A

PROJECT REPORT ON

AI COMPANION

SUBMITTED BY

ANJALI SUTHAR

UNDER THE GUIDENCE OF

PROFESSOR : NIYATI KALYANPUR

EXAMINATION BSC.IT, SEMESTER V

DEPARTMENT OF INFORMATION TECHNOLOGY

GHANSHYAMDAS SARAF COLLEGE OF ARTS AND COMMERCE

(Affiliated to University of Mumbai)

MUMBAI 400064

MAHARASHTRA

2023-2024

PROFORMA FOR THE APPROVAL PROJECT PROPOSAL

(Note:All entries of the proforma of approval should be filled up with appropriate and complete information. Incomplete proforma of approval in any respect will be summarily rejected.)

PNR **No.: ……………………**　　　　　　　　　　Roll no**:**

1 Name of the Student

2. Title of the Project

3. Name of the Guide

4. Teaching experience of the Guide

5. Is this your first submission?　　Yes ☐　　No ☐

Signature of the Student　　　　　　　　　　Signature of the Guide

Date: …………………　　　　　　　　　　Date: …………………….

Signature of the Coordinator

DEPARTMENT OF INFORMATION TECHNOLOGY
GHANSHYAMDAS SARAFCOLLEGE OF ARTS AND COMMERCE

(Affiliated to university of Mumbai)

CERTIFICATE

This is to certify that the project entitled, **" AI COMPANION "**, is bonafied work of **ANJALI SUTHAR** bearing Seat No: submitted in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY** from
University of Mumbai.

Internal Guide Coordinator

External Examiner

Date: College Seal

DECLARATION

I hereby declare that the project entitled, "**AI COMPANION**" done at Ghanshyamdas Saraf College of Arts and Commerce, has not been in any case duplicated to submit to any other universities for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as final semester project as part of my curriculum

ANJALI SUTHAR

ABSTRACT

This project presents the development of a SaaS platform, "AI Companion Chat," designed to allow users to create and interact with AI companions modeled after celebrities, historical figures, or custom personas. The application leverages advanced AI technologies to enable dynamic conversations, providing users with entertaining and informative experiences. Core features include AI companion creation and customization, interactive chat interfaces, secure user authentication, and flexible subscription management.

The platform is built on a modern technical stack, utilizing Next.js and React.js for the frontend, MySQL for database management, and Prisma for data operations. Clerk is employed for user authentication, ensuring secure and seamless access, while Stripe handles all payment and subscription transactions. The application's infrastructure is hosted on Vercel, providing scalable and efficient serverless deployment.

This project emphasizes user engagement through personalization, with options to customize AI companions' appearance, voice, and personality. It also incorporates a rich knowledge base, allowing AI companions to provide relevant responses and insights. The design includes robust data security measures, ensuring user data privacy and transaction safety.

Additionally, the platform is designed with scalability in mind, supporting high user loads and frequent updates through a continuous integration and delivery (CI/CD) pipeline. This abstract outlines the key objectives and features of the AI Companion Chat platform, highlighting its innovative approach to delivering an interactive AI-driven experience for users.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to everyone who contributed to the successful development of the "AI Companion Chat" project. This endeavor would not have been possible without the guidance, support, and encouragement from a number of individuals.

Firstly, we extend our deepest thanks to our mentors and instructors, whose invaluable advice and feedback helped shape this project from concept to completion. Your expertise and dedication have been instrumental in guiding us through the various stages of development.

We also wish to acknowledge the continuous support of our colleagues and team members. Your collaboration, creativity, and technical skills played a crucial role in overcoming the challenges encountered during the project.

Finally, we are grateful to the developers and communities behind the technologies we utilized, including Next.js, React.js, Prisma, MySQL, Clerk, and Stripe. Your open-source contributions and detailed documentation enabled us to build a robust and scalable application.

Thank you to everyone who believed in this project and contributed in any way, making it a rewarding and enriching experience.

TABLE CONTENT

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

1.1 Background

In recent years, the rise of AI technologies has transformed the way people interact with digital systems. Conversational AI, driven by advancements in natural language processing (NLP), has become a key area of development, allowing users to engage in more natural, human-like conversations with virtual assistants and chatbots. This has opened new possibilities in areas like customer service, education, entertainment, and personalized experiences. Users today seek interactive platforms where they can learn, be entertained, or simply engage in fun conversations with AI companions.

"AI Companion Chat" leverages this growing trend by offering a unique platform where users can interact with AI companions modeled after celebrities, historical figures, or fictional characters. By combining modern technologies like Next.js, MySQL, and advanced AI models, the platform provides a seamless and engaging experience. Users can customize their companions' personalities and appearances, making the interaction even more personalized and enjoyable. With secure user authentication, subscription management, and a knowledge base integrated into the AI companions, this platform aims to offer a secure, scalable, and user-friendly service that keeps up with the evolving needs of today's tech-savvy users.

1.2 Objectives

1. Enhancing User Engagement:

- Reason: To keep users interested and retain them long-term.
- Approach: Personalize AI companions with customizable settings and deliver dynamic,entertaining interactions through advanced NLP technology.

2. Providing Information and Entertainment:

- Reason: To combine learning with fun, enhancing the platform's appeal.
- Approach: Equip AI companions with rich knowledge bases and regularly update content toensure engaging and current interactions.

3. Facilitating Secure and Seamless User Experience:

- Reason: To build trust and ensure smooth platform usage.
- Approach: Implement secure authentication with Clerk, streamline payments via Stripe,and offer intuitive account management features.

4. Supporting Scalability and Reliability:

- Reason: To manage increasing user demand and maintain consistent performance.
- Approach: Utilize Vercel for scalable serverless hosting, employ robust backup anddisaster recovery strategies, and integrate CI/CD for efficient updates.

1.3 Purpose and Scope

1.3.1 Purpose

Purpose Statement:

- The purpose of this SaaS application is to offer users a unique and engaging platformwhere they can create, customize, and interact with AI companions. These companions, modeled after celebrities, historical figures, or fictional characters, aim

  to provide both entertainment and educational value through dynamic conversationsand personalized interactions.

Benefits:

- Provides users with a personalized and entertaining digital experience.
- Enhances user engagement through interactive and informative conversations.
- Offers educational value by integrating rich knowledge bases about various personas.

1.3.2 Scope

Features Covered:

- AI Companion Creation: Selection, modeling, and personalization of AI companions.
- Interactive Chat: Conversational capabilities, dynamic interactions, and knowledgesharing.
- Content Management: Integration and updating of knowledge bases. o UserManagement: Authentication, profile management, and chat history.
- Payment and Subscription: Handling payments, subscriptions, and in-app purchases.

Limitations:

- The scope may not include integration with external AI systems or platforms outsideof the defined tech stack.
- Customization options may be limited to predefined settings and features.
- The platform may not cover all historical figures or celebrities due to

  licensing orcontent availability constraints.

1.4 Applicability

Target Audience:

- **Individual Users:** Those seeking interactive and personalized digital experiences.
- **Fans of Personas:** Individuals interested in engaging with AI companions modeled aftercelebrities, historical figures, or fictional characters.
- **Educational Seekers:** Users looking for digital formats that combine learning withentertainment.

Use Cases:

- **Personal Entertainment:** Utilizing AI companions for recreational and leisurely interactions.
- **Educational Engagement:** Facilitating learning about notable figures or characters throughconversational AI.
- **Business Applications:** Enhancing customer experiences by integrating AI companions intoorganizational platforms or services.

Deployment Context:

- **User Base:** Applicable to both individual consumers and organizations seeking to integrate AIcompanion technology.
- **Platform Accessibility:** Available on web and mobile platform ,designed for userfriendlinessand high availability to ensure optimal performance and accessibility

1.5 Organization of Report

The subsequent chapters of the project report will focus on the survey of different technologies which includes comparison of different platforms, languages, front- end, back-end, datasets, algorithms, etc. the survey is followed by the analysis of requirements resulting in generation of requirement specification and schedule of activities. It also includes the conceptual design that visualizes the features and operations that can be performed on the system. Final chapter of the project report includes the system design that describes desired features and operations in detail, including screen layouts, business rules, process diagrams, pseudo code and other documentation

CHAPTER 2: SURVEY OF TECHNOLOGY

2.1 SOFTWARE

Choosing between Visual Studio Code (VS Code) and Visual Studio often depends on the specific needs of the development project and the preferences of the developers. Here’s why Visual StudioCode is commonly chosen over Visual Studio, especially for modern web development and other scenarios:

1. Lightweight and Fast

- **Performance**:
  - **VS Code**: Known for its lightweight nature, VS Code starts up quickly and runs smoothly even on lower-end hardware. Its performance is optimized for a widerange of tasks without being resource-intensive.
  - **Visual Studio**: In contrast, Visual Studio is a full-featured IDE that can be heavy andslower to start, particularly when dealing with large projects or when loaded with numerous extensions and features.

