but no UI development. This distribution resulted in four participants with programming experience (BethP, CharlieP, EveDP, FrankP) and only one with both design and programming experience (EveDP), compared to two participants (Alice and Diana) representing typical end users without technical background.

All participants reported experience using AI tools in various contexts, though none had used AI-powered UI generation specifically. Three participants (BethP, EveDP, FrankP) had prior experience with fitness applications. All participants demonstrated basic to advanced smartphone proficiency, suggesting comfort with mobile interfaces generally.

Different Patterns among Participants

The technical background classification revealed different patterns in how participants approached and evaluated the AI-generated personalization. Participants with programming experience tended to think from the perspective of how the model would fulfill specific requirements. For example, when CharlieP requested the UI to display an image of a dog and the model failed to retrieve a valid image, CharlieP modified the proposal to include a correct image URL that the model could use.

A notable trend among participants who have programming background was their "get things done" attitude. They actively sought to make a specific requirement work, iteratively modifying the proposal in an effort to realize their intended UI. For instance, FrankP wanted the displayed workout plans to arrange multi-line text more reasonably and repeatedly adjusted the proposal to achieve a better arrangement. Similarly, BethP tried to make the dialog box provide different options for different sports; they continuously refined their input by adding more details, aiming to help the model understand and implement the request.

In contrast, participants without a background in programming or design were less concerned with whether a particular feature was implemented. They tended to treat the system more as a playful experience, moving on to the next idea when a request was not fulfilled. For example, when Alice initially asked for a UI that supported selecting different font styles, the generated UI did provide multiple font options. However, due to structural limitations, these options existed only visually and did not trigger any corresponding actions. Alice did not dwell on the unimplemented functionality; instead, Alice removed the request and moved on to their next exploration. Similarly, Diana attempted to implement a Times New Roman font style, but upon finding that it was not rendered as expected, Diana also dropped the request and continued experimenting.

EveDP, possessing both design and programming experience, was better equipped to revise the model's proposals to meet their expectations. For instance, when the model failed to implement auto-filling the user's common

sports activity, EveDP directly inserted a placeholder (`android:text: <my common sport>`) into the proposal to indicate where changes were needed. This direct intervention helped reduce hallucinations and guided the model toward more accurate outputs.

Technical Bias

The overrepresentation of programming-experienced participants likely influenced study outcomes in several ways. These participants brought technical implementation awareness that typical users would not possess, potentially leading to more critical evaluations of AI-generated suggestions. Their feedback may reflect technical feasibility concerns rather than pure user experience preferences, creating a bias toward technically sound but potentially less innovative personalization approaches. The single participant with design experience (EveDP) provided the most comprehensive feedback compared to others, showing how professional expertise can significantly alter evaluation patterns. This suggests that findings may better represent technically literate users rather than the broader population of fitness application users.

4.2 Quantitative Results

System Usability Scale (SUS)

In this study, the SUS was employed to quantitatively evaluate the usability of the AI-generated UI personalization system. SUS scores range from 0 to 100, with a highly acknowledged industry benchmark average of 68. The calculated average SUS score of this study across the six participants was 69.17. The individual SUS scores of each participant are shown in Figure 4.1.

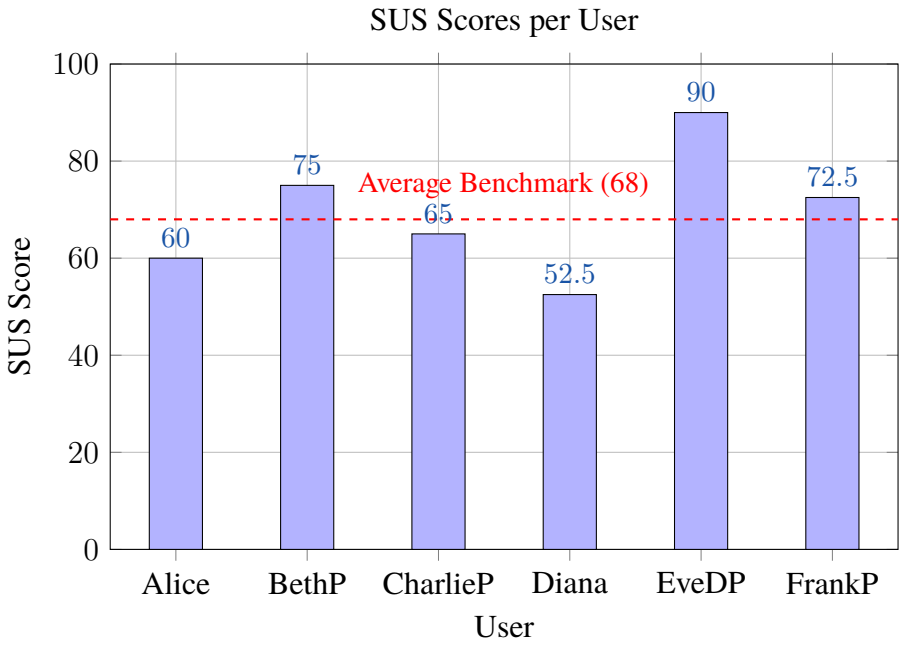


Figure 4.1: SUS Scores for all participants.

It reflects that the system slightly above the general usability acceptance threshold. But it also reveals room for improvement. One noticeable pattern is that participants with programming experience reported higher usability scores (75, 65, 90, 72.5) compared to those without programming experience (60, 52.5). Notably, the participant (EveDP) with both programming and UI design experience rated the system highest with a SUS score of 90. This suggests that prior expertise may facilitate more effective interaction with the system.

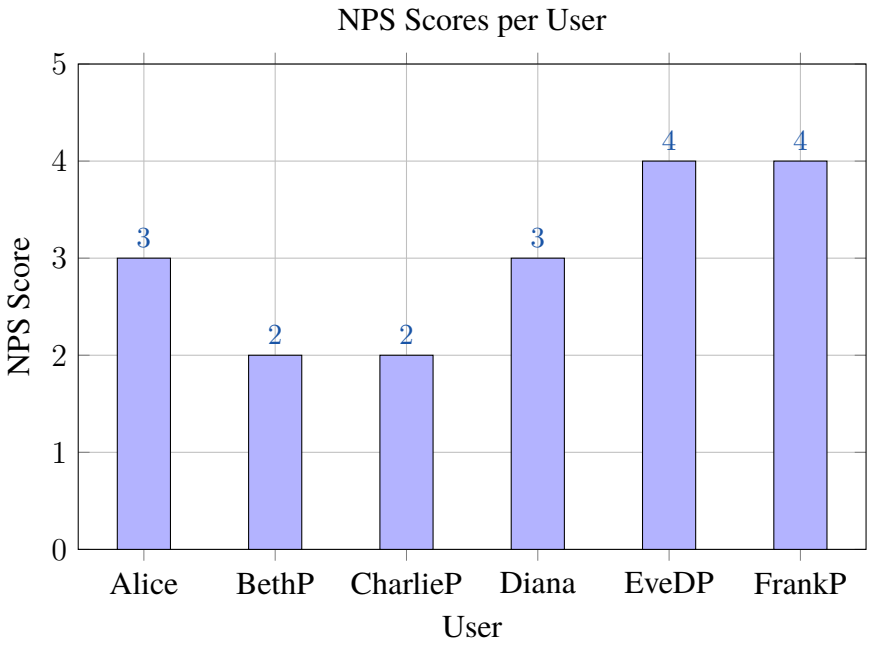


Figure 4.2: NPS Scores for all participants.

A more detailed analysis combining the NPS score (the recommendation likelihood of the participants) revealed some nuanced insights as follows. The individual NPS scores of each participant are shown in Figure 4.2.

- Promoters (NPS scores 4-5, participants likely to recommend the system): The promoters achieved an average SUS score of 81.25, presenting excellent usability. These participants frequently expressed their requirements clearly and iteratively refined their needs. This led to a more capable system to deliver tailored, personalized UIs effectively.

- Passives (NPS scores 3, participants hold a neutral stance): The passives had an average SUS score of 56.25, categorizing the usability as below average. This lower score might result from unclear or insufficiently articulated participant inputs. This demonstrates that participants' inability or hesitation to specify precise needs could negatively affect the usability of the system.

- Detractors (NPS scores 0-2, participants unlikely to recommend): The detractors had relatively high SUS score (average of 70.00), indicating acceptable usability. The reason why they are not likely to recommend might stem from mismatches between expectations set by

generated proposals and the actual implementation outcomes. This also demonstrates that fulfilling technical usability alone might not guarantee the overall satisfaction.

Questionnaire Analysis

This part includes the analysis from both the pre-test and post-test questionnaires. Before testing, most participants expected simple, intuitive UIs with personalization option. In terms of the post-test questionnaires, the results indicate that:

- Five out of six participants iterated (generate a new version of the enhanced UI) more than five times. Most participants were deliberately exploratory, aiming to improve specific functional areas or explore different styles, while the others were more passive by simply clicking "Generate" button a few times and relying on the model to make decisions for them.
- However, only two participants felt their UI improved with iterations, while three participants perceived no improvement, and one was unsure. The potential issues might be related to the ability of the system to fulfill the requests from the participants, the bias of LLMs, the ability of the participants to communicate their preferences effectively, and the feedback loop between participant input and system output.
- Half of the users wanted more control over the design generation process, while two of them were satisfied with system autonomy, and one was unsure. All of the participants reported moderate involvement during the process.

Iteration Patterns and Outcomes

Six participants demonstrated various iteration patterns as shown in table 4.1. Figure 4.3 illustrates the iteration behaviors of all six participants during the UI generation process. Each participant had distinct requirements, and their interactions with the system were tracked across multiple iterations. The chart captures four key aspects: the type of iteration performed (e.g., editing profile or proposal, generating proposals, or directly generating UI), the specificity of the prompt, whether the UI was successfully deployed, and the participant's subjective satisfaction.

Table 4.1: Iteration Patterns of each participant

Participant	Iteration Count	Iteration Pattern
Alice	5	Moderate exploration
BethP	6	Extended exploration
CharlieP	6	Persistent problem-solving
Diana	8	Highest engagement
EveDP	8	Extensive refinement
FrankP	9	Most persistent

Figure 4.3: Overview of participants' iteration behaviors and outcomes.

Only one of 42 iterations failed to deploy UI (Alice's first attempt), while 24 of 42 successful UIs still had usability issues, such as icon rendering failures, component parsing errors, layout inconsistencies, resource loading failures. These issues were likely caused by underspecified prompts as indicated by yellow bars beneath the iteration icons, LLM hallucination where the model generate non-existent resources or invalid code structures, incompatibility between LLM-generated code and the predefined system framework's parsing capabilities, and inherent system limitations in handling complex UI components and external resource integration.

Participant satisfaction varied significantly across iterations. Fully specified prompts (green underline) were more likely to yield usable and satisfying UIs. In contrast, vague or underspecified prompts often led to

low satisfaction or ambiguous outcomes, as seen in several iterations with neutral or negative facial icons. This suggests a strong correlation between prompt clarity and successful user experience outcomes. However, although fully specified prompts generally produced better results but revealed system limitations when implementing complex features.

In terms of iteration types, participants employed four different approaches. The most comprehensive approach, "Edit Needs + Generate Proposal + Generate UI" (55% of iterations), was most effective for implementing significant changes but carried the highest risk of technical failures due to complexity. Simple "Generate UI" iterations (19%) offered the safest path with fewer parsing errors, making them ideal for incremental refinements, while "Edit Proposal + Generate UI" (12%) provided a middle ground for targeted adjustments, though sometimes suffered from misinterpretation issues. The "Generate Proposal + Generate UI" approach (14%) produced unpredictable results but occasionally yielded unexpected improvements.

Besides, participants struggled to express their design preferences effectively:

- BethP's "make font fatter" was misinterpreted as "bold" instead of "rounded" which was the intended meaning.
- FrankP's "words should not be too intense at the same line" wasn't understood by the system
- Vague terms like "prettier", "cute", and "fancy" often led to unsatisfying results since these participants typically lacked a clear vision of what they wanted. Consequently, the output generated by the LLM based on these ambiguous requests often failed to fulfill their needs.

Approximately 35% of the iterations led to satisfying results, 30% resulted in unsatisfying outcomes, and the remaining iteration had uncertain results. In most cases, simple and well-defined requests (colors, fonts) were most successfully implemented while complicated functional features (timetables, sport-specific options, autofill) were inconsistently delivered. In general, visual features were easier to achieve than functional enhancements.

Additionally, participants developed different coping strategies during the process. For example, Diana and EveDP made small, targeted changes while participants CharlieP and FrankP repeated similar requests hoping for better results. Some users learned to add clarifying details about errors to prevent

recurrence. For example, after encountering an image that did not match the intended concept, CharlieP refined their prompt by explicitly requesting the system to retrieve an image from Wikipedia. A trend was observed in the study: when complex requests failed to generate the desired outcomes, participants often reverted to simpler requests to regain control. In some cases, participants chose to accept partially working solutions rather than continue iterating, especially when the issues seemed minor or difficult to resolve. Besides, some participants developed their own workarounds for recurring technical problems. For instance, when icon rendering frequently failed, Diana and FrankP explicitly added requests in their prompts to exclude generating icons to prevent further issues.

4.3 Qualitative Results

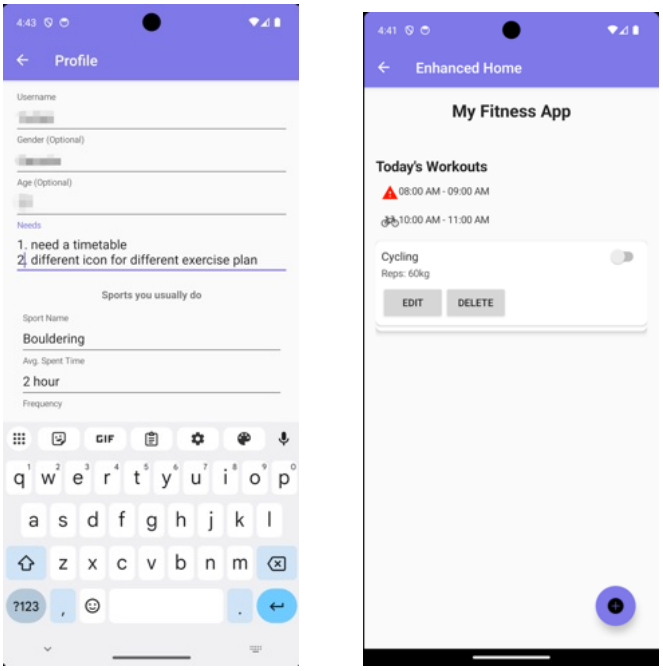
The thematic analysis underscores eight primary themes that represent users' experiences with AI-generated UI personalization based on the observation notes and post-test interview transcripts. To comprehensively illustrate user experiences, the eight themes are grouped into three categories: **positive user experiences**, **challenges and frustrations**, and **suggestions for improvement**. The complete results can be found in A.4.

Positive User Experiences

Positive Reception of the Concept

Despite some implementation barriers, participants were enthusiastic to experience AI-generated personalized UIs.

Alice, who had no prior technical background, initially approached the system out of curiosity rather than specific expectations. When the system successfully generated a personalized layout for Alice, rearranging workout cards from a list into a grid format to add a timetable as they requested, Alice expressed genuine excitement and described the tool as *"very novel"* (see Figure 4.4).



(a) (b)

Figure 4.4: (a) Alice specified their personalized UI needs, requesting an integrated timetable feature. (b) The resulting UI generated by the system.

Similarly, FrankP also appreciated the distinct novelty of the tool, saying that they had never previously used an interactive system capable of dynamically generating UIs with the LLMs.

Diana also appreciated the fundamental idea behind the system, stating, *"I feel like the concept in and of itself—the idea—is good."* They specifically emphasized the advantage of AI intervention in reducing cognitive load. Diana explained that they only needed to provide initial keywords, after which the AI expanded upon the ideas with additional suggestions and creative extensions. BethP echoed this sentiment, praising the concept of a UI continuously evolving based on user needs as *"really good"*.

CharlieP encountered a scenario where they requested the model to insert a dog image into the UI, but the model instead provided an unrelated image. Despite this mismatch, CharlieP still expressed satisfaction, noting positively, *"I'm happy the system managed to pull out some image, which demonstrated that it could do it in theory, and if it just found the right URL, it would be perfect."*

Benefits of Personalized Design

When successfully implemented, personalized UI enhanced user motivation and engagement with the system. BethP emphasized that AI-assisted personalization was particularly beneficial, as it could be tailored to match a user's personal habits, thereby creating a sense of *"intimacy"* between the user and the application. They further noted that if the other applications offered such customization and maintained consistency across them, it would make their daily usage far more convenient.

Diana shared that personalization would motivate them to check the app more frequently, explaining that they would *"want to change it all the time,"* which in turn would encourage them to use the application more often.

Alice requested that the AI generate a "mark complete" feature for their workout plans. Alice explained that this feature would enhance their motivation, because marking a task complete after finishing a workout would offer sense of accomplishment and reinforce their engagement with the application.

Enjoyment from Co-Creation

When the system worked well, participants experiences significant satisfaction from the collaborative design process with the LLM. This theme shows how co-creation with AI can create a sense of playfulness, curiosity, and personal investment, turning the design process into an engaging and motivating experience for users.

Diana described generating new UIs was like *"a little game."* Even though the outputs did not always work as expected, they found the process engaging and playful.

