TED UNIVERSITY

CMPE 491

Senior Design Project I

MESAJLA

High-Level Design Report

27/12/2024

Group Members:

- **Ahmet Tunahan Küçükgökce**
- **Altar Gürsoy**
- **Burak Eren Birinci**
- **Serdar Kemal Topkaya**

Table of Contents

1. Introduction

This report presents the high level design of “Mesajla”, a generative AI powered chatting application that plans to bring a new way for users communicate in digital environments. The project addresses the limitations of existing messaging platforms by combining traditional chat functionalities with innovative AI-generated response suggestions. With a focus on security, reliability, and easy to use, the system is designed to appeal to broad range of users while obeying to data protection rules.

The following sections consist detailed explanation of the system's design goals, architectural framework, and the services provided by its subsystems. The main objective is to create a working application that not only provides real time communication but also enhances user experience through intelligent features and usage of modern technologies.

1.1 Purpose of the system

The purpose of the system is to create a next generation chatting platform that uses generative AI to improve user communication. The application provides secure, real time communication between users through one on one and group chats while creating AI generated response options. By using machine learning algorithm, the system analyses context and brings relevant suggestions, continuing conversations and reducing user involvement.

The application also prioritizes user privacy and data security, ensuring total obedience to regulations such as GDPR. Additional features include media sharing, user profile management, and customizable privacy settings, all designed to create a reliable, safe and scalable communication app usable for personal and professional use cases.

1.2 Design goals

This application aims to offer users fast, secure, reliable and easy-to-use chatting experience. And while implementing these goals, it also differs from the existing chatting applications. One of the key aspects of these differences is that this app has Generative AI support that has suggestions offered simultaneously during chatting. However, developing an AI to provide these services may also have some setbacks. Overall, with the rules that define the structure of the system, there's also some goals which aim to solve these problems. Here's the completion of goals for the application:

- **Security:** Implementing encryption to enhance the authentication of users, protecting their personal data.
- **Performance:** Creating a fast-response environment for both chatting between users and responses of Gen-AI.
- **Usability:** Providing an interface which is straightforward to detect functionalities and easy to use them.
- **Adjusting the Capacity:** Defining the server capacity of handling the number of users with elements like stress tests with the help of tools (k6, Loader.io etc.).
- **Customizability:** Application allows users to change the color and theme of the UI.
- **Detection and Reduction of Bias:** Defining an algorithm while constructing the AI to detect any unintended biases and aim the remove or reduce them.
- **Transparency:** Notifies the parties in the chat when the response is generated by the AI.
- **Privacy:** Ensuring the dialogue between users and the dataset of the AI is hidden and protected from third parties.
- **Live-Trained AI:** The Gen-AI that this system uses is constantly trained with the outputs of the users in their dialogues. This way, the AI can also provide suggestions based on the existing chat history.
- **Synthesized With GDPR:** The application strictly follows the regulations of the GDPR.

1.3 Definitions, acronyms, and abbreviations

- Gen-AI (Generative AI): A sub-set of AI which is used to create images, videos and text-based responses supported by Deep Learning.

- General Data Protection Regulation (GDPR): A regulation of privacy and human-right laws that aim to protect personal data.
- Authentication: Ensures the identification of users with the help of elements such as password, personal e-mail address, username etc.
- User Interface (UI): A layer in websites or mobile applications to allow users to interact with the functionalities of the system.
- Direct Message (DM): Transfer of messages that occur privately between one user to another.

1.4 Overview

This project, a chatting application supported by Gen-AI, aims to achieve an innovative design and functionalities with using the new AI technologies. On top of that, it also applies to an environment for users which is trustful, swift and with uncomplicated structure. The application follows the known regulations such as GDPR. Because of that, the users of this application can communicate with their friends and relatives without concerns about their privacy. With the aid of Generative AI, this project offers a unique experience to their users.

2. Current software architecture (if any)

When it comes to the current applications, they are successful with providing a simple and efficient design to chat among people. But there’s no implementation of Generative AI in the system that aids dialogues between parties yet. Also, about some applications, there are some concerns about privacy and data protection. Here’s some positive aspects we take inspiration from and negative sides to avoid in these platforms:

Limitations:

- Doesn’t Provide AI Supportation: Current systems don’t contain AI integration.
- Lacking Dynamic Chatting: One-to-one communication only relies on the users, missing Gen-AI that provides extra options.
- Privacy and Data Protection: Some apps have some history with the unethical usage of their customers.