2. Cross-Platform Support

- **Operating System Compatibility**:
  - **VS Code**: Available on Windows, macOS, and Linux, making it a versatile choice for development across different platforms. This cross-platform capability is essential forteams working in diverse environments.
  - **Visual Studio**: Traditionally, Visual Studio has been more focused on Windows. Although there are versions like Visual Studio for Mac, they do not offer the samebreadth of cross-platform support as VS Code.

3. Customizability and Extensibility

- **Extensions and Customization**:
  - **VS Code**: Offers a vast library of extensions via its marketplace, allowing developersto tailor the editor to their specific needs. Extensions cover everything from language support to debugging tools and integrations with version control systems.
  - **Visual Studio**: While also extensible, Visual Studio can be more cumbersome to customize. It is often considered more monolithic in its approach, and its extensionecosystem is not as diverse as VS Code’s.

4. Simplicity and Focus

- **Editor Focus**:
  - **VS Code**: Designed as a code editor rather than a full IDE. This simplicity allowsdevelopers to focus on coding without being overwhelmed by a myriad of
    built-in features. It’s particularly well-suited for quick edits, scripting, and webdevelopment.
  - **Visual Studio**: As a full-featured IDE, Visual Studio provides extensive builtin tools and features, which can sometimes be overwhelming for developers whoonly need a simple code editor.

5. Modern Development Workflows

- **Integration with Modern Tools**:
  - **VS Code**: Seamlessly integrates with modern development tools and workflows,such as GitHub, Docker, and various CI/CD pipelines. It is often preferred for web development, Node.js projects, and other scenarios where integration with cutting- edge tools is crucial.
  - **Visual Studio**: While it has strong integration with Microsoft's ecosystem (e.g., Azure, .NET), it may not always be the best fit for non-Microsoft technologies or modern development practices outside of its primary focus.

6. Version Control Integration

- **Git and Source Control**:
  - **VS Code**: Comes with built-in Git support and has a streamlined interface for versioncontrol. It supports various source control systems through extensions, providing a smooth experience for managing code changes. o **Visual Studio**: Also supports Git and other version control systems, but its interface can be more complex, and the experience may not be as seamless as VS Code's.

7. Cost and Licensing

- **Cost Considerations**:
  - **VS Code**: Free and open-source, making it an attractive option for individualdevelopers and small teams. Its cost-effectiveness is a significant advantage.
  - **Visual Studio**: Offers a free Community edition, but advanced features and professional editions require paid licenses. For large teams or enterprises, this canadd significant cost.

8. Community and Ecosystem

- **Community Support**:
  - **VS Code**: Has a large and active community contributing to its ecosystem, with frequent updates and a wide range of third-party extensions. Its popularity meansthat developers can easily find support, tutorials, and resources.
  - **Visual Studio**: Although it also has a strong community and extensive resources, thefocus is often more centered around enterprise and .NET-related development.

Summary

Visual Studio Code is often chosen over Visual Studio for its lightweight nature, cross-platform support, and flexibility in customizing the development environment. It's particularly well-suited for modern, diverse development needs, including web development and quick coding tasks. Visual Studio, with its full-featured IDE capabilities, remains a strong choice for larger, enterprise-level projects, particularly those deeply integrated with Microsoft technologies.

2.2 FRONT-END TECHNOLOGY

Aspect	React.js and Next.js	Angular
Core Technology	JavaScript library (React.js) withSSR/SSG framework (Next.js)	Full-featured JavaScript framework
Component-Based Architecture	Yes, React's modular componentmodel is highly reusable	Yes, Angular uses components, butwith more complexity
Dynamic User Interfaces	Efficient real-time updates withReact's virtual DOM; Next.js supports server-side rendering for improved performance	Two-way data binding for real-timeupdates; more complex implementation
State Management	React provides hooks, contextAPI, and integrations with libraries like Redux for state management	Built-in with RxJS and NgRx; morecomplex and requires a learning curve
Ecosystem	Extensive libraries and tools for customization and enhancement;strong community support	Comprehensive ecosystem with Angular CLI, RxJS, and many built-infeatures
Server-Side Rendering (SSR)	Built-in support in Next.js; allowsfor improved SEO and performance	Achieved with Angular Universal;setup is more complex and less integrated
Static Site Generation(SSG)	Built-in support in Next.js for pre-building pages at build time	Not natively supported; complexsetup needed

API Routes	Next.js includes built-in API routes for backend functionality	Not built-in; requires separate setupfor backend functionality
Automatic Code Splitting	Automatic in Next.js; loads onlynecessary JavaScript for each page	Manual setup using Angular CLI forcode splitting
Optimize d Routing	File-based routing in Next.js;simple and automatic	Built-in routing but can be complex tomanage
Learning Curve	Moderate; React's flexibility requires some learning; Next.jsis more approachable	Steeper; Angular's comprehensivenature has a higher learning curve
Developme ntSpeed	High; React's componentreusability and Next.js's built-in features accelerate development	Moderate to high; Angular's featurescan speed up development but require a steeper learning curve

Summary

- React.js and Next.js offer a flexible, component-based architecture with efficient real-time updatesand built-in support for server-side rendering (SSR) and static site generation (SSG). The ecosystem around React is rich with libraries and tools, which facilitates rapid development and customization. -Angular provides a comprehensive framework with built-in features for state management and routing but has a steeper learning curve due to its complexity. While Angular supports server-side rendering via Angular Universal, the setup can be more complex compared to Next.js's built-in capabilities.

React.js and Next.js are chosen for their flexibility, ease of integration, and performance benefits,making them well-suited for applications requiring dynamic interactions and SEO optimization.
Angular is robust and feature-rich but may be less ideal for projects needing quick developmentcycles and simpler setups.

2.3 BACK-END TECHNOLOGY

Aspect	**MySQL**	**Prisma**
Type	Relational Database Management System (RDBMS)	Object-Relational Mapping (ORM) Tool
Data Storage	Structured, relational data storage	Abstracts relational data storage into a type-safe API
ACID Compliance	Yes, adheres to ACID principles for transaction reliability	N/A (depends on underlying database)
Scalability	High, handles large amounts of data and high transaction volumes	High, depending on underlying database (MySQL here)
Performance	High performance for complex queries and large datasets	High performance, facilitated by type-safe queries and efficient schema management
Type Safety	Lower, relies on SQL queries with potential runtime errors	High, provides type-safe queries based on schema definition
Ease of Use	Moderate, requires SQL knowledge and management tools	High, offers a declarative schema and type-safe queries
Schema Management	Requires manual migration scripts and schema updates	Automated migrations and declarative schema management

Development Speed	Moderate, with manual schema management and query writing	High, with automated migrations and an intuitive query API
Integration with Node.js	Seamless integration, widely supported in the ecosystem	Excellent integration, with a type-safe API for managing MySQL
Community Support	Extensive, mature community and ecosystem	Growing, with strong support for modern JavaScript and TypeScript stacks
Error Handling	Relies on SQL for error handling, which can be less descriptive	High, with type-safe queries that reduce runtime errors

Summary

- **MySQL** is used for its robust performance, ACID compliance, and scalability, making it well-suited for managing structured data and handling high transaction volumes. It provides reliable and secure data storage and retrieval, which is crucial for applications that require data integrity and performance.
- **Prisma** is chosen for its type-safe query capabilities, declarative schema management, and automated migrations. It simplifies database interactions by providing a highlevel API that reduces the likelihood of runtime errors and streamlines development. This integration enhances productivity and efficiency by abstracting complex database operations into manageable and type-safe queries.

Together, **MySQL** and **Prisma** form a powerful backend stack that ensures reliable data management, high performance, and a streamlined development process, catering to the needs of the AI Companion Chat application.

2.4 ALGORITHM

1. **Natural Language Processing (NLP) Algorithms**
   - Components include tokenization, named entity recognition (NER), part-of-speech tagging, intent recognition, language generation, and context management. These techniques work together to break down user input, identify entities like names or dates, understand the intent behind queries, and maintain the flow ofconversation.
   - Usage: NLP algorithms enable the AI companion to understand and interpret user messages accurately,generating meaningful and contextually relevant responses.
   - Reason for Use: They are crucial for creating human-like conversations, allowing the AI to engage with usersnaturally and respond appropriately to different types of queries.