FrankP also valued the sense of accomplishment and ownership that came from co-creating a UI with the AI, stating, *"it gives me a sense of accomplishment and fulfills my need for personalization."*

Alice referred to the experience as a *"curious journey,"* expressing intrigue about how the AI would interpret and implement their requests. Alice was not only interested in seeing whether the AI would fulfill their requests but also curious about when and why it might overlook them.

Challenges and Frustrations

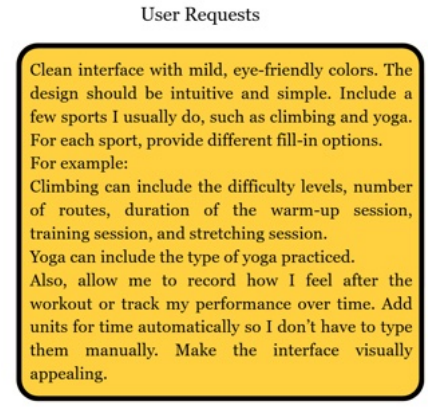
Mismatch Between Proposal and UI

Participants frequently experienced disconnection between what the AI proposed and what was actually implemented in the enhanced interface.

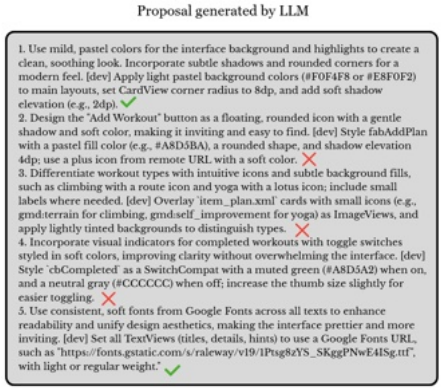
BethP's primary request was for a clean and intuitive UI design with eye-friendly colors, along with different fill-in options for the exercises they regularly performed (see Figure 4.5a). The LLM-generated proposal (Figure 4.5b) included five suggestions such as adjusting colors, buttons, and font styles, as well as adding icons and labels for different workouts. However, the final rendered UI did not implement suggestions 2, 3, and 4 from the proposal.

A closer inspection of the system logs revealed that the LLM did in fact attempt to implement the second suggestion by adding an icon and color to the "Add Workout" button. Yet, due to hallucination issues, the model generated an invalid icon URL. As a result, the button failed to display the intended icon and instead automatically fell back to a default error icon (see Figure 4.5c). Regarding the third and fourth suggestions, the logs also showed that the LLM attempted to realize them by adding new icons and labels. However, due to the system's inherent framework constraints, the newly introduced elements disrupted the existing layout, causing the "Add Workout Plan" dialog to fail to load properly. Figure 4.5c shows that the final UI applied the first and the fifth suggestions by changing background colors, styles and fonts. However, when BethP tried to add workout plans by clicking the floating button, the dialog form failed to appear and an error message — "Exercise Name is required" — was displayed, as shown in the figure.

From BethP's perspective as a user, they had no visibility into these backend attempts or technical failures. All they experienced was that the final UI fell short of the proposal. They expressed their frustration by stating, *"It generates too many things, but it doesn't implement over half of them."*



(a) BethP's requests



(b) The LLM-generated proposal

(c) The final UI

Frustration over Unmet Needs

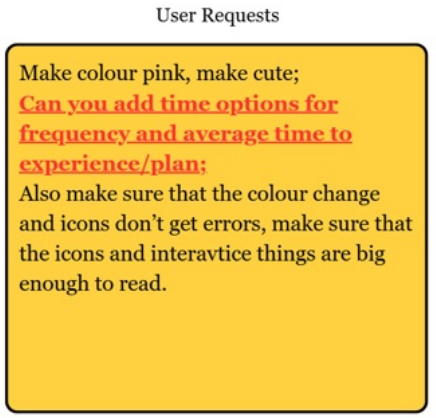
Participants felt frustrated when their needs didn't get fulfilled, particularly when the system seemed to understand their requirements but failed to deliver.

Diana clearly articulated their request for time-related options in the add workout plan dialog, fields for both workout frequency and average duration (see highlighted sections in Figure 4.6a). In the LLM-generated proposals, this request was explicitly addressed only in iterations 1, 2, 5, and 6, where the suggested design included a spinner dropdown for workout frequency (e.g., "Twice a month") and a separate input field for average duration (see highlighted proposal in Figure 4.6b).

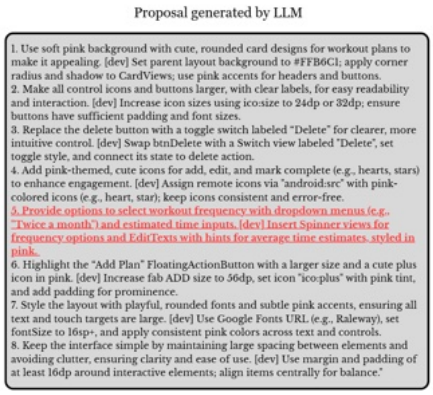
However, the actual UIs generated often failed to implement these suggestions. In the first four iterations, no elements for frequency or duration

were included at all. In the fifth and sixth iterations, partial implementation appeared: the system completely omitted the spinner for frequency, while the average duration field was merged into the general "Duration" input, indicated only by a "(mins)" hint (see Figure 4.6c for the sixth iteration). In effect, this was a simplified version that only partially touched on Diana's idea, without fully realizing the independent fields specified in the proposal.

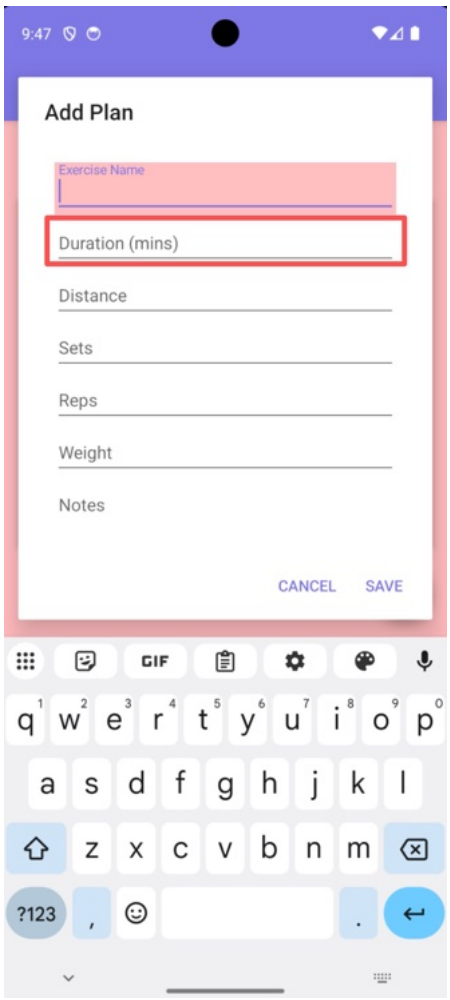
From the user's perspective, this inconsistency led to their disappointment. Diana remarked, *"Sometimes the system didn't follow my requests, which made me feel disappointed."* Yet, when the system eventually reflected their input more in later iterations, she acknowledged the improvement, stating, *"If it generates according to my needs, I think it's quite good."*



(a) Diana's requests



(b) The LLM-generated proposal at the sixth iteration



(c) The final UI at the sixth iteration

System Constraints & LLM Gap

This theme demonstrated several technical limitations and inconsistencies during the personalization process, showing a disconnect between users expectations and capabilities of the AI system.

Many users reported encountering several technical issues. The most common ones are icons rendering issues (as shown in the Figure 4.7a, 4.7b, 4.7c, 4.7d, 4.7g, 4.7i and layout issues (as shown in the Figure 4.7d, 4.7e, 4.7f). Besides, due to parsing issues, some pages or the Add Plan dialog cannot be displayed properly (Figures 4.7h, 4.7i, 4.7j).

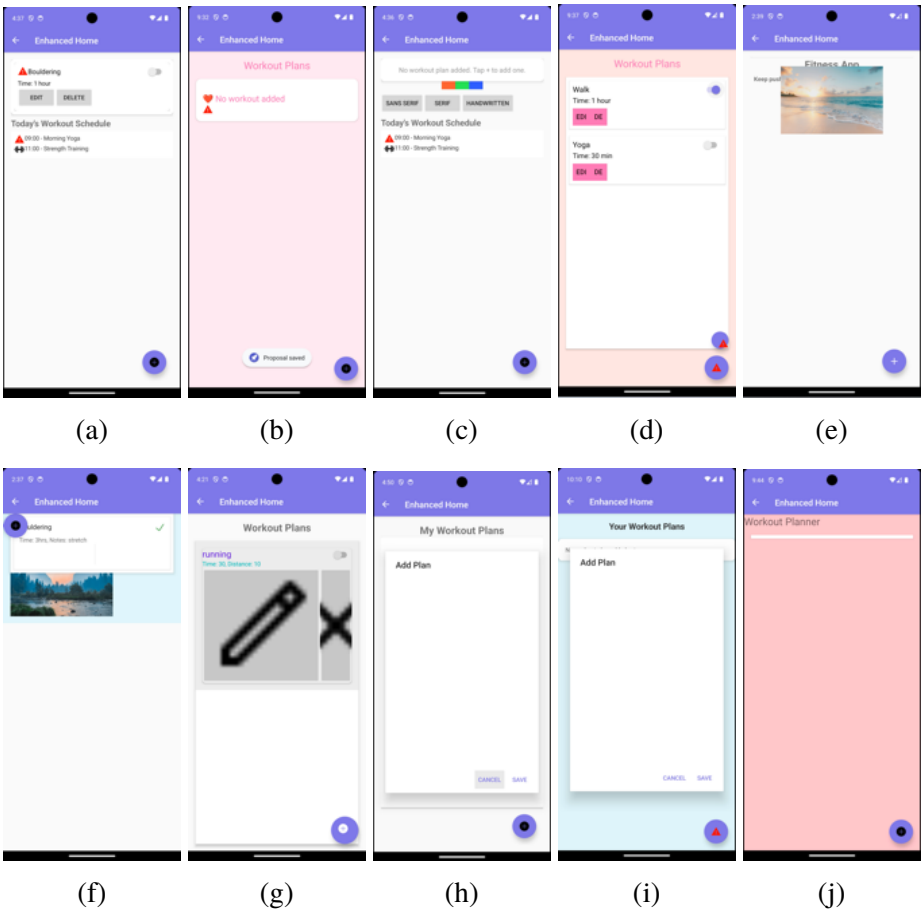


Figure 4.7: Examples of technical issues encountered during UI generation. (a–c, g, i) Icon rendering issues, such as invalid, broken or oversized icons. (d–f) Layout problems leading to misaligned or overlapped components. (h–j) Parsing failures causing incomplete pages or non-functional “Add Plan” dialog.

Participants also encountered issues where the LLM had hallucinated or over-promised outputs. For example, when CharlieP requested a dog image to be added to the UI, the system log revealed that the model generated links either pointing to unrelated images or to entirely made-up resources that did not exist. Similar problems appeared with font resources. When asked for *"fancy fonts"*, the LLM generated placeholder URLs, and when Diana requested *"Times New Roman"* font style, the system generated a link in Google Fonts' format, but Times New Roman is not actually available on Google Fonts. In another case, Diana noticed that the proposal included *"animated buttons"*, but this feature was never implemented in the final UI.

Suggestions for Improvement

Lack of Guidance & Prompting Support

Participants consistently struggled with constructing effective inputs for the AI system. Some are uncertain about how to communicate their needs effectively. This theme emerged as one of the most significant barriers to successful interaction. Some key findings are as follows.

FrankP struggled with constructing effective inputs for the AI system. They expressed that they felt lost when trying to expressing their requests due to their lack of vocabulary and familiarity of UI concepts. As a result, FrankP's requests tended to remain vague and generic, such as asking for a design that was *"simple and easy to read."* They suggested that the system could provide a starting template that outlines possible categories of requirements, which allows users to modify and personalize them rather than starting from scratch. Alice also mentioned that they simply put whatever came to mind into the requests without a clear structure. They argued that if the system could organize requirements into categories, such as layout, color, or functionality, the users would have a much clearer understanding of where improvements could be made and how their requests might translate into system changes.

User Desire and Suggestions

Participants expressed specific desires for how AI-driven personalization should operate and offered practical suggestions for system improvement.

CharlieP articulated this tension clearly, stating that having the interface *"dynamically change all the time"* would be disruptive and counterproductive. Instead, CharlieP preferred selective control, where they could keep an interface they felt comfortable with but request specific changes when needed.

For CharlieP, consistency was always a priority, while adaptability came second. Alice echoed this idea, describing how *"guided freedom"* could offer users a sense of autonomy without overwhelming them: for example, the system could present yes-or-no choices that simulate freedom while keeping complexity manageable. EveDP also envisioned a model in which the AI proactively offered possible modifications, leaving it up to the user to accept or decline.

Diana's experience during testing reinforced this point. Rather than manually editing proposals, which they described as *"exhausting,"* they often chose to simply generating an entirely new proposal to see whether it better fit their needs. Diana explained that typing the requests consumed significant effort to phrase correctly. They proposed that the conversational alternatives, such as directly telling the AI *"I don't want this change"* or *"please adjust it like this"*, would reduce the users' cognitive load.

4.4 Summary

This study combined quantitative and qualitative methods to evaluate the usability and user experience of an AI-powered UI personalization system. Quantitative data showed an overall acceptable usability (mean SUS = 69.17), with variations depending on user satisfaction and clarity of needs. While only one UI generation failed technically, around 57% of the generated UIs contained visual or structural issues, particularly when user input was underspecified. Composite interaction patterns (e.g., iterating between proposal and UI) generally led to more satisfactory outcomes. Eight primary themes were summarized from the thematic analysis, revealing a complex landscape of user expectations, system limitations, and design opportunities. While participants demonstrated strong enthusiasm for the concept of AI-adaptive interfaces, there were still some gaps between user needs and system capabilities creating substantial friction in the user experience.

Chapter 5

Discussion

This chapter interprets and synthesizes the findings regarding the research questions, and reflects their broader implications for LLM-driven personalization systems.

5.1 User Experience with LLM-Generated Personalized UIs (RQ1)

The overall user experience with the LLM-generated personalized UIs was mixed but promising. Participants appreciated the novelty of the system and were intrigued by its ability to generate UIs based on natural language prompts. The SUS score (M = 69.17) slightly exceeded the industry average benchmark, showing acceptable usability.

We also found out that usability perceptions were shaped differently according to different participants' backgrounds. Participants with programming experience generally evaluated the system more positively, possibly because they could better interpret the system's limitations and adapt their prompts accordingly. In contrast, participants without technical background reported lower SUS scores. It aligns with their struggles to articulate requirements and navigate the personalization process. These findings echo Pasch & Ha, who similarly found that different users bring different priorities to their evaluation of AI systems. They noticed that technical users are more likely to discuss how AI systems function internally and are therefore particularly noticeable to users who interact with or maintain system infrastructure, whereas non-technical users emphasize usability and customization [87]. Pasch & Cha also report that non-technical users

place greater weight on direct usability and human-centric aspects [88]. Additionally, Sun et al. show that experts / users with domain experience tend to generate higher quality, more novel outputs and rate generative AI tools more favorably on creativity tasks compared to novices [89].

Qualitative data revealed that many participants described the experience as fun, playful, and game-like. When the system was able to implement their ideas, they felt significantly satisfied. Participants also appreciated a sense of co-ownership and creativity, especially when their prompts yielded visually personalized results. These findings indicate that successful AI-driven UI personalization creates a unique form of user engagement that traditional static UIs are not able to offer. This resonates with Huang et al.'s vision of adaptive user experiences with generative AI [67], but our results show that the enjoyment relies on technical execution quality. Unlike Cao et al.'s theoretical framework for malleable UIs [69], our results show that users' emotional responses to AI collaboration also counts as a crucial factor to impact overall experience. This suggests that user experience for LLM-generated UIs must account for both functional and emotional perspectives.

However, participants also expressed disappointment and frustration when the generated UI did not align with their intentions. A recurring theme was the mismatch between the AI-generated proposals and the actual UI, particularly regarding incomplete or hallucinated features (see examples in 4.3). Some participants favored the baseline UI due to its reliability and predictability. This indicates that while the LLM-based system offered potential for greater personalization, its inconsistency affected perceived trust and satisfaction.

These findings echo with previous work on LLM hallucinations and their impact on user trust. Massenon et al. observed that hallucinations in AI mobile applications undermine user experience and diminish trust[90]. The importance of consistency for building user trust has also been emphasized in the Jakob Nielsen's ten heuristics [91], where predictable and stable design is seen as essential for decreasing cognitive load and encouraging continued use. Besides, Bach et al. emphasize that building trustworthy AI system requires not only technical improvements but also a balanced consideration of socio-ethical concerns and user experience angle to achieve trust between users and AI systems [92].

5.2 User Involvement in Enhancing UI Personalization (RQ2)

The correlation between prompt specificity and successful personalization strongly supports the value of user involvement. Participants who engaged actively by iteratively refining their prompts and editing proposals generally achieved higher satisfaction and more successful outcomes. These users were also more likely to be classified as "promoters" (NPS score 4–5) and scored above average on SUS.

However, not all users were willing to contribute. Those who relied heavily on system autonomy (e.g., simply clicking "generate") experienced more errors or irrelevant results. This demonstrates the importance of user involvement in leveraging the system's full potential.

These findings resonate with Gómez-Carmona et al, whose work shows that user involvement and customization capabilities significantly improve user satisfaction and engagement with AI-driven systems, which underscores the value of active user participation [93].