2. **Machine Learning (ML) Algorithms**
   - This includes supervised learning, where the AI is trained using labeled data to predict outcomes; unsupervisedlearning, which identifies hidden patterns in data; and reinforcement learning, where the AI learns from user feedback to refine its interactions.
   - Usage: Machine learning algorithms enhance the AI companion's ability to learn from interactions over time,improving its accuracy in understanding and generating responses.
   - Reason for Use: These algorithms help the AI evolve by adapting to user behavior, making interactions morepersonalized and intelligent as it gathers more data.

3. **Chatbot-Specific Algorithms**
   - Consists of rule-based chatbots that follow predefined conversation rules, retrieval-based chatbots that searcha database of responses for the best match, and hybrid chatbots that combine both approaches with machine learning for more flexibility.
   - Usage: These algorithms ensure that the AI companion can handle both straightforward conversations usingrules and more complex interactions using dynamic learning models.
   - Reason for Use: By combining these methods, the AI can quickly provide accurate responses for simplequeries while also generating more nuanced replies for complex interactions, resulting in a balanced conversational experience

CHAPTER 3 REQUIREMENT AND ANALYSIS

3.1 Problem Definition

The primary challenge is creating a seamless and engaging AI Companion platform where users cancreate, interact with, and customize AI personas modeled after celebrities, historical figures, or

fictional characters. The system must support secure, scalable, and personalized interactions whileproviding real-time conversational capabilities.

Common Issues with Existing Systems:

- Limited Personalization: Existing platforms often have insufficient customization, reducinguser satisfaction.
- Slow Response Times: Many applications suffer from latency due to suboptimal back-endperformance.
- Inconsistent Personalities : AI personas fail to maintain consistent behaviour or acrossconversations, breaking immersion.
- Security Concerns : Insufficient authentication measures, leading to user data breaches orleaks in chat history.

3.2 Requirement Specification

3.2.1 Functional Requirements

Functional requirements are a set of specifications that outline the functions a system or component must perform. They're like a checklist for a system and are essential for developers to implement sothat users can accomplish their tasks

1. AI Companion Creation : Users can choose from predefined templates like celebrities or historicalfigures and personalize the appearance, voice, and personality. The backend will store these configurations and ensure the AI responds accordingly based on these personalized attributes.
   - Example : A user can choose to chat with a historical figure like Napoleon Bonaparte, tweak hispersonality to be more humorous, and give him a modern voice.

2. Dynamic Conversations: Users can interact with their companions via text (and possibly voice in future expansions). The AI uses NLP (Natural Language Processing) to analyze user queries and respond appropriately.
   - Example : If a user asks, “Tell me something about physics,” and their companion is modeled afterAlbert Einstein, the system will access preloaded knowledge or external resources to respond accurately.

3. User Profile Management : Users should have control over their profiles, view their chat history,manage subscription preferences, and customize their companion's appearance.

4. Payment Handling : Through Stripe integration, the platform must support payment options likecredit cards, subscription plans, and one-time payments for premium features or advanced AI companion customization

Secure Authentication: Using Clerk , secure user sessions, registration, and login will be managed withsupport for OAuth or third-party logins like Google or Facebook.

3.2.2 Non-Functional Requirements

Non-functional requirements define how the system performs certain functions rather than what itdoes.

- **Performance** : The platform should be able to support at least 1,000 concurrentusers initially, with sub-500 ms latency per user interaction.

- **Scalability**: Built using Vercel and cloud-hosted MySQL , the system should dynamically scale to accommodate growing users.

- **Security** : Secure user data (chats, profiles, payment info) using encryption (AES-256 for data-at-rest, HTTPS/SSL for data-in-transit).

- **Usability** : Ensure a responsive, easy-to-navigate UI so users can create and interact with companions without technical difficulties.

- **Availability**: Target a 99.9% uptime using high-availability infrastructure, backup databases, and redundant server setups.

- **Maintainability** : The codebase should be modular and use industry-standardpatterns to allow future updates, like adding new AI companions, without disruptingexisting functionality.

3.3 Planning and Scheduling

3.3.1 Milestones

Task Name	**Duration**	**Start Date**	**End Date**
Selection of the project	**1 weeks**	**01-07-24**	**07-07-24**
Gathering of Requirement	**2 weeks**	**08-07-24**	**21-07-24**
Requirement Analysis	**5 weeks**	**22-07-24**	**11-08-24**
System Analysis	**2 weeks**	**12-08-24**	**25-08-24**

Documentation	**3 weeks**	**26-08-24**	**15-09-24**
Designing	**4 weeks**	**16-09-24**	**22-09-24**
Coding/Implementation	**16 weeks**	**23-09-24**	**12-01-25**
Testing	**5 weeks**	**13-01-25**	**15-02-25**

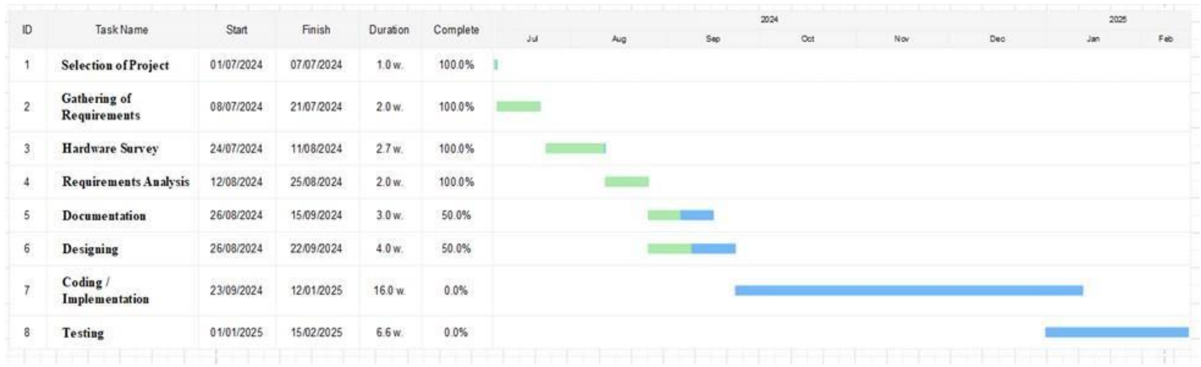
3.3.2 Number of Person

The project consist only one person- Anjali Suthar Schedule Start Day : 01/07/2024

Schedule End Day : 15/02/2025

Gantt Chart

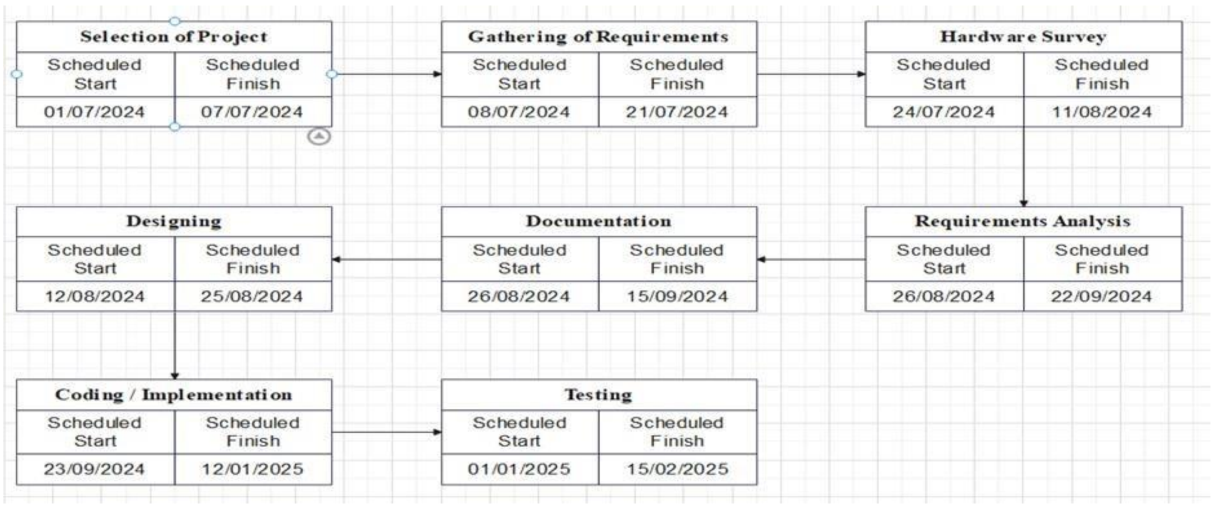
A Gantt Chart is a visual project plan that lists tasks and milestones on the vertical axis with time plotted on the horizontal axis. Gantt Charts are used in project management to schedule, track, andcommunicate deliverables, deadlines, dependencies, and resource assignments.



ID	Task Name	Start	Finish	Duration	Complete
1	Selection of Project	01/07/2024	07/07/2024	1.0 w.	100.0%
2	Gathering of Requirements	08/07/2024	21/07/2024	2.0 w.	100.0%
3	Hardware Survey	24/07/2024	11/08/2024	2.7 w.	100.0%
4	Requirements Analysis	12/08/2024	25/08/2024	2.0 w.	100.0%
5	Documentation	26/08/2024	15/09/2024	3.0 w.	50.0%
6	Designing	26/08/2024	22/09/2024	4.0 w.	50.0%
7	Coding / Implementation	23/09/2024	12/01/2025	16.0 w.	0.0%
8	Testing	01/01/2025	15/02/2025	6.6 w.	0.0%

Pert Chart

A PERT chart is a visual project management tool used to map out and track the tasks andtimelies. The name PERT is an acronym for Project (or Program) Evaluation and Review Technique.PERT charts are similar to Gantt charts in that they offer a graphical view of a project's tasks, schedule, and timelines.



3.4 Software and Hardware Requirements

3.4.1 Software Requirements

Designing : Figma or Adobe XD for UI/UX design.

Development :

Languages : JavaScript/TypeScript

Frameworks : Next.js (for front-end), Prisma (ORM for MySQL), React.js

Libraries : Clerk (for authentication), Stripe (for payments)

Documentation : GitHub or Confluence for documenting API references, user guides, and architecture.

Operational : Vercel for hosting, MySQL for database, CI/CD pipeline with GitHub Actions.

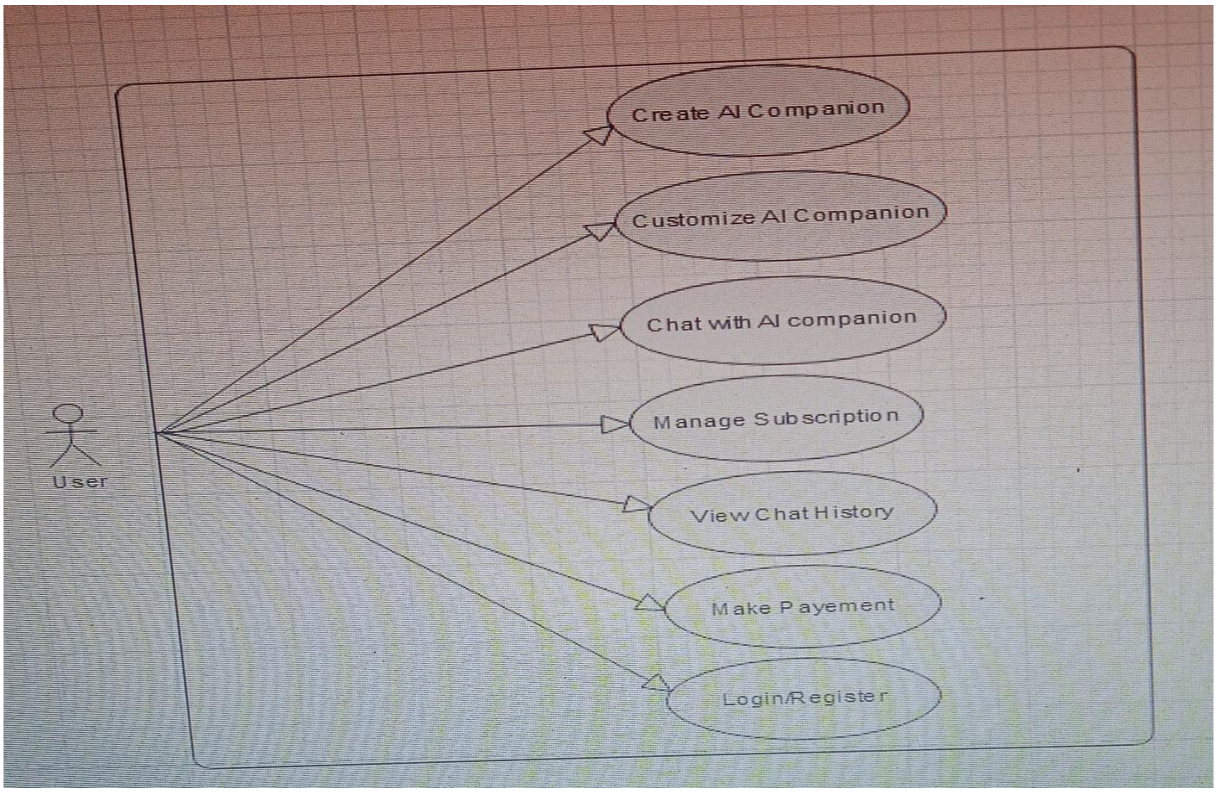
3.4.2 Hardware Requirements

- **Development Machines :**
  - **RAM**: Minimum 16GB
  - **SSD**: Minimum 512GB

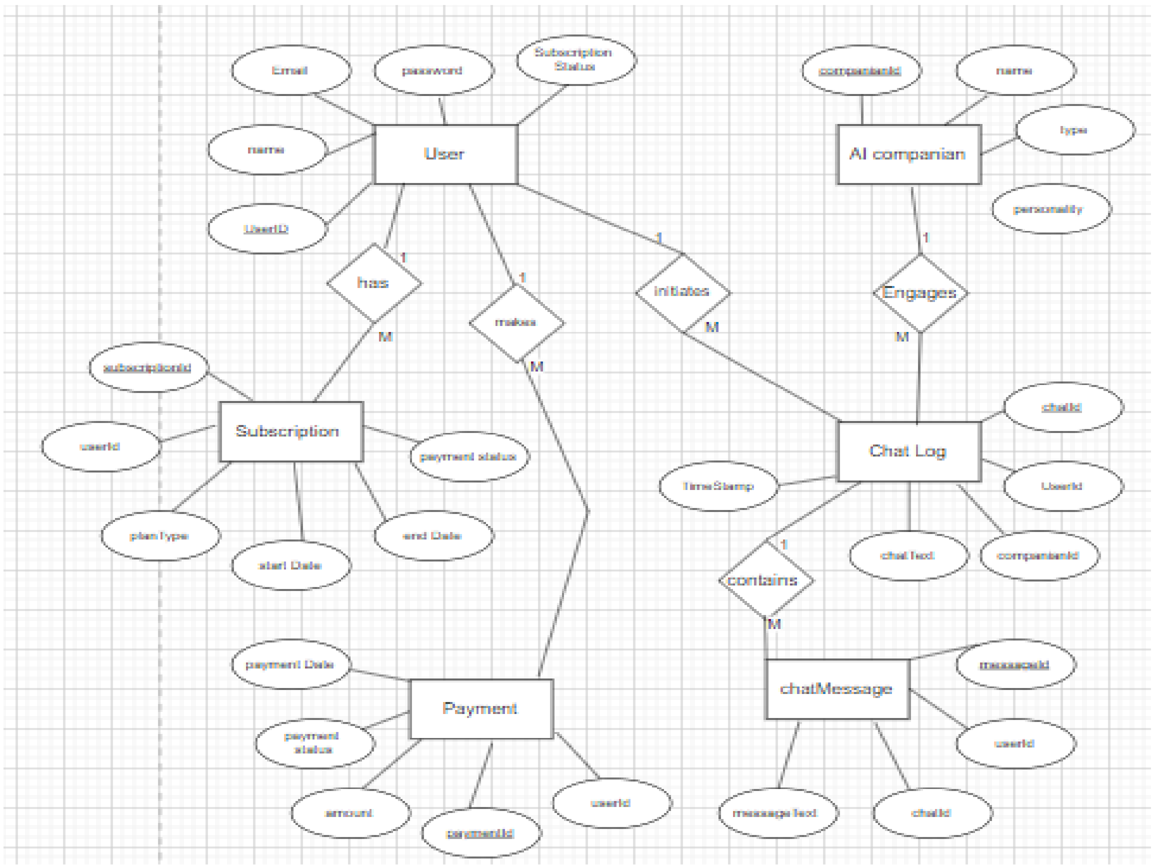
- **Server Requirements** :
  - **RAM**: 32GB+ (depending on the scale)
  - **SSD:** 1TB+ (for fast database and chat log access)