Participants valued the ability to inject personal input, especially when the system responded well to their prompts. The co-creative dynamic between user and AI created an emotional connection, increased perceived personalization, and often resulted in better-aligned designs. Even minimal contributions, such as specifying icons or layout tweaks, were sufficient to give users a sense of control and accomplishment. Similarly, Pasch & Ha found that customization in real-world AI products consistently enhance overall user satisfaction [87].

In addition, as shown in 4.1, we found that users with different background focused on different aspects of the system. Participants with programming experience concentrated on making a specific requirement work, often revising their prompts or proposals repeatedly. In contrast, those without technical backgrounds were more inclined to explore a variety of functions. This observation aligns with the findings of Pasch & Ha, who reported that technical professionals tend to prioritize system-level qualities such as reliability and explainability, whereas non-technical users place greater emphasis on usability and personalization [87]. Their study suggests the importance of role-sensitive AI design that can balance these differing priorities by providing both transparency for expert users and intuitive, user-friendly interfaces for general users.

Nevertheless, the effectiveness of user involvement is constrained by

domain knowledge limitations. The theme "Lack of Guidance & Prompting Support" reveals that participants struggled to articulate their needs into actionable AI inputs. They were often lack vocabulary to describe UI components or design concepts. This knowledge gap creates a barrier to effective involvement, which suggests that LLM personalization systems must offer more guidance and structured prompting frameworks. While Lu et al. proposed UI grammar-guided generation to structure LLM outputs [71], our results also indicate that structuring user inputs is equally important for successful outcomes.

The development of user coping strategies, such as making targeted changes, adding clarifying details, and reverting to simpler requests, demonstrates adaptive learning in human-AI collaboration. But this learning process requires system transparency and feedback mechanisms to be effective. Honeycutt et al. caution that human-in-the-loop feedback may paradoxically lower perceived system accuracy if not supported with feedback transparency and clear system response, hinting at the need for balanced design in user-driven AI systems [94].

5.3 Ease and Effectiveness of UI Personalization (RQ3)

Participants demonstrated mixed success in adapting UIs to their needs. Around 35% of iterations led to satisfying results, but nearly 30% were unsatisfactory. Simpler visual changes (e.g., font, color) were implemented effectively, while complex functional requests (e.g., scheduling, sport-specific features, dynamic components) were often unfulfilled. This suggests that current LLM capabilities are better at implementing aesthetic features than functional customization. Common issues including icon rendering failures, component parsing errors, and layout inconsistencies as shown in 4.3 suggest that current AI systems excel at generating code structures but struggle with integration details and resource management. These findings aligns with evidence from Lu et al., who report that current LLM agents handle visual constraints more reliably than functional ones when building multi-page websites [95].

Effectiveness was closely tied to prompt specificity. Clear, well-structured inputs (often resulting from more experienced or confident users) were significantly more likely to yield desirable results. On the opposite, vague or underspecified prompts frequently led to incomplete, incorrect, or hallucinated

features. Formal works also echo these findings that prompt effectiveness relies on clarity, context, structure, and domain specificity, with domain-specific vocabularies potentially having an optimal specificity threshold [96, 97].

Ease of personalization also depended on the system's feedback mechanisms. Many participants expressed a need for more guidance in formulating prompts, such as predefined templates or suggestion lists. The lack of UI-specific knowledge was a major barrier for some, leading to uncertainty and hesitation in expressing their needs. Techniques such as Structure–Aesthetics–Optimization (SAO) approach, which iteratively embeds aesthetic, functional, and contextual constraints, further underscore the importance of multi-dimensional prompt structuring for successful personalization [98]. Design guidance from Nielsen Norman Group also emphasizes use-case prompt suggestions to help users understand the tool's possibilities and engage more effectively [99].

5.4 Challenges in UI Generation and Personalization (RQ4)

There were several challenges found during the study.

Firstly, users often struggled with expressing their needs for the system to interpret. They reported confusion and requested more structured guidance such as example prompts, categorized input fields, chatbot-like system. This challenge aligns with Wei et al.'s observations about AI-inspire UI design [75], but our study offers spcific evidence of how communication failures manifest in personalization contexts.

The second challenge is the limitation of LLM. Despite the strong semantic interpretation of LLM, its generated results were often plagued by hallucinations, incomplete layout structures, invalid components, or made-up resources (See examples in 4.3). This not only led to user frustration but also took away user trust in the system's reliability. These technical limitations echo concerns raised in multiple studies [68, 71, 72] about the gap between LLM understanding and actual implementation capabilities.

Thirdly, the disconnect between expectations and implementations was a recurring reason to dissatisfaction. Participants noted that the AI-generated proposals often over-promised features that the final UI failed to deliver. This might be caused by the system structural limitations, which is another challenge. The JSON layout parser only accepted predefined resource types.

Consequently, many LLM-generated UIs containing uncommon attributes, unconcluded external resources, or unrecognized components could not be successfully parsed or displayed. Even when users provided clear prompts, the system sometimes failed to support the intended design outcomes, which also affected user satisfaction.

In addition, several participants found the task of typing or editing prompts mentally exhausting. Many preferred a system that could infer their needs or offer multiple options instead of requiring explicit input. These findings align with prior work on the cognitive burden of prompting in generative AI. For example, Tankelevitch et al. discussed how generative systems place metacognitive demands on users, particularly when users must refine, evaluate, and correct prompts [100]. A large-scale survey by Liang et al. similarly found that developers using AI programming assistants prefer tools that can infer context or require fewer explicit edits, as constant prompt tweaking incurs fatigue [101]. Research from Nielsen Norman group on early generative AI tools also shows that users favor templates, suggestions, or multiple options instead of having to craft prompt text from scratch, to reduce the cognitive effort involved [102].

Last but not least, users wanted control over what changed and what remained consistent. They favored systems that could adapt intelligently to their needs while still keeping core elements of their design. Too much automation or dynamism was viewed negatively. Similarly, the Adaptive Interface Framework emphasizes balancing automation with user control to preserve predictability and protect users' mental models [103].

5.5 Synthesis with Existing Literature

This project contributes to the growing body of literature on LLM-driven UI generation and personalization by offering empirical evidence and revealing new challenges not previously documented. Existing approaches, such as CrowdGenUI [66] and Huang et al. [67], have showed that the importance of incorporating user preferences to improve the alignment between user expectations and generated outcomes. These works often assume structured user preference data, while this project explores real-time personalization without any prior user data, making it more suitable for immediate personalization scenarios.

Moreover, several previous works, such as Low-code LLM [68], Cao et al. [69], have proposed interactive frameworks that allow users to visually modify generated UIs, which reduce the dependency on natural language

precision. While this study examines a pure natural language-based interaction without graphical support, we found that although natural language offers high freedom, non-expert users often struggle to specify needs precisely. This reveals the value of structured prompting scaffolds.

The results of this study also confirm and extend insights from prior work on prompt engineering and constraint-based approaches [69, 71], which address the value of UI grammar and intermediate representations to improve correctness and reduce hallucination in LLM outputs.

Previous research in UI generation has primarily emphasized direct code generation from natural language or multi-modal inputs. For example, Si et al. and Wu et al. demonstrate that LLMs can translate textual descriptions or design screenshots into executable UI code such as SwiftUI or HTML/CSS [40, 42]. While these approaches showcase the technical feasibility of end-to-end design-to-code pipelines, they also introduce challenges of safety, reliability, and maintainability, as generated code may contain errors or insecure patterns.

In contrast, more recent work has explored JSON-based or structured specification approaches that serve as an intermediate layer between user intent and final UI rendering. Systems like SpecifyUI and Json-GUI highlight how structured specifications can make generation more controllable, interpretable, and resilient by abstracting away low-level implementation details [46, 47]. This study builds on this latter line of work, showing how natural language prompts can be mapped into JSON specifications that are then dynamically rendered into UIs. This strategy prioritizes safety and transparency, but also revealing new user experience challenges such as mismatches between proposals and rendered outcomes.

While existing work has mostly focused on technical capabilities of LLM [68, 71, 72], this study reveals that user experience quality is equally crucial in AI-UI personalization. The difference between our user-centered findings and the more technically-focuses approaches suggests that future work should adopt more holistic evaluation frameworks that consider both technical performance and user experience aspects [74, 75]. Moreover, future research should research longitudinal user experiences with AI-driven personalization to understand how user expertise and system capabilities evolve over time. Additionally, comparative studies which examine different models and interaction paradigms could offer insights into optimal design patterns for human-AI collaborative UIs. It might potentially bridge the gap between professional design tools explored by Feng et al. [73] and daily user personalization needs.

Chapter 6

Conclusions and Future work

This study explored how users interact with a LLM-generated personalization system for generating personalized UIs according to users' requirements. By conducting usability tests and analyzing both quantitative and qualitative data, this study provides a comprehensive vision of the usability, limitations, and experiential dimensions of LLM-generated UIs.

This study extends prior work on LLM-driven UI personalization by providing empirical evidence of real-time personalization without prior user data, complementing approaches that rely on structured preference data [66, 67]. It also examines a purely natural language-based interaction, revealing both its freedoms and limitations instead of previous low-code frameworks [68, 69]. Moreover, it highlights that beyond technical correctness [68, 71, 72], user experience quality is a critical dimension, underscoring the need for more holistic evaluation frameworks in future AI-UI personalization research.

The study showed promising potential for LLM-generated UI personalization. Most participants appreciated the novelty of the system and enjoyed the sense of creative collaboration with AI. The system successfully enabled some users who iteratively clarified their needs to generate satisfying and personalized UI designs. The thematic analysis also reveled that when implemented successfully, the UI personalization increased user engagement, emotional connection the the interface, and motivation to use the application more often.

However, the system also showed significant drawbacks. Some primary drawbacks included the system structural limitations that could not render or parse many of the LLM's creative outputs, leading to broken interfaces or missing features. Users also struggled to articulate their requests clearly, and vague prompts frequently led to hallucinated or unimplementable results.

Furthermore, there was often a mismatch between the proposals generated by the LLM and the actual implemented UI, leading to frustration and unmet expectations.

There are also several limitations of this study must be acknowledged. The model occasionally hallucinated or returned overly ambitious results that the system could not implement. Furthermore, LLMs demonstrated bias toward certain design styles and struggled to generalize across diverse user preferences without explicit input. The sample size of this study was relatively small and within a narrow age range, which might limit generalizability to broader populations and lack diversity. This project also operated in a fixed UI framework that restricted the ability interpret and render more creative or flexible outputs from the LLM.

One of the most important insights is that AI-powered systems need to find a balance between automation and control. Users appreciated the autonomy of the AI but desired more guidance and conversational flexibility. Another insight is that users might struggle to specify their needs since they might not even know themselves, suggesting that structured support are crucial. For researchers and designers working in this field, we suggest that designing systems that tolerate vague input and gently guide users toward specificity.

If this project were to be done again, I would like to spend more time for iterative testing and debugging of the parsing engine to minimize deployment errors. I would also expand the participant pool to include users with a wider range of design expertise and demographic backgrounds. Moreover, I would consider integrating conversational agents or feedback loops to allow users to negotiate UI changes in more natural language. Finally, I might consider to integrate AI agents or a RAG pipeline into the system to decrease the LLM hallucination and enhance the retrieval of external resources.

In conclusion, this thesis contributes a foundational understanding of user experiences in LLM-driven UI personalization and offers actionable insights for improving the alignment between user intent and system output. It opens the door for future research and development in more intuitive, resilient, and co-creative AI design tools.

References

[1] J. Bieniek, M. Rahouti, and D. C. Verma, "Generative AI in Multimodal User Interfaces: Trends, Challenges, and Cross-Platform Adaptability," Nov. 2024, arXiv:2411.10234. [Online]. Available: http://arxiv.org/abs/2411.10234 [Page 1.]

[2] K. Moran and S. Gibbons, "Generative UI and Outcome-Oriented Design," 2024. [Online]. Available: https://www.nngroup.com/articles/generative-ui/ [Page 1.]

[3] F. Ricci, L. Rokach, and B. Shapira, "Recommender Systems: Techniques, Applications, and Challenges," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. New York, NY: Springer US, 2022, pp. 1–35. ISBN 9781071621974. [Online]. Available: https://doi.org/10.1007/978-1-0716-2197-4_1 [Page 2.]

[4] J. Hussain, A. Ul Hassan, H. S. Muhammad Bilal, R. Ali, M. Afzal, S. Hussain, J. Bang, O. Banos, and S. Lee, "Model-based adaptive user interface based on context and user experience evaluation," *Journal on Multimodal User Interfaces*, vol. 12, no. 1, pp. 1–16, Mar. 2018. doi: 10.1007/s12193-018-0258-2. [Online]. Available: http://link.springer.com/10.1007/s12193-018-0258-2 [Pages 2 and 14.]

[5] N. Carter, D. Bryant-Lukosius, A. DiCenso, J. Blythe, and A. J. Neville, "The Use of Triangulation in Qualitative Research," *Oncology Nursing Forum*, vol. 41, no. 5, pp. 545–547, Sep. 2014. doi: 10.1188/14.ONF.545-547. [Online]. Available: http://onf.ons.org/onf/41/5/use-triangulation-qualitative-research [Page 5.]

[6] Xiaoan Zhan, Yang Xu, and Yingchia Liu, "Personalized UI Layout Generation using Deep Learning: An Adaptive Interface Design Approach for Enhanced User Experience," Oct. 2024. doi:

10.5281/ZENODO.14190245. [Online]. Available: https://zenodo.org/doi/10.5281/zenodo.14190245 [Pages 6 and 14.]

[7] X. Wang, C. Na, E. Strubell, S. Friedler, and S. Luccioni, "Energy and Carbon Considerations of Fine-Tuning BERT," 2023. doi: 10.48550/ARXIV.2311.10267. [Online]. Available: https://arxiv.org/abs/2311.10267 [Page 6.]

[8] N. Jegham, M. Abdelatti, L. Elmoubarki, and A. Hendawi, "How Hungry is AI? Benchmarking Energy, Water, and Carbon Footprint of LLM Inference," 2025. [Online]. Available: https://arxiv.org/abs/2505.09598 [Page 6.]

[9] P. Wacnik, S. Daly, and A. Verma, "Participatory design: A systematic review and insights for future practice," 2024. [Online]. Available: https://arxiv.org/abs/2409.17952 [Page 6.]

[10] J. Zhao, Y. Ding, C. Jia, Y. Wang, and Z. Qian, "Gender Bias in Large Language Models across Multiple Languages," 2024. [Online]. Available: https://arxiv.org/abs/2403.00277 [Page 6.]

[11] T. Hu, Y. Kyrychenko, S. Rathje, N. Collier, S. Van Der Linden, and J. Roozenbeek, "Generative language models exhibit social identity biases," *Nature Computational Science*, vol. 5, no. 1, pp. 65–75, Dec. 2024. doi: 10.1038/s43588-024-00741-1. [Online]. Available: https://www.nature.com/articles/s43588-024-00741-1 [Page 6.]

[12] A.-E. Guriță and R.-D. Vatavu, "Good Accessibility, Handcuffed Creativity: AI-Generated UIs Between Accessibility Guidelines and Practitioners' Expectations," in *Proceedings of the 2025 ACM Designing Interactive Systems Conference*. Madeira Portugal: ACM, Jul. 2025. doi: 10.1145/3715336.3735691. ISBN 9798400714856 pp. 1197–1209. [Online]. Available: https://dl.acm.org/doi/10.1145/3715336.3735691 [Page 6.]

[13] K. Howell, G. Christian, P. Fomitchov, G. Kehat, J. Marzulla, L. Rolston, J. Tredup, I. Zimmerman, E. Selfridge, and J. Bradley, "The economic trade-offs of large language models: A case study," 2023. [Online]. Available: https://arxiv.org/abs/2306.07402 [Page 6.]

[14] C. Irugalbandara, A. Mahendra, R. Daynauth, T. K. Arachchige, J. Dantanarayana, K. Flautner, L. Tang, Y. Kang, and J. Mars,

"Scaling Down to Scale Up: A Cost-Benefit Analysis of Replacing OpenAI's LLM with Open Source SLMs in Production," 2023. doi: 10.48550/ARXIV.2312.14972. [Online]. Available: https://arxiv.org/abs/2312.14972 [Page 6.]

[15] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, and A. Mian, "A Comprehensive Overview of Large Language Models," 2023. [Online]. Available: https://arxiv.org/abs/2307.06435 [Page 9.]

[16] J. Jiang, F. Wang, J. Shen, S. Kim, and S. Kim, "A Survey on Large Language Models for Code Generation," 2024. [Online]. Available: https://arxiv.org/abs/2406.00515 [Pages 9 and 12.]

[17] A. Kumar and P. Sharma, "Open AI Codex: An Inevitable Future?" *International Journal for Research in Applied Science and Engineering Technology*, vol. 11, no. 2, pp. 539–543, Feb. 2023. doi: 10.22214/ijraset.2023.49048. [Online]. Available: https://www.ijraset.com/best-journal/open-ai-codex-an-inevitable-future [Page 9.]

[18] "GitHub Copilot · Your AI pair programmer." [Online]. Available: https://github.com/features/copilot [Page 9.]

[19] O. Asare, M. Nagappan, and N. Asokan, "Is GitHub's Copilot as Bad as Humans at Introducing Vulnerabilities in Code?" 2022. [Online]. Available: https://arxiv.org/abs/2204.04741 [Page 9.]

[20] P. Vaithilingam, T. Zhang, and E. L. Glassman, "Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models," in *CHI Conference on Human Factors in Computing Systems Extended Abstracts*. New Orleans LA USA: ACM, Apr. 2022. doi: 10.1145/3491101.3519665. ISBN 9781450391566 pp. 1–7. [Online]. Available: https://dl.acm.org/doi/10.1145/3491101.3519665 [Page 9.]