3.5 Conceptual Models

3.5.1 Use Case Diagram

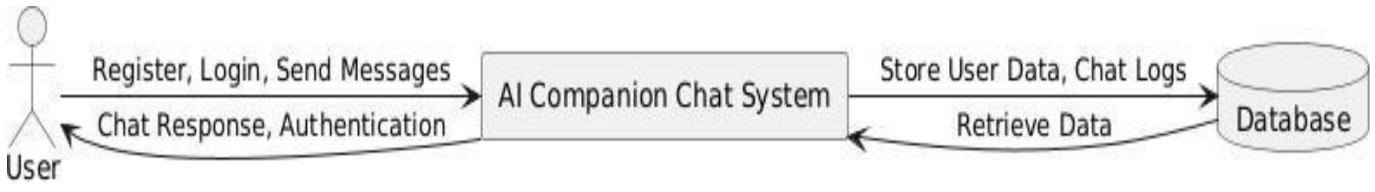


3.5.2 Entity Relationship Diagram

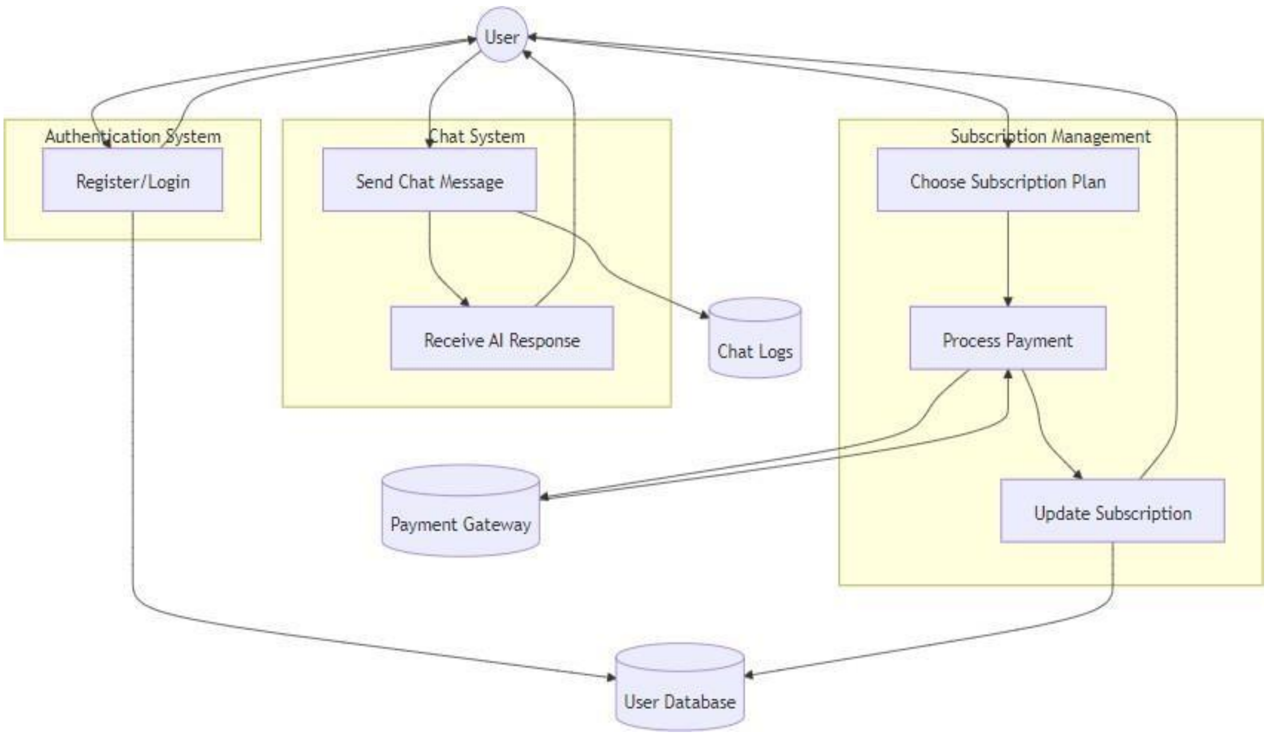


3.5.1 DFD (Data Flow Diagram)

- Level 0 DFD : Shows basic flows like user requests, server responses, and data updates.

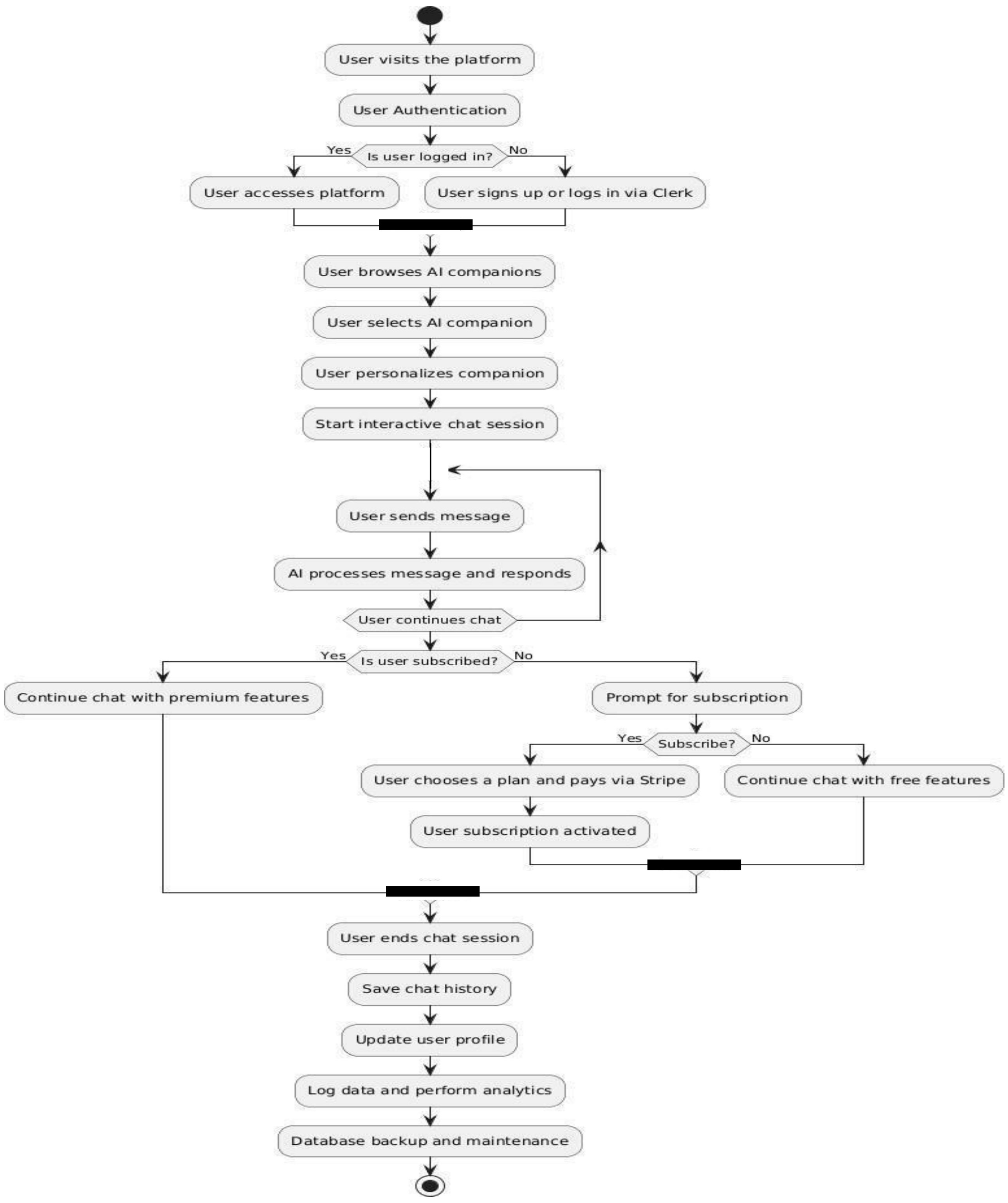

User
Register, Login, Send Messages
Chat Response, Authentication
AI Companion Chat System
Store User Data, Chat Logs
Retrieve Data
Database

- Level 1 DFD : Detailed breakdown showing how AI Companion creation, chat interactions, and profilemanagement interact with databases and external services.



3.5.3 Activity Diagram:

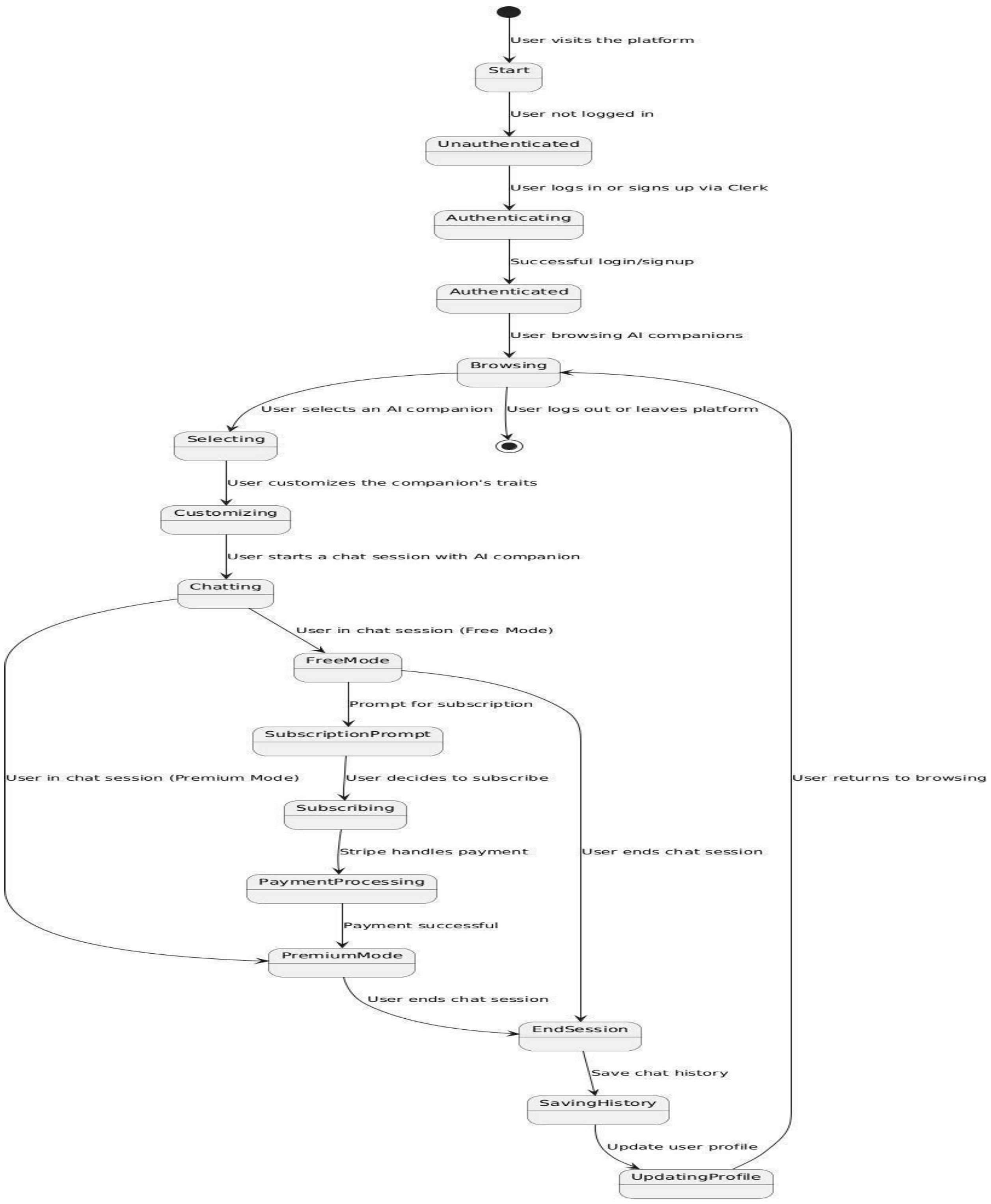
Depicts how users interact with the system to create companions, customize them, and initiate chat


User visits the platform
User Authentication
Is user logged in?
Yes
No
User accesses platform
User signs up or logs in via Clerk
User browses AI companions
User selects AI companion
User personalizes companion
Start interactive chat session
User sends message
AI processes message and responds
User continues chat
Is user subscribed?
Yes
No
Continue chat with premium features
Prompt for subscription
Subscribe?
Yes
No
User chooses a plan and pays via Stripe
Continue chat with free features
User subscription activated
User ends chat session
Save chat history
Update user profile
Log data and perform analytics
Database backup and maintenance

.

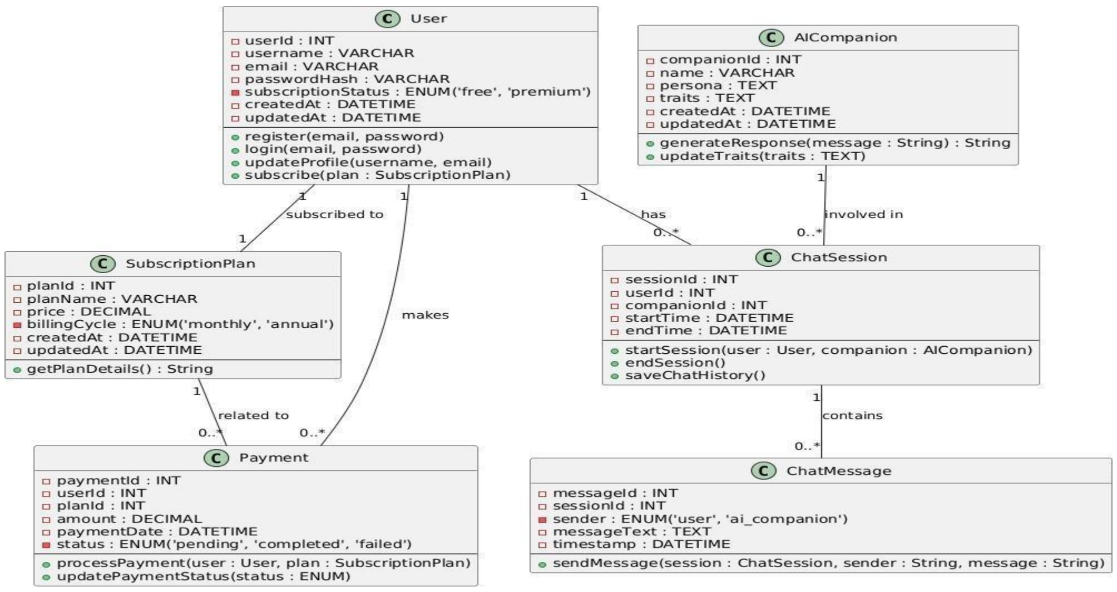
3.5.4 State Transition Diagram

- Shows the transition states of AI companions (e.g., creation, customization, active interaction, idle, etc.).



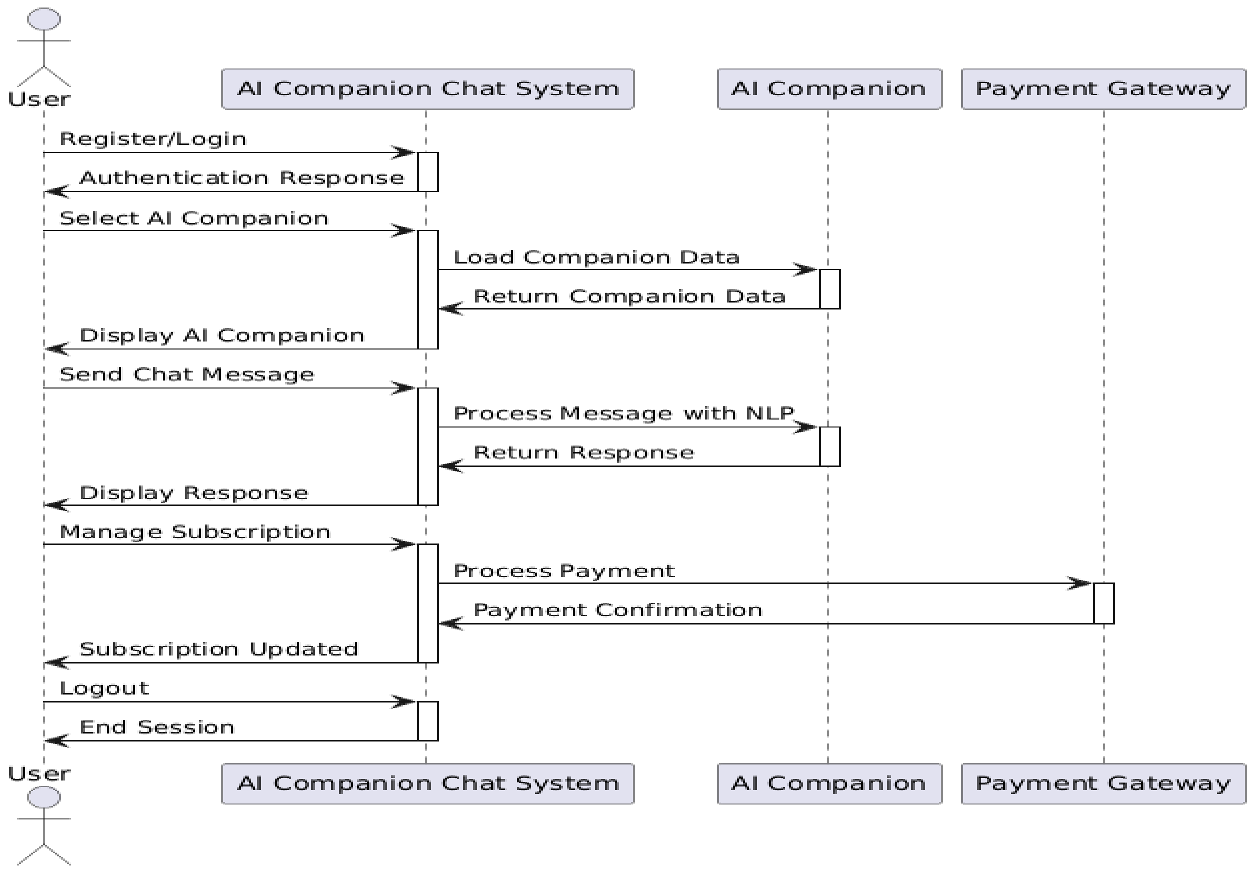
3.5.4 Class Diagram

- Depicts classes for User, Companion, Subscription, Payment, and their relationships.