[21] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing," 2021. [Online]. Available: https://arxiv.org/abs/2107.13586 [Page 9.]

[22] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT," 2023. [Online]. Available: https://arxiv.org/abs/2302.11382 [Pages 9 and 10.]

[23] S. Ekin, "Prompt Engineering For ChatGPT: A Quick Guide To Techniques, Tips, And Best Practices," Apr. 2023. [Online]. Available: https://www.techrxiv.org/doi/full/10.36227/techrxiv.22683919.v1 [Pages 9 and 10.]

[24] B. Xu, A. Yang, J. Lin, Q. Wang, C. Zhou, Y. Zhang, and Z. Mao, "ExpertPrompting: Instructing Large Language Models to be Distinguished Experts," 2023. [Online]. Available: https://arxiv.org/abs/2305.14688 [Page 10.]

[25] B. Wang, S. Min, X. Deng, J. Shen, Y. Wu, L. Zettlemoyer, and H. Sun, "Towards Understanding Chain-of-Thought Prompting: An Empirical Study of What Matters," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Toronto, Canada: Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.153 pp. 2717–2739. [Online]. Available: https://aclanthology.org/2023.acl-long.153 [Page 10.]

[26] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," 2022. [Online]. Available: https://arxiv.org/abs/2201.11903 [Page 10.]

[27] M. Renze and E. Guven, "The Effect of Sampling Temperature on Problem Solving in Large Language Models," 2024. doi: 10.48550/ARXIV.2402.05201. [Online]. Available: https://arxiv.org/abs/2402.05201 [Page 10.]

[28] J. Holdsworth and M. Kosinski, "What is tokenization?" [Page 10.]

[29] J. W. Zimmerman, D. Hudon, K. Cramer, A. J. Ruiz, C. Beauregard, A. Fehr, M. I. Fudolig, B. Demarest, Y. M. Bird, M. Z. Trujillo, C. M. Danforth, and P. S. Dodds, "Tokens, the oft-overlooked appetizer: Large language models, the distributional hypothesis, and meaning," 2024. [Online]. Available: https://arxiv.org/abs/2412.10924 [Page 10.]

[30] Supal Kanti Chowdhury, "Token optimization: The backbone of effective prompt engineering," Sep. 2024. [Online]. Available: https://developer.ibm.com/articles/awb-token-optimization-backbone-of-effective-prompt-engineering/ [Page 10.]

[31] R. Duan, N. Karthik, J. Shi, R. Jain, M. C. Yang, and K. Ramani, "ConceptVis: Generating and Exploring Design Concepts for Early-Stage Ideation Using Large Language Model," in *Volume 3B: 50th Design Automation Conference (DAC)*. Washington, DC, USA: American Society of Mechanical Engineers, Aug. 2024. doi: 10.1115/DETC2024-146409. ISBN 9780791888377 p. V03BT03A042. [Online]. Available: https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings/IDETC-CIE2024/88377/V03BT03A042/1208891 [Pages 10 and 11.]

[32] A. Schuller, D. Janssen, J. Blumenröther, T. M. Probst, M. Schmidt, and C. Kumar, "Generating personas using LLMs and assessing their viability," in *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*. Honolulu HI USA: ACM, May 2024. doi: 10.1145/3613905.3650860. ISBN 9798400703317 pp. 1–7. [Online]. Available: https://dl.acm.org/doi/10.1145/3613905.3650860 [Page 10.]

[33] S. Petridis, M. Terry, and C. J. Cai, "PromptInfuser: How Tightly Coupling AI and UI Design Impacts Designers' Workflows," in *Designing Interactive Systems Conference*. IT University of Copenhagen Denmark: ACM, Jul. 2024. doi: 10.1145/3643834.3661613. ISBN 9798400705830 pp. 743–756. [Online]. Available: https://dl.acm.org/doi/10.1145/3643834.3661613 [Page 10.]

[34] J. Marklund, "The Evolution of UX Design: Exploring the Collaboration of UX Practitioners and AI Tools in the Design Process." [Online]. Available: https://api.semanticscholar.org/CorpusID:272603334 [Page 10.]

[35] S. G. Bouschery, V. Blazevic, and F. T. Piller, "Augmenting human innovation teams with artificial intelligence: Exploring transformer-based language models," *Journal of Product Innovation Management*, vol. 40, no. 2, pp. 139–153, Mar. 2023. doi:

10.1111/jpim.12656. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1111/jpim.12656 [Page 10.]

[36] S. Suh, M. Chen, B. Min, T. J.-J. Li, and H. Xia, "Luminate: Structured Generation and Exploration of Design Space with Large Language Models for Human-AI Co-Creation," in *Proceedings of the CHI Conference on Human Factors in Computing Systems*. Honolulu HI USA: ACM, May 2024. doi: 10.1145/3613904.3642400. ISBN 9798400703300 pp. 1–26. [Online]. Available: https://dl.acm.org/doi/10.1145/3613904.3642400 [Page 11.]

[37] X. Zhang, L. Liu, Y. Wang, X. Liu, H. Wang, C. Arora, H. Liu, W. Wang, and T. Hoang, "Auto-Generated Personas: Enhancing User-centered Design Practices among University Students," in *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*. Honolulu HI USA: ACM, May 2024. doi: 10.1145/3613905.3651043. ISBN 9798400703317 pp. 1–7. [Online]. Available: https://dl.acm.org/doi/10.1145/3613905.3651043 [Page 11.]

[38] V. Bilgram and F. Laarmann, "Accelerating Innovation With Generative AI: AI-Augmented Digital Prototyping and Innovation Methods," *IEEE Engineering Management Review*, vol. 51, no. 2, pp. 18–25, Jun. 2023. doi: 10.1109/EMR.2023.3272799. [Online]. Available: https://ieeexplore.ieee.org/document/10115412/ [Page 11.]

[39] J. Ma, K. Sreedhar, V. Liu, S. Wang, P. A. Perez, and L. B. Chilton, "DIDUP: Dynamic Iterative Development for UI Prototyping," 2024. [Online]. Available: https://arxiv.org/abs/2407.08474 [Page 11.]

[40] C. Si, Y. Zhang, R. Li, Z. Yang, R. Liu, and D. Yang, "Design2Code: Benchmarking Multimodal Code Generation for Automated Front-End Engineering," 2024. [Online]. Available: https://arxiv.org/abs/2403.03163 [Pages 11, 12, and 57.]

[41] Y. Wan, C. Wang, Y. Dong, W. Wang, S. Li, Y. Huo, and M. R. Lyu, "Automatically Generating UI Code from Screenshot: A Divide-and-Conquer-Based Approach," 2024. doi: 10.48550/ARXIV.2406.16386. [Online]. Available: https://arxiv.org/abs/2406.16386 [Pages 11 and 12.]

[42] J. Wu, E. Schoop, A. Leung, T. Barik, J. P. Bigham, and J. Nichols, "UICoder: Finetuning Large Language Models to Generate User Interface Code through Automated Feedback," 2024. [Online]. Available: https://arxiv.org/abs/2406.07739 [Pages 11, 12, and 57.]

[43] Y. Gui, Z. Li, Z. Zhang, G. Wang, T. Lv, G. Jiang, Y. Liu, D. Chen, Y. Wan, H. Zhang, W. Jiang, X. Shi, and H. Jin, "LaTCoder: Converting Webpage Design to Code with Layout-as-Thought," 2025. [Online]. Available: https://arxiv.org/abs/2508.03560 [Pages 11 and 12.]

[44] F. Zhang, L. Wu, H. Bai, G. Lin, X. Li, X. Yu, Y. Wang, B. Chen, and J. Keung, "HumanEval-V: Benchmarking High-Level Visual Reasoning with Complex Diagrams in Coding Tasks," 2024. [Online]. Available: https://arxiv.org/abs/2410.12381 [Page 12.]

[45] Z. Tang, C. Wu, J. Li, and N. Duan, "LayoutNUWA: Revealing the Hidden Layout Expertise of Large Language Models," 2023. [Online]. Available: https://arxiv.org/abs/2309.09506 [Page 12.]

[46] Y. Chen, C. Shi, and L. Chen, "SpecifyUI: Supporting Iterative UI Design Intent Expression through Structured Specifications and Generative AI," 2025. [Online]. Available: https://arxiv.org/abs/2509.07334 [Pages 13 and 57.]

[47] A. Galizia, G. Zereik, L. Roverelli, E. Danovaro, A. Clematis, and D. D'Agostino, "Json-GUI—A module for the dynamic generation of form-based web interfaces," *SoftwareX*, vol. 9, pp. 28–34, Jan. 2019. doi: 10.1016/j.softx.2018.11.007. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2352711018300505 [Pages 13 and 57.]

[48] E. Chavarriaga, F. Jurado, and F. D. Rodríguez, "An approach to build JSON-based Domain Specific Languages solutions for web applications," *Journal of Computer Languages*, vol. 75, p. 101203, Jun. 2023. doi: 10.1016/j.cola.2023.101203. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2590118423000138 [Page 13.]

[49] C. M. Gallardo Paredes, C. Machuca, and Y. M. Semblantes Claudio, "ChatGPT API: Brief overview and integration in Software Development," *International Journal of Engineering Insights*, vol. 1,

no. 1, pp. 25–29, Nov. 2023. doi: 10.61961/injei.v1i1.7. [Online]. Available: https://injei.sci-thoth.com/index.php/journal/article/view/7 [Page 13.]

[50] "Introducing GPT-4.1 in the API," Apr. 2025. [Online]. Available: https://openai.com/index/gpt-4-1/ [Page 13.]

[51] W. Stuerzlinger, O. Chapuis, D. Phillips, and N. Roussel, "User interface façades: towards fully adaptable user interfaces," in *Proceedings of the 19th annual ACM symposium on User interface software and technology*. Montreux Switzerland: ACM, Oct. 2006. doi: 10.1145/1166253.1166301. ISBN 9781595933133 pp. 309–318. [Online]. Available: https://dl.acm.org/doi/10.1145/1166253.1166301 [Pages 14 and 15.]

[52] A. Leonidis, M. Antona, and C. Stephanidis, "Rapid Prototyping of Adaptable User Interfaces," *International Journal of Human-Computer Interaction*, vol. 28, no. 4, pp. 213–235, Apr. 2012. doi: 10.1080/10447318.2011.581891. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/10447318.2011.581891 [Page 14.]

[53] A. Bunt, "Mixed-initiative support for customizing graphical user interfaces," Ph.D. dissertation, University of British Columbia, 2007. [Online]. Available: https://open.library.ubc.ca/soa/cIRcle/collections/ubctheses/831/items/1.0051695 [Page 14.]

[54] A. Bunt, C. Conati, and J. McGrenere, "What role can adaptive support play in an adaptable system?" in *Proceedings of the 9th international conference on Intelligent user interfaces*. Funchal, Madeira Portugal: ACM, Jan. 2004. doi: 10.1145/964442.964465. ISBN 9781581138153 pp. 117–124. [Online]. Available: https://dl.acm.org/doi/10.1145/964442.964465 [Page 14.]

[55] S. Abrahão, E. Insfran, A. Sluÿters, and J. Vanderdonckt, "Model-based intelligent user interface adaptation: challenges and future directions," *Software and Systems Modeling*, vol. 20, no. 5, pp. 1335–1349, Oct. 2021. doi: 10.1007/s10270-021-00909-7. [Online]. Available: https://link.springer.com/10.1007/s10270-021-00909-7 [Pages 14 and 15.]

[56] J. Zhu, J. Hong, and J. G. Hughes, "Using Markov Chains for Link Prediction in Adaptive Web Sites," in *Soft-Ware 2002: Computing in an Imperfect World*, D. Bustard, W. Liu, and R. Sterritt, Eds. Berlin, Heidelberg: Springer, 2002. doi: 10.1007/3-540-46019-5_5.$ISBN 9783540460190 pp. 60 - -73$. [$Page$ 14.]

[57] A. Carrera-Rivera, D. Reguera-Bakhache, F. Larrinaga, and G. Lasa, "Exploring the transformation of user interactions to Adaptive Human-Machine Interfaces," in *XXIII International Conference on Human Computer Interaction*. Lleida Spain: ACM, Sep. 2023. doi: 10.1145/3612783.3612807. ISBN 9798400707902 pp. 1–7. [Online]. Available: https://dl.acm.org/doi/10.1145/3612783.3612807 [Page 14.]

[58] K. Todi, G. Bailly, L. Leiva, and A. Oulasvirta, "Adapting User Interfaces with Model-based Reinforcement Learning," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. Yokohama Japan: ACM, May 2021. doi: 10.1145/3411764.3445497. ISBN 9781450380966 pp. 1–13. [Online]. Available: https://dl.acm.org/doi/10.1145/3411764.3445497 [Pages 14 and 15.]

[59] J. Cho, J. Kim, D. Bae, J. Choo, Y. Gwon, and Y.-D. Kwon, "CAAP: Context-Aware Action Planning Prompting to Solve Computer Tasks with Front-End UI Only," 2024. [Online]. Available: https://arxiv.org/abs/2406.06947 [Page 14.]

[60] P. Ackermann, C. A. Velasco, and C. Power, "Developing a semantic user and device modeling framework that supports UI adaptability of web 2.0 applications for people with special needs," in *Proceedings of the International Cross-Disciplinary Conference on Web Accessibility*. Lyon France: ACM, Apr. 2012. doi: 10.1145/2207016.2207018. ISBN 9781450310192 pp. 1–4. [Online]. Available: https://dl.acm.org/doi/10.1145/2207016.2207018 [Page 14.]

[61] H. Soh, S. Sanner, M. White, and G. Jamieson, "Deep Sequential Recommendation for Personalized Adaptive User Interfaces," in *Proceedings of the 22nd International Conference on Intelligent User Interfaces*. Limassol Cyprus: ACM, Mar. 2017. doi: 10.1145/3025171.3025207. ISBN 9781450343480 pp. 589–593. [Online]. Available: https://dl.acm.org/doi/10.1145/3025171.3025207 [Page 14.]

[62] R. Zhang, S. Wang, T. Xie, S. Duan, and M. Chen, "Dynamic User Interface Generation for Enhanced Human-Computer Interaction Using Variational

Autoencoders," in *2024 4th International Conference on Communication Technology and Information Technology (ICCTIT)*. Guangzhou, China: IEEE, Dec. 2024. doi: 10.1109/ICCTIT64404.2024.10928384. ISBN 9798331528973 pp. 260–265. [Online]. Available: https://ieeexplore.ieee.org/document/10928384/ [Page 14.]

[63] T. Langerak, S. Christen, M. Albaba, C. Gebhardt, C. Holz, and O. Hilliges, "MARLUI: Multi-Agent Reinforcement Learning for Adaptive Point-and-Click UIs," *Proceedings of the ACM on Human-Computer Interaction*, vol. 8, no. EICS, pp. 1–27, Jun. 2024. doi: 10.1145/3661147. [Online]. Available: https://dl.acm.org/doi/10.1145/3661147 [Page 15.]

[64] O. J. Romero, A. Haig, L. Kirabo, Q. Yang, J. Zimmerman, A. Tomasic, and A. Steinfeld, "A Long-Term Evaluation of Adaptive Interface Design for Mobile Transit Information," in *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services*. Oldenburg Germany: ACM, Oct. 2020. doi: 10.1145/3379503.3403536. ISBN 9781450375160 pp. 1–11. [Online]. Available: https://dl.acm.org/doi/10.1145/3379503.3403536 [Page 15.]

[65] Z. Cen and Y. Zhao, "Enhancing User Engagement through Adaptive Interfaces: A Study on Real-time Personalization in Web Applications," *Journal of Economic Theory and Business Management*, vol. 1, no. 6, pp. 1–7, Dec. 2024. doi: 10.70393/6a6574626d.323332. [Online]. Available: https://www.suaspress.org/ojs/index.php/JETBM/article/view/v1n6a01 [Page 15.]

[66] Y. Liu, M. Sra, and C. Xiao, "CrowdGenUI: Aligning LLM-Based UI Generation with Crowdsourced User Preferences," 2024. [Online]. Available: https://arxiv.org/abs/2411.03477 [Pages 16, 56, and 59.]

[67] Y. Huang, T. Kanij, A. Madugalla, S. Mahajan, C. Arora, and J. Grundy, "Unlocking Adaptive User Experience with Generative AI," 2024. [Online]. Available: https://arxiv.org/abs/2404.05442 [Pages 16, 18, 52, 56, and 59.]

[68] Y. Cai, S. Mao, W. Wu, Z. Wang, Y. Liang, T. Ge, C. Wu, W. You, T. Song, Y. Xia, J. Tien, N. Duan, and F. Wei, "Low-code LLM: Graphical User Interface over Large Language Models," 2023. [Online]. Available: https://arxiv.org/abs/2304.08103 [Pages 16, 55, 56, 57, and 59.]

[69] Y. Cao, P. Jiang, and H. Xia, "Generative and Malleable User Interfaces with Generative and Evolving Task-Driven Data Model," 2025. [Online].

Available: https://arxiv.org/abs/2503.04084 [Pages 17, 18, 52, 56, 57, and 59.]

[70] D. Gaspar-Figueiredo, M. Fernández-Diego, S. Abrahão, and E. Insfran, "Integrating Human Feedback into a Reinforcement Learning-Based Framework for Adaptive User Interfaces," 2025. [Online]. Available: https://arxiv.org/abs/2504.20782 [Page 17.]