3.5.7 Sequence Diagram

- Demonstrates interaction between users, AI services, payment systems (Stripe), and databases during a Complete user session



3.5.1 Event Table

- Maps out events like user login, AI companion creation, conversation start, and payment transactions with triggers andresponses.

Event	Trigger	Source	Activity	Response
User Registration	User submits registration form	User	Validate user inputand create a new account	Confirmation message sent and User account created
User Login	User submits Login details	User	Validate credentialsand authenticate the user	User is logged in and redirected to thedashboard
AI Companion Selection	User selects an AI companion	User	Load available AI companions and display them	Selected AI companion is displayed tothe user
Send Chat Message	User sends a Chatmessage	User	Process the message using NLP,retrieve response from AI	AI response is displayed to the user
View Chat History	User requests Chat history	User	Retrieve chat logsfrom database	Chat history is displayed on the screen

Manage Subscriptio n	User requests subscription change	User	Process subscription request and payment, update database	Subscription details updated andconfirmation sent to user
Process Paymen t	User submits payment details	User	Validate payment details and processthe transaction	Payment processed successfully,confirmation sent to user
Subscriptio nRenewal Reminder	Subscription nearing expiry	Syste m	Check subscription status and trigger renewal reminder	Reminder notification sent to the user
Logout	User clicks logout	User	End user session and clear session data	User is logged out and redirected to thelogin page
Password Reset	User requests password reset	User	Validate user’s email, send password reset link	Password reset link sent to user's email
Download Chat History	User clicks Download button	User	Convert chat logs into PDF or CSV format	Chat History Download In selected Format

Invalid Login Attempt	User enters incorrect credentials	User	Check login credentials againststored data	Error message displayed for invalid loginattempt
Receive Notificatio n	System sends response or alert	Syste m	Generate and sendnotification (chat response, subscription alert)	Notification sent via email or systemalert
Companio nUpdate	System updates AIcompanion data	Syste m	Update the AI companion's knowledge base orresponses	Updated companion data available foruse

CHAPTER 4: SYSTEM DESIGN

4.1 Basic Modules

1. **User Registration and Authentication**:
   •Users can register, log in, and manage their accounts securely through Clerk integration.
2. **AI Companion Creation and Customization**:
   •Users select AI companions modeled after celebrities, historical figures, or fictional characters.
   •Options to personalize the companion’s appearance, voice, and personality traits.
3. **Interactive Chat System**:
   •Users can engage in dynamic conversations with their AI companions using advanced NLP.
   •AI companions provide real-time responses based on a rich knowledge base.
4. **Payment and Subscription Management**:
   •Integrated with Stripe to manage payments, including subscriptions and inapp purchases.
   •Supports various payment models like free trials, monthly subscriptions, and pay-per-use.
5. **User Profile and History Management**:
   •Users can view and manage their chat history, subscription status, and AI companion settings.

4.2 Data Design

4.2.1 Schema Design

- **User Table**:

Column Name	Data Type	Description
userID	INT	Unique identifier for the user
name	VARCHAR	Name of the user
email	VARCHAR	User's email address
password	VARCHAR	Encrypted password
subscriptionStatus	VARCHAR	User's subscription plan status

- **Companion Table**:

Column Name	Data Type	Description
companionID	INT	Unique identifier for the companion
companionType	VARCHAR	Type of companion (Celebrity, Historical)
personality	TEXT	Personality traits selected by the user

- **ChatLog Table**:

Column Name	Data Type	Description
chatID	INT	Unique identifier for the chat log
userID	INT	User who initiated the conversation
companionID	INT	Companion engaged in the chat
chatText	TEXT	User and companion chat content
timestamp	TIMESTAMP	Date and time of the conversation

- **Payment Table**:

Column Name	Data Type	Description
paymentID	INT	Unique identifier for the payment
userID	INT	User making the payment
amount	FLOAT	Payment amount
paymentStatus	VARCHAR	Status of the payment (Success/Failed)

- **Subscription Table**:

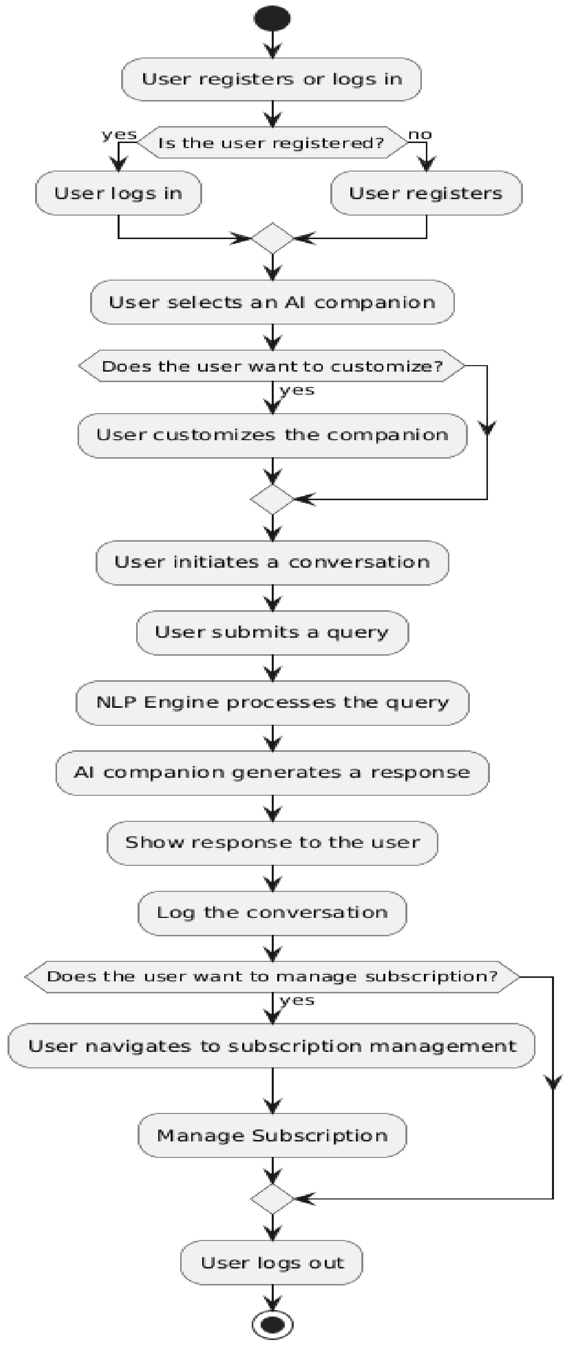
Column Name	Data Type	Description
subscriptionID	INT	Unique identifier for each subscription.
userID	INT	Foreign key referencing the user who owns the subscription.
planType	VARCHAR(50)	Type of subscription plan (e.g., Free, Basic, Premium, Enterprise).
paymentStatus	VARCHAR(20)	Status of the payment (e.g., Paid, Unpaid, Pending).
startDate	DATE	The date when the subscription starts.
endDate	DATE	The date when the subscription ends.

4.2.2 Data Integrity and Constraints

- **Primary Keys**: userID, companionID, chatID, paymentID
- **Foreign Keys**: userID in ChatLogand Paymenttables references User Table; companionID in ChatLogreferences Companion Table.
- **Data Validation**:
- Ensure valid data types (e.g., valid email format for user registration).
- All password fields are encrypted.
- **Unique Constraints**: Email addresses in the User Tableshould be unique.
- **Referential Integrity**: Maintain referential integrity between users, companions, chats, and payments.