[71] Y. Lu, Z. Tong, Q. Zhao, C. Zhang, and T. J.-J. Li, "UI Layout Generation with LLMs Guided by UI Grammar," Oct. 2023, arXiv:2310.15455. [Online]. Available: http://arxiv.org/abs/2310.15455 [Pages 17, 54, 55, 57, and 59.]

[72] J.-M. Rydholm, "From UI images to application logic using vision-capable LLMs: Building an automated pipeline for generating code from UI image inputs," Ph.D. dissertation, KTH, 2024. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-360839 [Pages 18, 55, 57, and 59.]

[73] S. Feng, M. Yuan, J. Chen, Z. Xing, and C. Chen, "Designing with Language: Wireframing UI Design Intent with Generative Large Language Models," 2023. [Online]. Available: https://arxiv.org/abs/2312.07755 [Pages 18 and 57.]

[74] H. Wen, W. Du, Y. Li, and Y. Liu, "Poster: Enabling Agent-centric Interaction on Smartphones with LLM-based UI Reassembling," in *Proceedings of the 22nd Annual International Conference on Mobile Systems, Applications and Services*. Minato-ku, Tokyo Japan: ACM, Jun. 2024. doi: 10.1145/3643832.3661432. ISBN 9798400705816 pp. 706–707. [Online]. Available: https://dl.acm.org/doi/10.1145/3643832.3661432 [Pages 18 and 57.]

[75] J. Wei, A.-L. Courbis, T. Lambolais, G. Dray, and W. Maalej, "On AI-Inspired UI-Design," 2024. [Online]. Available: https://arxiv.org/abs/2406.13631 [Pages 18, 55, and 57.]

[76] J. Nielsen, "Enhancing the explanatory power of usability heuristics," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Boston Massachusetts USA: ACM, Apr. 1994. doi: 10.1145/191666.191729. ISBN 9780897916509 pp. 152–158. [Online]. Available: https://dl.acm.org/doi/10.1145/191666.191729 [Page 28.]

[77] J. Golzar, S. Noor, and O. Tajik, "Convenience Sampling," *International Journal of Education & Language Studies*, vol. 1, no. 2, pp. 72–77, Dec. 2022. doi: 10.22034/ijels.2022.162981. [Online]. Available: https://www.ijels.net/article_162981.html [Page 29.]

[78] A. Joshi, S. Kale, S. Chandel, and D. Pal, "Likert Scale: Explored and Explained," *British Journal of Applied Science & Technology*, vol. 7, no. 4, pp. 396–403, Jan. 2015. doi: 10.9734/BJAST/2015/14975. [Online]. Available: https://journalcjast.com/index.php/CJAST/article/view/381 [Page 30.]

[79] J. Brooke, "SUS: A quick and dirty usability scale," 1995. [Page 30.]

[80] P. Laubheimer, "Beyond the NPS: Measuring Perceived Usability with the SUS, NASA-TLX, and the Single Ease Question After Tasks and Usability Tests," 2018. [Online]. Available: https://www.nngroup.com/articles/measuring-perceived-usability/ [Page 30.]

[81] J. R. Lewis and J. Sauro, "Item benchmarks for the system usability scale," *J. Usability Studies*, vol. 13, no. 3, pp. 158–167, 2018. doi: 10.5555/3294033.3294037 [Page 31.]

[82] F. F. Reichheld, "The one number you need to grow," *Harvard business review*, vol. 81, no. 12, pp. 46–124. [Page 31.]

[83] "ATLAS.ti," 2025. [Online]. Available: https://atlasti.com/ [Page 32.]

[84] M. E. Kiger and L. Varpio, "Thematic analysis of qualitative data: AMEE Guide No. 131," *Medical Teacher*, vol. 42, no. 8, pp. 846–854, Aug. 2020. doi: 10.1080/0142159X.2020.1755030. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/0142159X.2020.1755030 [Page 32.]

[85] B. K. Sovacool, M. Iskandarova, and J. Hall, "Industrializing theories: A thematic analysis of conceptual frameworks and typologies for industrial sociotechnical change in a low-carbon future," *Energy Research & Social Science*, vol. 97, p. 102954, Mar. 2023. doi: 10.1016/j.erss.2023.102954. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2214629623000142 [Page 32.]

[86] M. Rosala, "How to Analyze Qualitative Data from UX Research: Thematic Analysis," 2022. [Pages ix and 32.]

[87] S. Pasch and S.-Y. Ha, "Human-AI Interaction and User Satisfaction: Empirical Evidence from Online Reviews of AI Products," 2025. [Online]. Available: https://arxiv.org/abs/2503.17955 [Pages 51 and 53.]

[88] S. Pasch and M. C. Cha, "Do Ethical AI Principles Matter to Users? A Large-Scale Analysis of User Sentiment and Satisfaction," 2025. [Online]. Available: https://arxiv.org/abs/2508.05913 [Page 52.]

[89] L. Sun, Y. Liu, G. Joseph, Z. Yu, H. Zhu, and S. P. Dow, "Comparing Experts and Novices for AI Data Work: Insights on Allocating Human Intelligence to Design a Conversational Agent," *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, vol. 10, pp. 195–206, Oct. 2022. doi: 10.1609/hcomp.v10i1.21999. [Online]. Available: https://ojs.aaai.org/index.php/HCOMP/article/view/21999 [Page 52.]

[90] R. Massenon, I. Gambo, J. A. Khan, C. Agbonkhese, and A. Alwadain, ""My AI is Lying to Me": User-reported LLM hallucinations in AI mobile apps reviews," *Scientific Reports*, vol. 15, no. 1, p. 30397, Aug. 2025. doi: 10.1038/s41598-025-15416-8. [Online]. Available: https://www.nature.com/articles/s41598-025-15416-8 [Page 52.]

[91] J. Nielsen, "10 Usability Heuristics for User Interface Design," 2005. [Online]. Available: https://www.nngroup.com/articles/ten-usability-heuristics/ [Page 52.]

[92] T. A. Bach, A. Khan, H. Hallock, G. Beltrão, and S. Sousa, "A Systematic Literature Review of User Trust in AI-Enabled Systems: An HCI Perspective," 2023. doi: 10.48550/ARXIV.2304.08795. [Online]. Available: https://arxiv.org/abs/2304.08795 [Page 52.]

[93] O. Gómez-Carmona, D. Casado-Mansilla, D. López-de Ipiña, and J. García-Zubia, "Human-in-the-loop machine learning: Reconceptualizing the role of the user in interactive approaches," *Internet of Things*, vol. 25, p. 101048, Apr. 2024. doi: 10.1016/j.iot.2023.101048. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2542660523003712 [Page 53.]

[94] D. R. Honeycutt, M. Nourani, and E. D. Ragan, "Soliciting Human-in-the-Loop User Feedback for Interactive Machine Learning Reduces User Trust and Impressions of Model Accuracy," 2020. [Online]. Available: https://arxiv.org/abs/2008.12735 [Page 54.]

[95] Z. Lu, Y. Yang, H. Ren, H. Hou, H. Xiao, K. Wang, W. Shi, A. Zhou, M. Zhan, and H. Li, "WebGen-Bench: Evaluating LLMs on Generating Interactive and Functional Websites from Scratch," Aug. 2025, arXiv:2505.03733. [Online]. Available: http://arxiv.org/abs/2505.03733 [Page 54.]

[96] L. J. Jacobsen and K. E. Weber, "The Promises and Pitfalls of Large Language Models as Feedback Providers: A Study of Prompt Engineering and the Quality of AI-Driven Feedback," *AI*, vol. 6, no. 2, p. 35, Feb. 2025. doi: 10.3390/ai6020035. [Online]. Available: https://www.mdpi.com/2673-2688/6/2/35 [Page 55.]

[97] D. Schreiter, "Prompt Engineering: How Prompt Vocabulary affects Domain Knowledge," 2025. [Online]. Available: https://arxiv.org/abs/2505.17037 [Page 55.]

[98] W. Qingwen, "Research on Personalized Art Design and Customized Product Development Driven by Artificial Intelligence," *International Journal of High Speed Electronics and Systems*, p. 2540502, Apr. 2025. doi: 10.1142/S0129156425405029. [Online]. Available: https://www.worldscientific.com/doi/10.1142/S0129156425405029 [Page 55.]

[99] T. Neusesser and K. Moran, "Designing Use-Case Prompt Suggestions," Jun. 2025. [Online]. Available: https://www.nngroup.com/articles/designing-use-case-prompt-suggestions/ [Page 55.]

[100] L. Tankelevitch, V. Kewenig, A. Simkute, A. E. Scott, A. Sarkar, A. Sellen, and S. Rintel, "The Metacognitive Demands and Opportunities of Generative AI," in *Proceedings of the CHI Conference on Human Factors in Computing Systems*. Honolulu HI USA: ACM, May 2024. doi: 10.1145/3613904.3642902. ISBN 9798400703300 pp. 1–24. [Online]. Available: https://dl.acm.org/doi/10.1145/3613904.3642902 [Page 56.]

[101] J. T. Liang, C. Yang, and B. A. Myers, "A Large-Scale Survey on the Usability of AI Programming Assistants: Successes and Challenges," 2023. [Online]. Available: https://arxiv.org/abs/2303.17125 [Page 56.]

[102] T. Mugunthan, "Researching the Usability of Early Generative-AI Tools," Dec. 2023. [Online]. Available: https://www.nngroup.com/articles/early-ai-tools-methodology/ [Page 56.]

[103] S. Patel and A. S. Gadwal, "Adaptive Interface Framework: Balancing Automation and user Control for an Intuitive and Predictable user

Experience," *International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)*, vol. 14, no. 3, Mar. 2025. doi: 10.15680/IJIRSET.2025.1403360 [Page 56.]

Appendix A

Supporting Materials

A.1 Usability Testing Materials

Informed Consent Form

You are invited to participate in a study to assess a generative user interface driven by a large language model (LLM). This study aims to compare the traditional static User Interface (UI) with an LLM-generated dynamic UI in a fitness planning app.

Procedures: You will be asked to interact with two different versions of the AdaptFit app. In the baseline phase, you will perform tasks such as adding, editing, and deleting fitness plans. In the adaptive phase, you will provide personal information, view a personalized UI/UX proposal generated by the app, modify the proposal, and then generate an enhanced UI. You will perform the same tasks as the last step on the enhanced UI. You will complete a pre-test questionnaire and a post-test survey and participate in a brief interview about your experience.

Confidentiality: Your data and responses will be kept confidential and anonymized. All recorded data will be stored securely in the local storage of the researcher's computer.

Voluntary Participation: Your participation is voluntary, and you may withdraw from the study at any time without penalty.

Contact Information: For questions about the study, contact Yutong Fang

at yutongfang918@gmail.com. By signing below, you acknowledge that you have read the above information and agree to participate in this study.

Task Instructions

Please go through the following tasks step by step, following the given order.

Task 1: Interact with initial UI

- Add a new fitness plan.
- Edit an existing plan.
- Delete a plan.

Task 2: Input information for personalized UI

- Navigate to the profile page and enter your personal details *†.
- Click "Generate personalized UI/UX proposal" trigger the LLM, you can click multiple times to generate different proposals.
- View and modify the generated proposal to tailor to your needs.
- Once you are done, click "Generate enhanced UI" and wait until the new UI is ready.

Task 3: Interact with personalized UI

- Add a new fitness plan.
- Edit an existing plan.
- Delete a plan.

Task 4: Iterate

We encourage you to modify the proposal a couple more times to interact with more generated UIs.

---

*To respect your privacy, please feel free to skip the gender and age questions if you're uncomfortable sharing that information.

†Note that needs don't necessarily need to be accessibility needs, it can be any preference you want for the UI (e.g. color, icons, font styles, intuitive UI, etc.)

Interview Questions

1. Can you describe your overall experience? How would you compare your experience using the baseline UI with the enhanced UI?
2. How satisfied are you with the enhanced UI? Which features of the adaptive interface were the most useful or appealing? Did you find that the enhanced UI improved your efficiency?
3. Did you encounter any errors, unexpected behavior, or unclear instructions in the enhanced UI that you did not experience in the baseline version? Any pain points/frustrations?
4. Do you feel that the enhanced interface was well-tailored to your personal needs and preferences? Why or why not?
5. How did modifying the proposal (profile) affect your satisfaction with the final UI?/Did having the option to tailor the suggestions increase your sense of control over the app's look and functionality?
6. Would you feel comfortable with an interface that continuously adapts based on your input? What would be your ideal balance between adaptation and consistency?
7. What improvements would you suggest for making the adaptive UI more personalized?
8. Do you have any additional comments or feedback that you would like to share?

Pre-testing Questionnaire

https://forms.gle/KCqp62xx821kudiD6

Post-testing Questionnaire

https://forms.gle/GCGiZBDkpsig8h3n9

A.2 Prompts

Prompt for Proposal Generation

```
Context:
You are a professional UI/UX designer specializing in
   ↪ fitness applications.
Your task is to refine and enhance the homepage of the
   ↪ workout app Adapt Fit, ensuring it aligns with
   ↪ user needs.
The app currently features a section where users can
   ↪ add, edit, and delete their workout plans with a
   ↪ simple, basic UI layout.

Objective:
Based on the given user info and workout routine,
   ↪ analyze the user's needs and suggest UI/UX
   ↪ proposals for the homepage to improve usability,
   ↪ engagement, and personalization.
Suggestions should only relate to layout, style, and
   ↪ control properties - no logic, navigation, or
   ↪ backend modifications.

User info:
- Username: $username
- Age: $age
- Gender: $gender
- Needs: $needs

Workout routine (including exerciseName, Avg. Spent
   ↪ Time on this sport, How often do they do this
   ↪ sport per week, and special notes for this
   ↪ sport): $sportsDetails

Output Format:
Please only provide a numbered list of 5-8 UI/UX
   ↪ suggestions. Each suggestion should be:
- Written in simple, user-facing language without any
   ↪ technical details (max ~15 words)
- Immediately followed by a [dev] line that provides a
   ↪ concise developer-facing implementation hint

These hints are used by the development team to
   ↪ implement your design using a JSON-to-UI layout
```

```
   ↪ system.
```

Listing A.1: Prompt instruction for generating UI/UX proposals

LLM Prompt for UI JSON specification Generation

```
Your task is to interpret and implement UI/UX changes
   ↪ requested by a designer. The suggestions are in a
   ↪ two-tier format:
A user-facing description and a developer-facing note
   ↪ prefixed with [dev], which contains implementable
   ↪ guidance.

Input:'
Designers requests: $uiEnhancementProposal

Structure:
You will create JSON representations of three Android
   ↪ layout files based on the proposal:

- fragment_home.xml:
    - Defines the structure of the fitness app's
       ↪ homepage.
    - Includes:
        - Title (tvTitle)
        - CardView (cardPlan), containing:
            - Placeholder (tvNoPlan)
            - RecyclerView (rvPlans)
        - Floating Action Button (fabAddPlan), always
           ↪ assigned icons via remote URL using
           ↪ "android:src": "https://..."

- item_plan.xml:
    - Defines workout list item template.
    - CardView with:
        - Exercise Name (tvExerciseName)
        - Completion Control (cbCompleted)
        - Details (tvDetails)
        - Edit/Delete Buttons (btnEdit, btnDelete)

- dialog_add_plan.xml:
```

```
    - The workout adding/editing form.
    - Seven TextInputLayouts (tilExerciseName,
       ↪ tilDuration, ... tilNotes).
    - Use "android:hint" in TextInputLayout and
       ↪ optional pre-filled "android:text".

Expected Output:
Return pure JSON:
{
  "fragment_home": { ... },
  "item_plan": { ... },
  "dialog_add_plan": { ... },
  "resources": {
    "drawables": { ... },
    "fonts": { ... }
  }
}

Instructions:
- Understand 'Designers Requests
- Follow Android Development Best Practices
- Use hex codes for colors
- Use Google Fonts via fontFamily URLs
- Only allow icons via:
    1. Remote image (android:src)
    2. IconicsTextView with "ico:text" (gmd: prefix)

Resources Reference Constraints:
- Any "@drawable/xxx" reference must have full
   ↪ definition in resources.drawables
- Drawable type must be "shape" or remote "src" URL
- Do NOT include vectorDrawable content or XML paths

Technical Constraints:
- All layout constraints inside attributes with app:
   ↪ prefix (e.g., app:layout_constraintTop_toTopOf)
- Use layout_width "0dp" inside ConstraintLayout
- Preserve all view IDs
- Correct attribute namespaces (android: app:)
- Ensure the layout is valid and ready for Android
```

```
↪ project usage.
```

Listing A.2: Prompt instruction for generating UI JSON specification

A.3 Pseudocode

JsonLayoutInflater (to parse the generated JSON layout into actual Android UI components)

```
class JsonLayoutInflater {

    fun inflate(jsonLayout: Java, parent: ViewGroup): View {
        val layoutType = jsonLayout.getString("type")
        val view: View = when (layoutType) {
            "ConstraintLayout" -> createConstraintLayout(parent).apply {
                applyConstraints(this, jsonLayout.getJSONObject("constraints"))
            }
            "TextView" -> createTextView(parent).apply {
                text = jsonLayout.getString("text")
                setFont(jsonLayout.getString("font"))
                setColor(jsonLayout.getString("color"))
            }
            "Button" -> createButton(parent).apply {
                text = jsonLayout.getString("text")
            }
            else -> throw IllegalArgumentException("Unsupported component type")
        }

        // Recursively handle child views
        jsonLayout.optJSONArray("children")?.forEach { childJson ->
            val childView = inflate(childJson, view as ViewGroup)
            (view as ViewGroup).addView(childView)
        }
        return view
```

```
    }
}
```

Listing A.3: Core logic of JsonLayoutInflater

EnhancedHomeFragment (to attach behaviors to dynamically rendered views)

```
class EnhancedHomeFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
            savedInstanceState: Bundle?
    ): View? {
        val jsonLayout = loadGeneratedJson()
        val rootView = JsonLayoutInflater().inflate(
           jsonLayout, container!!)

        attachClickListeners(rootView)
        bindDynamicData(rootView)
        return rootView
    }

    private fun attachClickListeners(view: View) {
        if (view is Button && view.id == R.id.btnSubmit)
            {
            view.setOnClickListener { handleSubmit() }
        }
        // Recursively attach listeners for child views
    }
}
```

Listing A.4: EnhancedHomeFragment logic

A.4 Thematic Analysis Results

Theme	Code
~~Aesthetic Strengths of the System~~	Positive reaction to visual styling

Theme	**Code**
	Functional affordance increases motivation
Benefits of Personalized Design	Benefits of adaptive personalization Personalization increases engagement Functional affordance increases motivation Personalization appreciated Feeling of co-creation and ownership Personalized emotional experience
Enjoyment from Co-Creation	Process satisfaction Appreciation for extended suggestion based on the input Enjoyment from co-creative personalization Feeling of co-creation and ownership Curiosity of AI-generated results Exploration as play Preference for generated version (when successful)
Exploration-Oriented Interaction	Personalization appreciated Curiosity of AI-generated results Exploration as play Explore the potential of LLM Iterative experimentation Trial-and-error driven learning Resistance to manual adjustment
Frustration from Manual Editing	Preference for automatic inference over manual input Frustration with manual editing Request for conversational UI
Frustration over Unmet Needs	Trial-and-error driven learning Frustration to unmet expectations Disappointment from lack of variation Frustration between intent and outcome Disappointment with unfulfilled needs Icon rendering error causes frustration Reliability concerns Proposal-UI mismatch Inconsistent implementation System doesn't follow user input

Theme	Code
	Incomplete execution despite good interpretation
	Disappointment with unfulfilled needs
Lack of Guidance & Prompting Support	User confusion when formulating input
	Uncertainty about what's possible to change
	Request for prompting templates
	User lacks domain knowledge of UI components
	Prompt formulation difficulty
	Over-freedom leads to paralysis
	Desire for more guidance
	Preference for "guided freedom"
	Preference for choice over coming up with needs
Mismatch Between Proposal and UI	Proposal-UI mismatch
	Inconsistent implementation
	Disappointment with unfulfilled needs
	System doesn't follow user input
	Hallucination
Positive Reception of the Concept	Novelty of AI-generated UI
	Positive attitude toward core idea
	Positive perception of adaptive UI
	Positive attitude to AI collaboration
	Recognizing potential despite limitations
	Wish for improved performance
Prefer Controlled Adaptation	Prefer selective control over changes
	Over-freedom leads to paralysis
	Desire for more guidance
	Preference for "guided freedom"
	Preference for choice over coming up with needs
	Preference controlled adaptation
Simple Onboarding	Ease of initial interaction
System Constraints & LLM Gap	Desire for more icon variety
	Usability issue with button flow
	Friction from input method
	Incomplete execution despite good interpretation
	Hallucination
	Icon rendering error causes frustration
	System doesn't follow user input

Theme	Code
	Recognizing potential despite limitations
Trust and Performance Dependency	Satisfaction linked to need fulfillment Partial satisfaction Wish for improved performance Conditional trust Recognizing potential despite limitations Reliability concerns Preference for generated version (when successful)
User Desire and Suggestions	Conditional trust Opposition to excessive dynamism Favoring predictable behavior Personalized emotional experience Resistance to manual adjustment Preference for automatic inference over manual input Frustration with manual editing Request for conversational UI Preference controlled adaptation Prefer selective control over changes Over-freedom leads to paralysis Desire for more guidance Preference for “guided freedom” Preference for choice over coming up with needs

A.5 Example Generated Results

Example 1

User Needs

```
User needs: "clean interface, mild colors that are not
   too strong for eyes, intuitive design, simple"
```

Listing A.5: User needs

AI-generated Proposal

```
1. Use a soft, neutral background color like #FAFAFA for
    the entire homepage, and apply gentle accent colors
   such as #A8D5BA for headers and highlights to reduce
   eye strain. [dev] Set `android:backgroundColor="#
   FAFAFA"` for the root layout; change header text
   color to #A8D5BA in `tvTitle` and other headers.
2. Replace the default placeholder and list with a clean
   , centered message in a subtle color (#757575) when
   no plans exist, and smoothly fade in the plan list
   when available. [dev] Make `tvNoPlan` centered with `
   android:textColor="#757575"`; set its visibility to
   visible only when plans are empty; animate `rvPlans`
   to fade in on data load.
3. Style `CardView` containers and plan items with
   rounded corners, subtle shadow, and mild background
   colors (#FFFFFF with slight elevation) to create a
   gentle, organized look. [dev] Add `cardElevation="2dp
   "`, `cornerRadius="8dp"` to `CardView` in `
   fragment_home.xml`, and set background color to #
   FFFFFF.
4. Simplify controls: change the toggle for completion
   to a small, soothing switch in soft green (#8BC34A),
   and give all buttons rounded corners and a muted
   color palette for consistency. [dev] Use `Switch`
   with `android:thumbTint="#8BC34A"`; style `btnEdit`
   and `btnDelete` with rounded corners, light gray
   background, and grey text.
5. Enhance the add/edit dialog with a clean, vertical
   layout, gentle padding, and hint texts in a calm font
   . Use a light background (#FFFFFF) with subtle
   borders to focus user attention. [dev] Use a `
   LinearLayout` with `orientation="vertical"` inside `
   dialog_add_plan.xml`, set background to #FFFFFF,
   apply consistent `marginTop="8dp"` between `
   TextInputLayout`s, and ensure all inputs have clear
   hints."
```

Listing A.6: AI-generated proposal

AI-generated UI JSON Specification

```
{
  "fragment_home": {
    "type": "ConstraintLayout",
    "attributes": {
      "android:layout_width": "match_parent",
      "android:layout_height": "match_parent",
      "android:background": "#FAFAFA"
    },
    "children": [
      {
        "type": "TextView",
        "attributes": {
          "android:id": "@+id/tvTitle",
          "android:layout_width": "wrap_content",
          "android:layout_height": "wrap_content",
          "android:text": "Your␣Workouts",
          "android:textSize": "24sp",
          "android:textColor": "#A8D5BA",
          "android:textStyle": "bold",
          "android:fontFamily": "sans-serif-medium",
          "android:layout_marginTop": "16dp",
          "android:layout_marginBottom": "16dp",
          "android:layout_marginStart": "16dp",
          "android:layout_marginEnd": "16dp",
          "app:layout_constraintTop_toTopOf": "parent",
          "app:layout_constraintStart_toStartOf": "parent",
          "app:layout_constraintEnd_toEndOf": "parent"
        }
      },
      {
        "type": "CardView",
        "attributes": {
          "android:id": "@+id/cardPlan",
          "android:layout_width": "0dp",
          "android:layout_height": "0dp",
          "android:layout_margin": "16dp",
          "android:background": "#FFFFFF",
          "app:cardElevation": "2dp",
          "app:cornerRadius": "8dp",
          "app:layout_constraintTop_toBottomOf": "@+id/tvTitle",
          "app:layout_constraintStart_toStartOf": "parent",
          "app:layout_constraintEnd_toEndOf": "parent",
          "app:layout_constraintBottom_toTopOf": "@+id/fabAddPlan"
        },
        "children": [
          {
            "type": "TextView",
            "attributes": {
              "android:id": "@+id/tvNoPlan",
              "android:layout_width": "match_parent",
              "android:layout_height": "match_parent",
              "android:text": "No␣plans␣available.␣Tap␣+␣to␣add␣a␣workout.",
              "android:textColor": "#757575",
              "android:textSize": "16sp",
              "android:gravity": "center",
              "android:visibility": "visible"
            }
          },
          {
            "type": "RecyclerView",
            "attributes": {
              "android:id": "@+id/rvPlans",
              "android:layout_width": "match_parent",
              "android:layout_height": "match_parent",
              "android:visibility": "gone"
            }
          }
        ]
      },
      {
        "type": "FloatingActionButton",
        "attributes": {
          "android:id": "@+id/fabAddPlan",
          "android:layout_width": "wrap_content",
          "android:layout_height": "wrap_content",
          "android:src": "https://img.icons8.com/ios-filled/24/000000/plus.png",
          "android:contentDescription": "Add␣plan",
          "app:layout_constraintBottom_toBottomOf": "parent",
          "app:layout_constraintEnd_toEndOf": "parent",
          "android:layout_margin": "16dp"
```

```
        }
      }
    ]
  },
  "item_plan": {
    "type": "CardView",
    "attributes": {
      "android:id": "@+id/cardItem",
      "android:layout_width": "match_parent",
      "android:layout_height": "wrap_content",
      "android:layout_margin": "8dp",
      "android:background": "#FFFFFF",
      "app:cardElevation": "2dp",
      "app:cornerRadius": "8dp"
    },
    "children": [
      {
        "type": "LinearLayout",
        "attributes": {
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:orientation": "vertical",
          "android:padding": "16dp"
        },
        "children": [
          {
            "type": "LinearLayout",
            "attributes": {
              "android:layout_width": "match_parent",
              "android:layout_height": "wrap_content",
              "android:orientation": "horizontal",
              "android:layout_marginBottom": "8dp",
              "android:gravity": "center_vertical"
            },
            "children": [
              {
                "type": "TextView",
                "attributes": {
                  "android:id": "@+id/tvExerciseName",
                  "android:layout_width": "0dp",
                  "android:layout_height": "wrap_content",
                  "android:text": "Exercise␣Name",
                  "android:textSize": "18sp",
                  "android:textColor": "#212121",
                  "android:layout_weight": "1"
                }
              },
              {
                "type": "Switch",
                "attributes": {
                  "android:id": "@+id/cbCompleted",
                  "android:layout_width": "wrap_content",
                  "android:layout_height": "wrap_content",
                  "android:thumbTint": "#8BC34A"
                }
              }
            ]
          },
          {
            "type": "TextView",
            "attributes": {
              "android:id": "@+id/tvDetails",
              "android:layout_width": "match_parent",
              "android:layout_height": "wrap_content",
              "android:text": "Details␣here",
              "android:textSize": "14sp",
              "android:textColor": "#757575"
            }
          },
          {
            "type": "LinearLayout",
            "attributes": {
              "android:layout_width": "match_parent",
              "android:layout_height": "wrap_content",
              "android:orientation": "horizontal",
              "android:layout_gravity": "end",
              "android:layout_marginTop": "8dp"
            },
            "children": [
              {
```

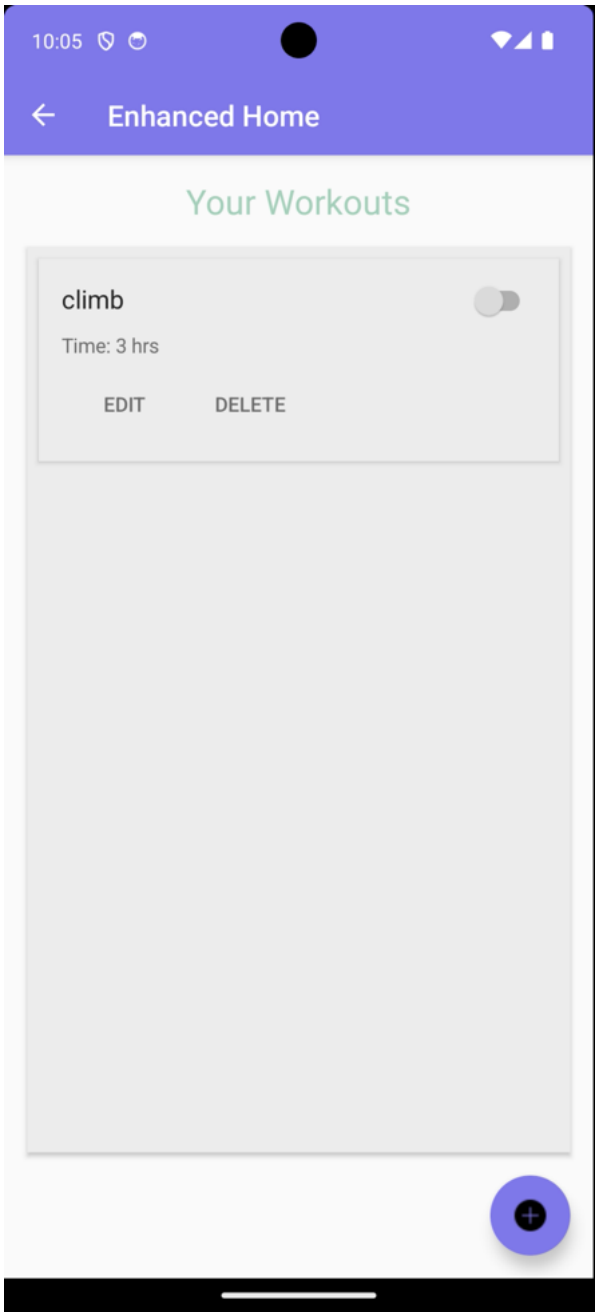
```
            "type": "Button",
            "attributes": {
              "android:id": "@+id/btnEdit",
              "android:layout_width": "wrap_content",
              "android:layout_height": "wrap_content",
              "android:text": "Edit",
              "android:background": "@drawable/rounded_button",
              "android:textColor": "#757575",
              "android:padding": "8dp"
            }
          },
          {
            "type": "Button",
            "attributes": {
              "android:id": "@+id/btnDelete",
              "android:layout_width": "wrap_content",
              "android:layout_height": "wrap_content",
              "android:layout_marginStart": "8dp",
              "android:text": "Delete",
              "android:background": "@drawable/rounded_button",
              "android:textColor": "#757575",
              "android:padding": "8dp"
            }
          }
        ]
      }
    ]
  }
},
"dialog_add_plan": {
  "type": "LinearLayout",
  "attributes": {
    "android:layout_width": "match_parent",
    "android:layout_height": "wrap_content",
    "android:orientation": "vertical",
    "android:padding": "16dp",
    "android:background": "#FFFFFF"
  },
  "children": [
    {
      "type": "TextInputLayout",
      "attributes": {
        "android:id": "@+id/tilExerciseName",
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:hint": "Exercise␣Name",
        "android:layout_marginTop": "8dp"
      },
      "children": [
        {
          "type": "TextInputEditText",
          "attributes": {
            "android:id": "@+id/etExerciseName",
            "android:layout_width": "match_parent",
            "android:layout_height": "wrap_content",
            "android:minHeight": "48dp",
            "android:paddingVertical": "12dp",
            "android:hint": "Enter␣exercise␣name"
          }
        }
      ]
    },
    {
      "type": "TextInputLayout",
      "attributes": {
        "android:id": "@+id/tilTime",
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:hint": "Time",
        "android:layout_marginTop": "8dp"
      },
      "children": [
        {
          "type": "TextInputEditText",
          "attributes": {
            "android:id": "@+id/etTime",
            "android:layout_width": "match_parent",
            "android:layout_height": "wrap_content",
            "android:minHeight": "48dp",
```

```
          "android:paddingVertical": "12dp",
          "android:hint": "Enter time"
        }
      }
    ]
  },
  {
    "type": "TextInputLayout",
    "attributes": {
      "android:id": "@+id/tilDistance",
      "android:layout_width": "match_parent",
      "android:layout_height": "wrap_content",
      "android:hint": "Distance",
      "android:layout_marginTop": "8dp"
    },
    "children": [
      {
        "type": "TextInputEditText",
        "attributes": {
          "android:id": "@+id/etDistance",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:minHeight": "48dp",
          "android:paddingVertical": "12dp",
          "android:hint": "Enter distance"
        }
      }
    ]
  },
  {
    "type": "TextInputLayout",
    "attributes": {
      "android:id": "@+id/tilSets",
      "android:layout_width": "match_parent",
      "android:layout_height": "wrap_content",
      "android:hint": "Sets",
      "android:layout_marginTop": "8dp"
    },
    "children": [
      {
        "type": "TextInputEditText",
        "attributes": {
          "android:id": "@+id/etSets",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:minHeight": "48dp",
          "android:paddingVertical": "12dp",
          "android:hint": "Enter number of sets"
        }
      }
    ]
  },
  {
    "type": "TextInputLayout",
    "attributes": {
      "android:id": "@+id/tilReps",
      "android:layout_width": "match_parent",
      "android:layout_height": "wrap_content",
      "android:hint": "Reps",
      "android:layout_marginTop": "8dp"
    },
    "children": [
      {
        "type": "TextInputEditText",
        "attributes": {
          "android:id": "@+id/etReps",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:minHeight": "48dp",
          "android:paddingVertical": "12dp",
          "android:hint": "Enter number of repetitions"
        }
      }
    ]
  },
  {
    "type": "TextInputLayout",
    "attributes": {
      "android:id": "@+id/tilWeight",
      "android:layout_width": "match_parent",
```

```
          "android:layout_height": "wrap_content",
          "android:hint": "Weight",
          "android:layout_marginTop": "8dp"
        },
        "children": [
          {
            "type": "TextInputEditText",
            "attributes": {
              "android:id": "@+id/etWeight",
              "android:layout_width": "match_parent",
              "android:layout_height": "wrap_content",
              "android:minHeight": "48dp",
              "android:paddingVertical": "12dp",
              "android:hint": "Enter weight"
            }
          }
        ]
      },
      {
        "type": "TextInputLayout",
        "attributes": {
          "android:id": "@+id/tilNotes",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:hint": "Notes",
          "android:layout_marginTop": "8dp"
        },
        "children": [
          {
            "type": "TextInputEditText",
            "attributes": {
              "android:id": "@+id/etNotes",
              "android:layout_width": "match_parent",
              "android:layout_height": "wrap_content",
              "android:minHeight": "48dp",
              "android:paddingVertical": "12dp",
              "android:hint": "Additional notes"
            }
          }
        ]
      }
    ]
  },
  "resources": {
    "drawables": {
      "rounded_button": {
        "type": "shape",
        "content": {
          "shape": "rectangle",
          "corners": {
            "radius": "8dp"
          },
          "solid": {
            "color": "#E0E0E0"
          }
        }
      }
    },
    "fonts": {
      "sans-serif": {
        "fontFamilyUrl": "https://fonts.gstatic.com/s/roboto/v30/KFOmCnqEu92Fr1Mu4mxM.woff2"
      },
      "sans-serif-medium": {
        "fontFamilyUrl": "https://fonts.gstatic.com/s/roboto/v30/KFOjCnqEu92Fr1Mu51xIIzIXKM6vWA.woff2"
      }
    }
  }
}
```