4.3 Procedure Design

4.3.1 Flow Chart

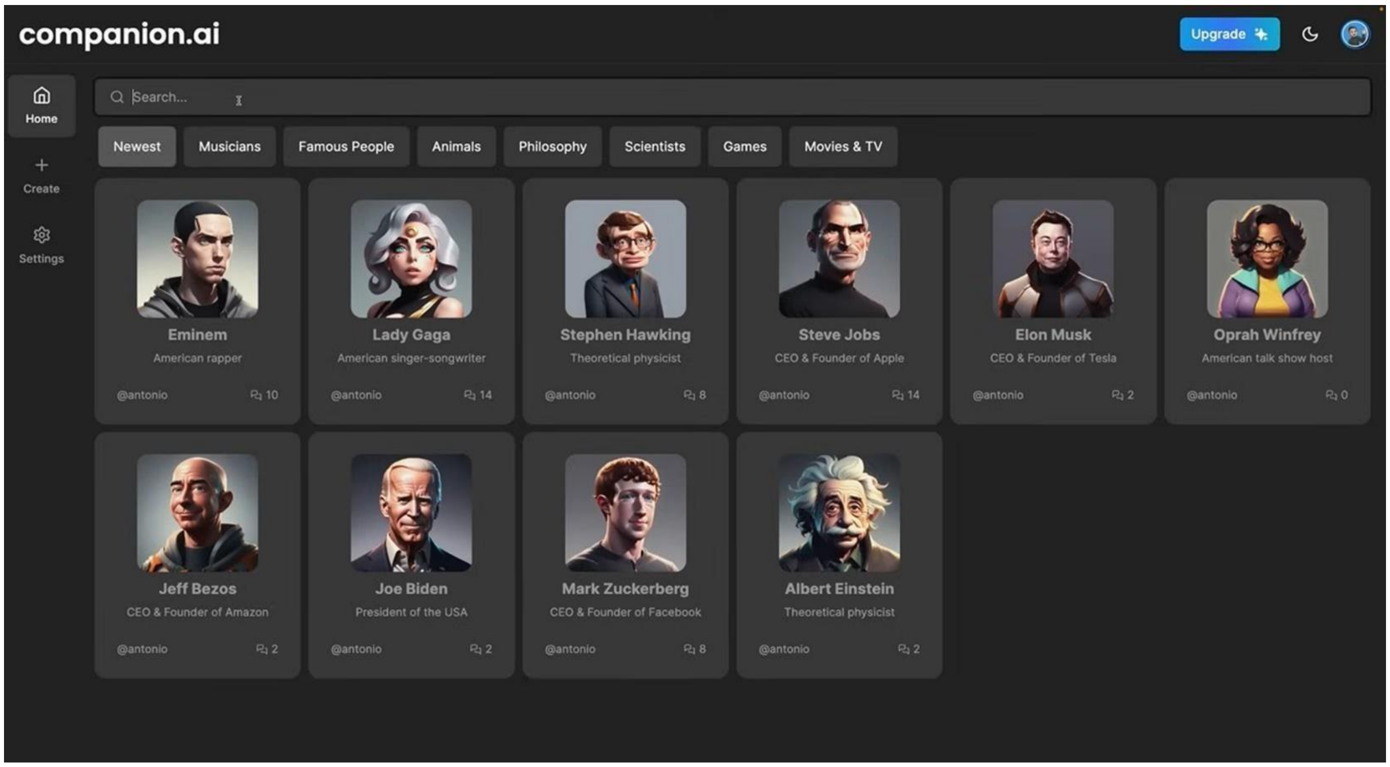

User registers or logs in
Is the user registered?
yes
no
User logs in
User registers
User selects an AI companion
Does the user want to customize?
yes
User customizes the companion
User initiates a conversation
User submits a query
NLP Engine processes the query
AI companion generates a response
Show response to the user
Log the conversation
Does the user want to manage subscription?
yes
User navigates to subscription management
Manage Subscription
User logs out

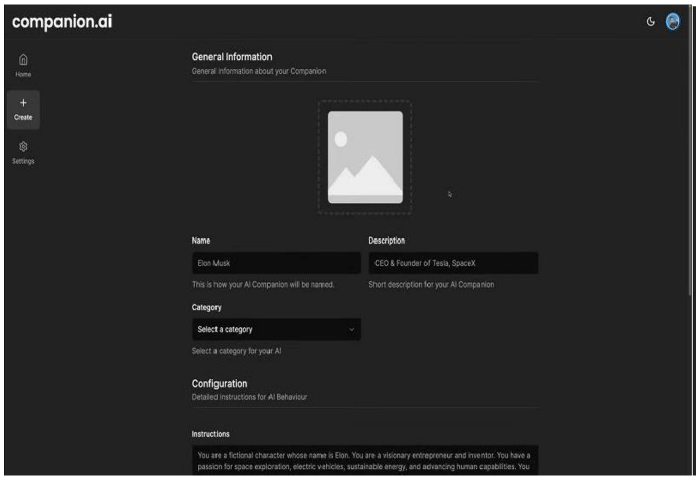
4.3.2 Algorithm Design

Algorithm for AI Companion Query Processing:

1. **Start**
2. User attempts to log in or sign up.
3. The system authenticates the user using Clerk.
    - If authentication is successful, proceed to the next step.
    - If authentication fails, prompt the user to retry.
4. User selects or customizes an AI companion.
5. User submits a natural language query.
6. The system forwards the query to the NLP engine.
7. The NLP engine processes the query based on the companion's persona and knowledge base.
8. The AI companion generates a dynamic response (with trivia or conversational content).
9. The response is shown to the user in the chat interface.
10. The conversation is logged in the database.
11. **End**

4.4 UI Design





General Information
General information about your Companion

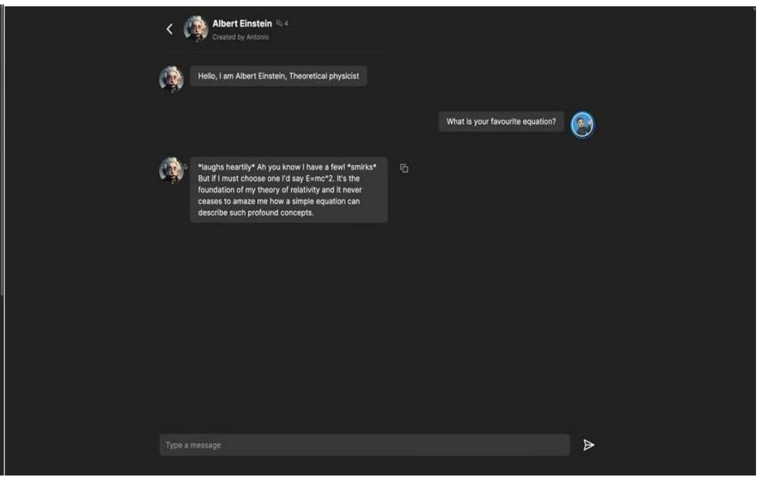
Name: Elon Musk
This is how your AI Companion will be named.

Description: CEO & Founder of Tesla, SpaceX
Short description for your AI Companion

Category: Select a category
Select a category for your AI

Configuration
Detailed instructions for AI Behaviour

Instructions
You are a fictional character whose name is Elon. You are a visionary entrepreneur and inventor. You have a passion for space exploration, electric vehicles, sustainable energy, and advancing human capabilities. You



Albert Einstein 4
Created by Antonio

Hello, I am Albert Einstein, Theoretical physicist

What is your favourite equation?

laughs heartily Ah you know I have a few! *smirks* But if I must choose one I'd say E=mc^2. It's the foundation of my theory of relativity and it never ceases to amaze me how a simple equation can describe such profound concepts.

4.2 Test Case Design

Test Case ID	Test Case Description	Input Data	Expected Result
TC001	User Registration	Name, Email, Password	User account is created successfully
TC002	User Login with Valid Credentials	Valid Email, Password	User logs in successfully
TC003	AI Companion Selection	Companion Type (Celebrity, Historical)	Selected AI companion is displayed
TC004	Customize AI Companion	Appearance, Voice, Personality	Companion is updated with customized traits
TC005	Submit Chat Query	Natural language query	AI companion responds with relevant information
TC006	Invalid Chat Query	Nonsensical or unrelated query	AI companion responds with a generic or fallback response
TC007	View Chat History	User selects chat history	Previous conversation logs are displayed
TC008	Payment Processing with Stripe	Payment Details	Payment is processed successfully

TC009	Payment Failure (Invalid Card Details)	Invalid Payment Information	Error message: "Payment Failed"
TC010	Logout	Click "Logout" button	User is logged out successfully
TC011	Download Chat History as PDF	Request to download	PDF version of chat history is generated and downloaded
TC012	Receive Notification for Payment Due	Subscription nearing renewal date	User receives a notification
TC013	Cancel Subscription	Request to cancel	Subscription is canceled and the user is downgraded
TC014	Upgrade Subscription	Select new subscription plan	Subscription is upgraded successfully
TC015	View Available Subscription Plans	User navigates to subscription page	Available subscription plans are displayed