Listing A.7: AI-generated UI JSON specification

The Rendered UI



Example 2

User Needs

```
User needs: ”Make colour pink, make cute, can you add time options for frequency and average time to experience/plan also make sure that the colour change and icons dont get errors, make sure that the icons and interactive things are big enough to read, can you make the delete button the switch because its confusing, also dont add any extra icons
”
```

Listing A.8: User needs

AI-generated Proposal

```
1. Use soft pink background with cute, rounded card designs for workout plans to make it appealing. [dev] Set parent layout background to #FFB6C1; apply corner radius and shadow to CardViews; use pink accents for headers and buttons. 2. Make all control icons and buttons larger, with clear labels, for easy readability and interaction. [dev] Increase icon sizes using ico:size to 24dp or 32dp; ensure buttons have sufficient padding and font sizes. 3. Replace the delete button with a toggle switch labeled “”Delete for clearer, more intuitive control. [dev] Swap btnDelete with a Switch view labeled ”Delete”, set toggle style, and connect its state to delete action. 4. Add pink-themed, cute icons for add, edit, and mark complete (e.g., hearts, stars) to enhance engagement. [dev] Assign remote icons via ”android: src” with pink-colored icons (e.g., heart, star); keep icons consistent and error-free. 5. Provide options to select workout frequency with dropdown menus (e.g., ”Twice a month”) and estimated time inputs. [dev] Insert Spinner views for frequency options and EditTexts with hints for average time estimates, styled in pink. 6. Highlight the “Add ”Plan FloatingActionButton with a larger size and a cute plus icon in pink. [dev] Increase fab ADD size
```

```
to 56dp, set icon "ico:plus" with pink tint, and add
padding for prominence. 7. Style the layout with
playful, rounded fonts and subtle pink accents,
ensuring all text and touch targets are large. [dev]
Use Google Fonts URL (e.g., Raleway), set fontSize to
 16sp+, and apply consistent pink colors across text
and controls. 8. Keep the interface simple by
maintaining large spacing between elements and
avoiding clutter, ensuring clarity and ease of use. [
dev] Use margin and padding of at least 16dp around
interactive elements; align items centrally for
balance.
```

Listing A.9: AI-generated proposal

AI-generated UI JSON Specification


```
{
  "fragment_home": {
    "type": "ConstraintLayout",
    "attributes": {
      "android:layout_width": "match_parent",
      "android:layout_height": "match_parent",
      "android:background": "#FFB6C1"
    },
    "children": [
      {
        "type": "TextView",
        "attributes": {
          "android:id": "@+id/tvTitle",
          "android:layout_width": "wrap_content",
          "android:layout_height": "wrap_content",
          "android:text": "Workout␣Plans",
          "android:textSize": "24sp",
          "android:fontFamily": "@font/raleway_regular",
          "android:textColor": "#FF69B4",
          "app:layout_constraintTop_toTopOf": "parent",
          "app:layout_constraintStart_toStartOf": "parent",
          "app:layout_constraintEnd_toEndOf": "parent",
          "android:layout_marginTop": "24dp",
          "android:layout_marginBottom": "16dp"
        }
      },
      {
        "type": "CardView",
        "attributes": {
          "android:id": "@+id/cardPlan",
          "android:layout_width": "0dp",
          "android:layout_height": "wrap_content",
          "android:background": "#FFE4E1",
          "app:cardCornerRadius": "16dp",
          "app:cardElevation": "4dp",
          "android:layout_margin": "16dp",
          "app:layout_constraintTop_toBottomOf": "@+id/tvTitle",
          "app:layout_constraintStart_toStartOf": "parent",
          "app:layout_constraintEnd_toEndOf": "parent"
        },
        "children": [
          {
            "type": "FrameLayout",
            "attributes": {
              "android:layout_width": "match_parent",
              "android:layout_height": "wrap_content"
            },
```

```
        "children": [
          {
            "type": "TextView",
            "attributes": {
              "android:id": "@+id/tvNoPlan",
              "android:layout_width": "wrap_content",
              "android:layout_height": "wrap_content",
              "android:text": "No workout plan added yet",
              "android:textSize": "16sp",
              "android:textColor": "#FF69B4",
              "android:visibility": "visible",
              "android:layout_gravity": "center",
              "android:padding": "16dp"
            }
          },
          {
            "type": "RecyclerView",
            "attributes": {
              "android:id": "@+id/rvPlans",
              "android:layout_width": "match_parent",
              "android:layout_height": "wrap_content",
              "android:visibility": "gone",
              "android:padding": "8dp"
            }
          }
        ]
      }
    ]
  },
  {
    "type": "com.google.android.material.floatingactionbutton.FloatingActionButton",
    "attributes": {
      "android:id": "@+id/fabAddPlan",
      "android:layout_width": "56dp",
      "android:layout_height": "56dp",
      "android:src": "https://img.icons8.com/ios-filled/48/ff69b4/plus-math.png",
      "android:tint": "#FF69B4",
      "android:contentDescription": "Add Plan",
      "android:padding": "16dp",
      "app:layout_constraintBottom_toBottomOf": "parent",
      "app:layout_constraintEnd_toEndOf": "parent",
      "android:layout_margin": "16dp"
    }
  }
]
},
"item_plan": {
  "type": "CardView",
  "attributes": {
    "android:layout_width": "match_parent",
    "android:layout_height": "wrap_content",
    "android:background": "#FFE4E1",
    "app:cardCornerRadius": "16dp",
    "app:cardElevation": "4dp",
    "android:layout_margin": "8dp"
  },
  "children": [
    {
      "type": "LinearLayout",
      "attributes": {
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:orientation": "vertical",
        "android:padding": "16dp"
      },
      "children": [
        {
          "type": "LinearLayout",
          "attributes": {
            "android:layout_width": "match_parent",
            "android:layout_height": "wrap_content",
            "android:orientation": "horizontal",
            "app:layout_constraintBottom_toBottomOf": "",
            "android:layout_marginBottom": "8dp",
            "android:gravity": "center_vertical"
          },
          "children": [
            {
              "type": "TextView",
              "attributes": {
```

```
            "android:id": "@+id/tvExerciseName",
            "android:layout_width": "0dp",
            "android:layout_height": "wrap_content",
            "android:layout_weight": "1",
            "android:text": "Exercise",
            "android:textSize": "18sp",
            "android:fontFamily": "@font/raleway_regular",
            "android:textColor": "#FF69B4"
          }
        },
        {
          "type": "Switch",
          "attributes": {
            "android:id": "@+id/cbCompleted",
            "android:layout_width": "wrap_content",
            "android:layout_height": "wrap_content",
            "android:text": "Completed",
            "android:textSize": "16sp",
            "android:textColor": "#FF69B4",
            "android:paddingStart": "8dp"
          }
        }
      ]
    },
    {
      "type": "TextView",
      "attributes": {
        "android:id": "@+id/tvDetails",
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:text": "Workout␣details...",
        "android:textSize": "14sp",
        "android:textColor": "#FF69B4"
      }
    },
    {
      "type": "LinearLayout",
      "attributes": {
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:orientation": "horizontal",
        "android:layout_gravity": "end",
        "android:marginTop": "8dp"
      },
      "children": [
        {
          "type": "Button",
          "attributes": {
            "android:id": "@+id/btnEdit",
            "android:layout_width": "wrap_content",
            "android:layout_height": "wrap_content",
            "android:text": "Edit",
            "android:textSize": "16sp",
            "android:fontFamily": "@font/raleway_regular",
            "android:textColor": "#FF69B4",
            "android:padding": "8dp"
          }
        },
        {
          "type": "Button",
          "attributes": {
            "android:id": "@+id/btnDelete",
            "android:layout_width": "wrap_content",
            "android:layout_height": "wrap_content",
            "android:text": "Delete",
            "android:textSize": "16sp",
            "android:fontFamily": "@font/raleway_regular",
            "android:textColor": "#FF69B4",
            "android:padding": "8dp",
            "android:layout_marginStart": "8dp"
          }
        }
      ]
    }
  ]
},
"dialog_add_plan": {
  "type": "ScrollView",
```

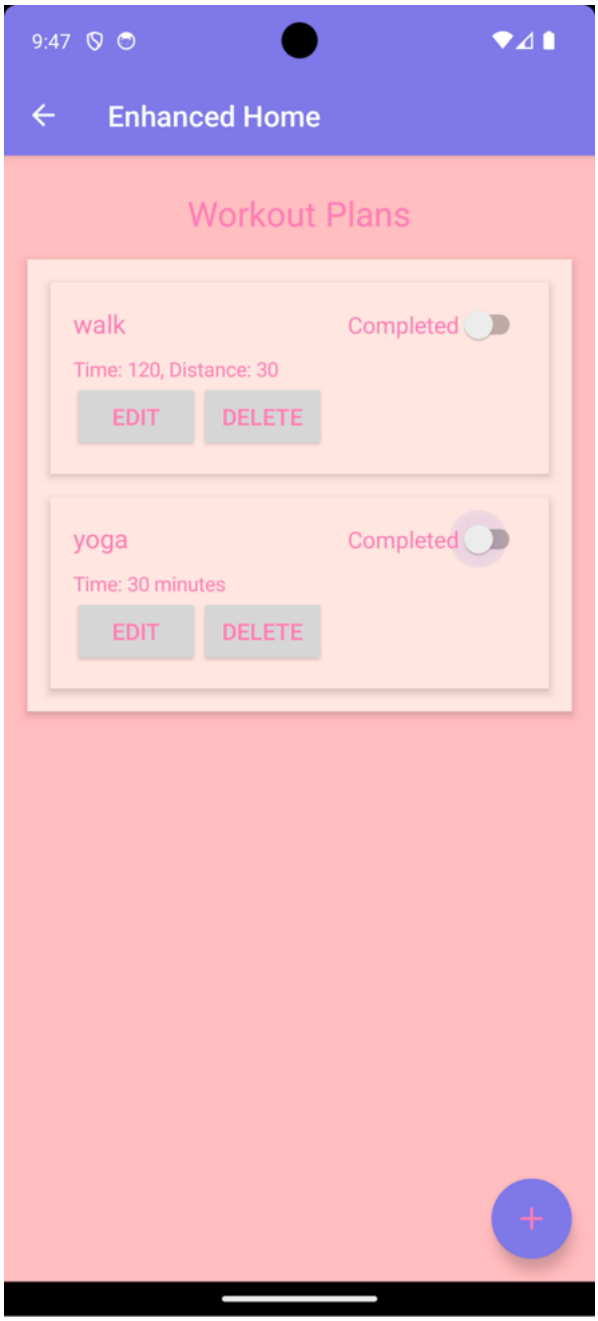
```
"attributes": {
  "android:layout_width": "match_parent",
  "android:layout_height": "wrap_content",
  "android:padding": "24dp"
},
"children": [
  {
    "type": "LinearLayout",
    "attributes": {
      "android:layout_width": "match_parent",
      "android:layout_height": "wrap_content",
      "android:orientation": "vertical"
    },
    "children": [
      {
        "type": "TextInputLayout",
        "attributes": {
          "android:id": "@+id/tilExerciseName",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:hint": "Exercise␣Name",
          "android:background": "#FFB6C1",
          "android:shapeAppearance": "rounded"
        },
        "children": [
          {
            "type": "TextInputEditText",
            "attributes": {
              "android:id": "@+id/etExerciseName",
              "android:layout_width": "match_parent",
              "android:layout_height": "wrap_content",
              "android:fontFamily": "@font/raleway_regular",
              "android:textSize": "16sp",
              "android:text": "",
              "android:hint": ""
            }
          }
        ]
      },
      {
        "type": "TextInputLayout",
        "attributes": {
          "android:id": "@+id/tilDuration",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:hint": "Duration␣(mins)"
        },
        "children": [
          {
            "type": "TextInputEditText",
            "attributes": {
              "android:id": "@+id/etDuration",
              "android:layout_width": "match_parent",
              "android:layout_height": "wrap_content",
              "android:fontFamily": "@font/raleway_regular",
              "android:textSize": "16sp",
              "android:hint": ""
            }
          }
        ]
      },
      {
        "type": "TextInputLayout",
        "attributes": {
          "android:id": "@+id/tilDistance",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:hint": "Distance"
        },
        "children": [
          {
            "type": "TextInputEditText",
            "attributes": {
              "android:id": "@+id/etDistance",
              "android:layout_width": "match_parent",
              "android:layout_height": "wrap_content",
              "android:fontFamily": "@font/raleway_regular",
              "android:textSize": "16sp",
              "android:hint": ""
            }
```

```
      }
    ]
  },
  {
    "type": "TextInputLayout",
    "attributes": {
      "android:id": "@+id/tilSets",
      "android:layout_width": "match_parent",
      "android:layout_height": "wrap_content",
      "android:hint": "Sets"
    },
    "children": [
      {
        "type": "TextInputEditText",
        "attributes": {
          "android:id": "@+id/etSets",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:fontFamily": "@font/raleway_regular",
          "android:textSize": "16sp",
          "android:hint": ""
        }
      }
    ]
  },
  {
    "type": "TextInputLayout",
    "attributes": {
      "android:id": "@+id/tilReps",
      "android:layout_width": "match_parent",
      "android:layout_height": "wrap_content",
      "android:hint": "Reps"
    },
    "children": [
      {
        "type": "TextInputEditText",
        "attributes": {
          "android:id": "@+id/etReps",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:fontFamily": "@font/raleway_regular",
          "android:textSize": "16sp",
          "android:hint": ""
        }
      }
    ]
  },
  {
    "type": "TextInputLayout",
    "attributes": {
      "android:id": "@+id/tilWeight",
      "android:layout_width": "match_parent",
      "android:layout_height": "wrap_content",
      "android:hint": "Weight"
    },
    "children": [
      {
        "type": "TextInputEditText",
        "attributes": {
          "android:id": "@+id/etWeight",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:fontFamily": "@font/raleway_regular",
          "android:textSize": "16sp",
          "android:hint": ""
        }
      }
    ]
  },
  {
    "type": "TextInputLayout",
    "attributes": {
      "android:id": "@+id/tilNotes",
      "android:layout_width": "match_parent",
      "android:layout_height": "wrap_content",
      "android:hint": "Notes"
    },
    "children": [
      {
        "type": "TextInputEditText",
```

```
              "attributes": {
                "android:id": "@+id/etNotes",
                "android:layout_width": "match_parent",
                "android:layout_height": "wrap_content",
                "android:fontFamily": "@font/raleway_regular",
                "android:textSize": "16sp",
                "android:hint": ""
              }
            }
          ]
        }
      ]
    }
  ],
  "resources": {
    "drawables": {
      "bg_rounded_card": {
        "shape": "shape",
        "attributes": {
          "shape": "rectangle",
          "corners": {
            "radius": "16dp"
          },
          "shadow": {
            "shadowColor": "#88000000",
            "shadowRadius": "4dp",
            "shadowDx": "0dp",
            "shadowDy": "2dp"
          }
        }
      },
      "ico_add": {
        "shape": "src",
        "src": "https://img.icons8.com/ios-filled/48/ff69b4/plus-math.png"
      },
      "ico_edit": {
        "shape": "src",
        "src": "https://img.icons8.com/ios-filled/48/ff69b4/edit.png"
      },
      "ico_complete": {
        "shape": "src",
        "src": "https://img.icons8.com/ios-filled/48/ff69b4/star.png"
      }
    },
    "fonts": {
      "raleway_regular": {
        "fontFamilyUrl": "https://fonts.gstatic.com/s/raleway/v30/KFOmCnqEu92Fr1Mu4mxM.woff2"
      }
    }
  }
}
```


Listing A.10: AI-generated UI JSON specification

The Rendered UI



Example 3

User Needs

```
User needs: "1. need a timetable 2. can be marked as
   completed"
```

Listing A.11: User needs

AI-generated Proposal

```
1. **Add a color palette toggle at the top for users to
   switch between different color schemes, enhancing
   personalization and visual comfort.** [dev] Include a
    horizontal LinearLayout beneath `tvTitle` with color
    option buttons or icons, each triggering theme
   changes; style buttons with round backgrounds and
   accessible labels.
2. **Incorporate a timetable section below the title,
   displaying workouts scheduled for the day with
   distinctive icons and colors for each activity.** [
   dev] Add a vertical LinearLayout with a header
   TextView; within it, create small item views
   containing icon (e.g., gmd:local_cafe) and text for
   each exercise time, styled with user-selected colors.
3. (User deleted this suggestion) **Provide a font style
    selector in the homepage allowing users to choose
   from popular fonts to customize text appearance.** [
   dev] Insert a dropdown or horizontal button group
   below the title; populate with options like "Sans
   Serif," "Serif," "Handwritten"; on selection, apply
   fontFamily from Google Fonts URLs to all text views
   including titles and plans.
4. **Replace the default exercise plan icons with unique
   , exercise-specific icons that reflect each workout
   type for quick visual recognition.** [dev] In `
   item_plan.xml`, add an `ImageView` or `
   IconicsTextView` at the start of each plan item;
   assign dynamic icons using URLs based on exerciseName
   , styled with distinct colors.
5. **Enhance the homepage layout with a subtle
   background gradient and modern card styles for
```

```
workout plans to improve visual engagement.** [dev]
Set a GradientDrawable background on `fragment_home.
xml`; style `CardView` with corner radius and shadow
elevation for `cardPlan` and individual plan cards,
maintaining existing IDs. --- These suggestions focus
 exclusively on layout, style, and control properties
 to enrich user experience and perception of
customization options."
```

Listing A.12: AI-generated proposal

AI-generated UI JSON Specification


```
{
  "fragment_home": {
    "type": "ConstraintLayout",
    "attributes": {
      "android:layout_width": "match_parent",
      "android:layout_height": "match_parent",
      "android:padding": "16dp",
      "android:background": "@drawable/bg_gradient"
    },
    "children": [
      {
        "type": "TextView",
        "attributes": {
          "android:id": "@+id/tvTitle",
          "android:layout_width": "wrap_content",
          "android:layout_height": "wrap_content",
          "android:text": "My␣Fitness␣App",
          "android:textSize": "24sp",
          "android:textStyle": "bold",
          "android:textColor": "#222222",
          "app:layout_constraintTop_toTopOf": "parent",
          "app:layout_constraintStart_toStartOf": "parent",
          "app:layout_constraintEnd_toEndOf": "parent",
          "android:layout_marginBottom": "8dp"
        }
      },
      {
        "type": "LinearLayout",
        "attributes": {
          "android:id": "@+id/llColorPalette",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:orientation": "horizontal",
          "app:layout_constraintTop_toBottomOf": "@id/tvTitle",
          "app:layout_constraintStart_toStartOf": "parent",
          "app:layout_constraintEnd_toEndOf": "parent",
          "android:layout_marginBottom": "8dp"
        },
        "children": [
          {
            "type": "Button",
            "attributes": {
              "android:id": "@+id/btnColor1",
              "android:layout_width": "40dp",
              "android:layout_height": "40dp",
              "android:background": "@drawable/round_button_bg_scheme1",
              "android:contentDescription": "Color␣Scheme␣1",
              "android:tag": "scheme1"
            }
          },
          {
            "type": "Button",
            "attributes": {
              "android:id": "@+id/btnColor2",
              "android:layout_width": "40dp",
```

```
        "android:layout_height": "40dp",
        "android:background": "@drawable/round_button_bg_scheme2",
        "android:contentDescription": "Color Scheme 2",
        "android:tag": "scheme2"
      }
    },
    {
      "type": "Button",
      "attributes": {
        "android:id": "@+id/btnColor3",
        "android:layout_width": "40dp",
        "android:layout_height": "40dp",
        "android:background": "@drawable/round_button_bg_scheme3",
        "android:contentDescription": "Color Scheme 3",
        "android:tag": "scheme3"
      }
    }
  ]
},
{
  "type": "LinearLayout",
  "attributes": {
    "android:id": "@+id/llTimetable",
    "android:layout_width": "match_parent",
    "android:layout_height": "wrap_content",
    "android:orientation": "vertical",
    "app:layout_constraintTop_toBottomOf": "@id/llColorPalette",
    "app:layout_constraintStart_toStartOf": "parent",
    "app:layout_constraintEnd_toEndOf": "parent",
    "android:layout_marginTop": "8dp"
  },
  "children": [
    {
      "type": "TextView",
      "attributes": {
        "android:id": "@+id/tvTimetableHeader",
        "android:layout_width": "wrap_content",
        "android:layout_height": "wrap_content",
        "android:text": "Today's Workouts",
        "android:textSize": "20sp",
        "android:textStyle": "bold",
        "android:textColor": "#222222",
        "android:layout_marginBottom": "4dp"
      }
    },
    {
      "type": "LinearLayout",
      "attributes": {
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:orientation": "vertical"
      },
      "children": [
        {
          "type": "LinearLayout",
          "attributes": {
            "android:layout_width": "match_parent",
            "android:layout_height": "wrap_content",
            "android:orientation": "horizontal",
            "android:padding": "8dp",
            "android:background": "@drawable/timetable_item_bg",
            "android:layout_marginBottom": "4dp"
          },
          "children": [
            {
              "type": "ImageView",
              "attributes": {
                "android:id": "@+id/ivExerciseIcon",
                "android:layout_width": "24dp",
                "android:layout_height": "24dp",
                "android:contentDescription": "Exercise Icon",
                "android:src": "https://img.icons8.com/ios-filled/24/000000/local-cafe.png"
              }
            },
            {
              "type": "TextView",
              "attributes": {
                "android:id": "@+id/tvExerciseTime",
                "android:layout_width": "wrap_content",
                "android:layout_height": "wrap_content",
```

```
                "android:text": "08:00 AM - 09:00 AM",
                "android:textSize": "14sp",
                "android:layout_marginStart": "8dp",
                "android:gravity": "center_vertical",
                "android:textColor": "#333333"
              }
            }
          ]
        },
        {
          "type": "LinearLayout",
          "attributes": {
            "android:layout_width": "match_parent",
            "android:layout_height": "wrap_content",
            "android:orientation": "horizontal",
            "android:padding": "8dp",
            "android:background": "@drawable/timetable_item_bg",
            "android:layout_marginBottom": "4dp"
          },
          "children": [
            {
              "type": "ImageView",
              "attributes": {
                "android:id": "@+id/ivExerciseIcon2",
                "android:layout_width": "24dp",
                "android:layout_height": "24dp",
                "android:contentDescription": "Exercise Icon",
                "android:src": "https://img.icons8.com/ios-filled/24/000000/bicycle.png"
              }
            },
            {
              "type": "TextView",
              "attributes": {
                "android:id": "@+id/tvExerciseTime2",
                "android:layout_width": "wrap_content",
                "android:layout_height": "wrap_content",
                "android:text": "10:00 AM - 11:00 AM",
                "android:textSize": "14sp",
                "android:layout_marginStart": "8dp",
                "android:gravity": "center_vertical",
                "android:textColor": "#333333"
              }
            }
          ]
        }
      ]
    }
  ]
},
{
  "type": "FrameLayout",
  "attributes": {
    "android:id": "@+id/fragmentContainer",
    "android:layout_width": "0dp",
    "android:layout_height": "0dp",
    "app:layout_constraintTop_toBottomOf": "@+id/llTimetable",
    "app:layout_constraintStart_toStartOf": "parent",
    "app:layout_constraintEnd_toEndOf": "parent",
    "app:layout_constraintBottom_toTopOf": "@+id/bitButton"
  }
},
{
  "type": "CardView",
  "attributes": {
    "android:id": "@+id/cardPlan",
    "android:layout_width": "match_parent",
    "android:layout_height": "wrap_content",
    "app:cardCornerRadius": "8dp",
    "app:cardElevation": "4dp",
    "app:layout_constraintTop_toBottomOf": "@+id/fragmentContainer",
    "app:layout_constraintStart_toStartOf": "parent",
    "app:layout_constraintEnd_toEndOf": "parent",
    "android:layout_marginTop": "8dp"
  },
  "children": [
    {
      "type": "FrameLayout",
      "attributes": {
        "android:layout_width": "match_parent",
        "android:layout_height": "match_parent"
```

```
          },
          "children": [
            {
              "type": "TextView",
              "attributes": {
                "android:id": "@+id/tvNoPlan",
                "android:layout_width": "wrap_content",
                "android:layout_height": "wrap_content",
                "android:text": "No␣workout␣plan␣added",
                "android:textSize": "16sp",
                "android:textColor": "#888888",
                "android:visibility": "visible",
                "android:layout_gravity": "center"
              }
            },
            {
              "type": "RecyclerView",
              "attributes": {
                "android:id": "@+id/rvPlans",
                "android:layout_width": "match_parent",
                "android:layout_height": "match_parent",
                "android:visibility": "gone"
              }
            }
          ]
        }
      ]
    },
    {
      "type": "com.google.android.material.floatingactionbutton.FloatingActionButton",
      "attributes": {
        "android:id": "@+id/fabAddPlan",
        "android:layout_width": "wrap_content",
        "android:layout_height": "wrap_content",
        "android:layout_margin": "16dp",
        "android:src": "https://img.icons8.com/ios-filled/24/000000/plus.png",
        "app:layout_constraintBottom_toBottomOf": "parent",
        "app:layout_constraintEnd_toEndOf": "parent"
      }
    }
  ]
},

"item_plan": {
  "type": "CardView",
  "attributes": {
    "android:layout_width": "match_parent",
    "android:layout_height": "wrap_content",
    "android:layout_marginBottom": "8dp",
    "app:cardCornerRadius": "8dp",
    "app:cardElevation": "2dp"
  },
  "children": [
    {
      "type": "LinearLayout",
      "attributes": {
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:orientation": "vertical",
        "android:padding": "8dp"
      },
      "children": [
        {
          "type": "LinearLayout",
          "attributes": {
            "android:layout_width": "match_parent",
            "android:layout_height": "wrap_content",
            "android:orientation": "horizontal",
            "android:layout_marginBottom": "8dp"
          },
          "children": [
            {
              "type": "TextView",
              "attributes": {
                "android:id": "@+id/tvExerciseName",
                "android:layout_width": "0dp",
                "android:layout_height": "wrap_content",
                "android:layout_weight": "1",
                "android:text": "Exercise␣Name",
                "android:textSize": "16sp",
```

```
            "android:textColor": "#222222"
          }
        },
        {
          "type": "Switch",
          "attributes": {
            "android:id": "@+id/cbCompleted",
            "android:layout_width": "wrap_content",
            "android:layout_height": "wrap_content",
            "android:contentDescription": "Mark␣as␣completed"
          }
        }
      ]
    },
    {
      "type": "TextView",
      "attributes": {
        "android:id": "@+id/tvDetails",
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:text": "Details␣about␣the␣workout",
        "android:textSize": "14sp",
        "android:textColor": "#555555"
      }
    },
    {
      "type": "LinearLayout",
      "attributes": {
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:orientation": "horizontal",
        "android:layout_marginTop": "8dp"
      },
      "children": [
        {
          "type": "Button",
          "attributes": {
            "android:id": "@+id/btnEdit",
            "android:layout_width": "wrap_content",
            "android:layout_height": "wrap_content",
            "android:text": "Edit"
          }
        },
        {
          "type": "Button",
          "attributes": {
            "android:id": "@+id/btnDelete",
            "android:layout_width": "wrap_content",
            "android:layout_height": "wrap_content",
            "android:layout_marginStart": "8dp",
            "android:text": "Delete"
          }
        }
      ]
    }
  ]
},

"dialog_add_plan": {
  "type": "ConstraintLayout",
  "attributes": {
    "android:layout_width": "match_parent",
    "android:layout_height": "wrap_content",
    "android:padding": "16dp"
  },
  "children": [
    {
      "type": "TextInputLayout",
      "attributes": {
        "android:id": "@+id/tilExerciseName",
        "android:layout_width": "0dp",
        "android:layout_height": "wrap_content",
        "android:hint": "Exercise␣Name",
        "app:layout_constraintTop_toTopOf": "parent",
        "app:layout_constraintStart_toStartOf": "parent",
        "app:layout_constraintEnd_toEndOf": "parent"
      },
      "children": [
```

```
      {
        "type": "TextInputEditText",
        "attributes": {
          "android:id": "@+id/etExerciseName",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:minHeight": "48dp",
          "android:paddingVertical": "12dp"
        }
      }
    ]
  },
  {
    "type": "TextInputLayout",
    "attributes": {
      "android:id": "@+id/tilTime",
      "android:layout_width": "0dp",
      "android:layout_height": "wrap_content",
      "android:hint": "Time",
      "app:layout_constraintTop_toBottomOf": "@id/tilExerciseName",
      "app:layout_constraintStart_toStartOf": "parent",
      "app:layout_constraintEnd_toEndOf": "parent",
      "android:layout_marginTop": "8dp"
    },
    "children": [
      {
        "type": "TextInputEditText",
        "attributes": {
          "android:id": "@+id/etTime",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:minHeight": "48dp",
          "android:paddingVertical": "12dp"
        }
      }
    ]
  },
  {
    "type": "TextInputLayout",
    "attributes": {
      "android:id": "@+id/tilDistance",
      "android:layout_width": "0dp",
      "android:layout_height": "wrap_content",
      "android:hint": "Distance",
      "app:layout_constraintTop_toBottomOf": "@id/tilTime",
      "app:layout_constraintStart_toStartOf": "parent",
      "app:layout_constraintEnd_toEndOf": "parent",
      "android:layout_marginTop": "8dp"
    },
    "children": [
      {
        "type": "TextInputEditText",
        "attributes": {
          "android:id": "@+id/etDistance",
          "android:layout_width": "match_parent",
          "android:layout_height": "wrap_content",
          "android:minHeight": "48dp",
          "android:paddingVertical": "12dp"
        }
      }
    ]
  },
  {
    "type": "TextInputLayout",
    "attributes": {
      "android:id": "@+id/tilSets",
      "android:layout_width": "0dp",
      "android:layout_height": "wrap_content",
      "android:hint": "Sets",
      "app:layout_constraintTop_toBottomOf": "@id/tilDistance",
      "app:layout_constraintStart_toStartOf": "parent",
      "app:layout_constraintEnd_toEndOf": "parent",
      "android:layout_marginTop": "8dp"
    },
    "children": [
      {
        "type": "TextInputEditText",
        "attributes": {
          "android:id": "@+id/etSets",
          "android:layout_width": "match_parent",
```

```
        "android:layout_height": "wrap_content",
        "android:minHeight": "48dp",
        "android:paddingVertical": "12dp"
      }
    }
  ]
},
{
  "type": "TextInputLayout",
  "attributes": {
    "android:id": "@+id/tilReps",
    "android:layout_width": "0dp",
    "android:layout_height": "wrap_content",
    "android:hint": "Reps",
    "app:layout_constraintTop_toBottomOf": "@id/tilSets",
    "app:layout_constraintStart_toStartOf": "parent",
    "app:layout_constraintEnd_toEndOf": "parent",
    "android:layout_marginTop": "8dp"
  },
  "children": [
    {
      "type": "TextInputEditText",
      "attributes": {
        "android:id": "@+id/etReps",
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:minHeight": "48dp",
        "android:paddingVertical": "12dp"
      }
    }
  ]
},
{
  "type": "TextInputLayout",
  "attributes": {
    "android:id": "@+id/tilWeight",
    "android:layout_width": "0dp",
    "android:layout_height": "wrap_content",
    "android:hint": "Weight",
    "app:layout_constraintTop_toBottomOf": "@id/tilReps",
    "app:layout_constraintStart_toStartOf": "parent",
    "app:layout_constraintEnd_toEndOf": "parent",
    "android:layout_marginTop": "8dp"
  },
  "children": [
    {
      "type": "TextInputEditText",
      "attributes": {
        "android:id": "@+id/etWeight",
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:minHeight": "48dp",
        "android:paddingVertical": "12dp"
      }
    }
  ]
},
{
  "type": "TextInputLayout",
  "attributes": {
    "android:id": "@+id/tilNotes",
    "android:layout_width": "0dp",
    "android:layout_height": "wrap_content",
    "android:hint": "Notes",
    "app:layout_constraintTop_toBottomOf": "@id/tilWeight",
    "app:layout_constraintStart_toStartOf": "parent",
    "app:layout_constraintEnd_toEndOf": "parent",
    "android:layout_marginTop": "8dp"
  },
  "children": [
    {
      "type": "TextInputEditText",
      "attributes": {
        "android:id": "@+id/etNotes",
        "android:layout_width": "match_parent",
        "android:layout_height": "wrap_content",
        "android:minHeight": "48dp",
        "android:paddingVertical": "12dp"
      }
    }
```

```
        ]
      }
    ]
  },

  "resources": {
    "drawables": {
      "bg_gradient": {
        "shape": "layer",
        "content": [
          {
            "type": "shape",
            "attributes": {
              "android:shape": "rectangle"
            },
            "children": [
              {
                "type": "gradient",
                "attributes": {
                  "android:startColor": "#FFDEE9",
                  "android:endColor": "#B5FFFC",
                  "android:angle": "45"
                }
              }
            ]
          }
        ]
      },
      "round_button_bg_scheme1": {
        "shape": "oval",
        "attributes": {
          "android:shape": "oval",
          "android:color": "#FF5722"
        }
      },
      "round_button_bg_scheme2": {
        "shape": "oval",
        "attributes": {
          "android:shape": "oval",
          "android:color": "#4CAF50"
        }
      },
      "round_button_bg_scheme3": {
        "shape": "oval",
        "attributes": {
          "android:shape": "oval",
          "android:color": "#2196F3"
        }
      },
      "timetable_item_bg": {
        "shape": "rectangle",
        "attributes": {
          "android:shape": "rectangle",
          "android:cornerRadius": "4dp",
          "android:color": "#FFFFFF"
        }
      }
    }
  }
}
```

Listing A.13: AI-generated UI JSON specification

The Rendered UI