Achievements:

- User Authentication: The existing applications are providing reliable and secure options to authenticate their users.
- Clean UI: UIs of these systems are simple to understand and easy-to-use.
- Fast Transmission of Responses: Having fast and uninterrupted network to transfer the dialogues between users.

These aspects from current systems are efficient feedbacks for us to build our project. By considering them, we aim to build our structure more reformed.

3. Proposed software architecture

3.1 Overview

Mesajla aims to make communication secure and make this communication effective with artificial intelligence support. It is a real-time chat application that adapts to the technologies of its age and aims to provide its users with an effective and unique messaging experience. The Mesajla platform uses the latest technologies of machine learning to create effective and consistent response suggestions during user chats, thus enabling faster, more effective and creative chats between users. By integrating these features into a scalable architecture that prioritizes security, it provides an effective user experience by working seamlessly on different devices.

The application serves as a web-based system, so users do not need advanced hardware requirements. Being a web-based system, it can be accessed from anywhere. Artificial intelligence operations are performed on a cloud-based infrastructure, so users do not need to perform artificial intelligence operations that require intensive calculations on their devices, and these operations are completed more quickly on the cloud infrastructure. The most important advantage of this design is that it increases user accessibility by adapting to both mobile and desktop platforms. Thanks to the microservice architecture, the adaptability of our system is increased, ensuring that needs that arise over time can be met. This architecture also allows the components in the system to be developed independently.

3.2 Subsystem decomposition

The messaging system is divided into subsystems that are interconnected. These subsystems are responsible for different functionalities in the application. This division ensures modularity and maintainability.

- Authentication Subsystem: With this subsystem, the user's access to the system is secured. This system manages the users' registration, login and account recovery activities. Users need to log in or register to the system to access their chats and profile. Thanks to the authentication system, this user information is protected.

- Chat Management Subsystem: This system ensures that real-time messaging is carried out properly. It supports file and photo sharing. It provides group chat to enhance communication between different users.
- Generative AI Subsystem: This subsystem uses machine learning to analyze the conversation between users and create appropriate responses. The model trained on user data creates intelligent and relevant responses. It offers users the option to request new suggestions through the filtering option.
- Data Management Subsystem: This system securely stores users' profile information and settings, chat histories, and data. It uses these chat histories for context analysis. It performs regular data backups to prevent data loss.
- User Profile Management Subsystem: This system provides the user interface. It allows users to use the application comfortably, set their profiles and access their chats. With profile setting, they can change or update personal information, profile photo and name, with security setting, choosing who to block or not, with friend management, accepting or rejecting friend requests and controlling their friend lists.

3.3 Hardware/software mapping

The hardware and software components of the message are mapped to keep performance at an optimum level and to ensure reliability. In order to create a successful system, these two components must be very well connected. Using the latest technologies, the system can perform user-oriented operations and backend operations.

Hardware Components:

We divide hardware components into those used by developers and those used by clients.

Developer Side

- Developer Infrastructure: It requires high-performance servers with GPUs to train AI models and perform backend operations. It prevents interruptions in communication thanks to the secure networks it uses. It avoids situations such as unauthorized access by keeping security at a high level.

Client Side

- User Devices: Users can access the system not only from a web browser but also from platforms such as mobile and desktop, which makes accessibility to the system easier and more flexible.

Software Components:

Our system has different areas such as machine learning and web development. Different software systems are used in these areas. The software systems we use in these areas are:

- AI and Data Processing: Machine learning tasks are performed using TensorFlow and PyTorch. Message contents are analyzed with OpenCV. Using these tools, accurate results are obtained.
- Web Development:

  Frontend Development:

  React.js is used to create dynamic and smooth user interfaces. React Router allows the user to navigate between application pages. By improving the visual design with CSS and libraries, a better user experience is provided.

  - React.js: Creating a user interface with a component-based structure.
  - React Router: Navigation between different pages.
  - CSS Libraries (TailwindCSS): Used for styling the user interface.

  Backend Development (Backend-as-a-Service):

Firebase is a back-end platform that uses authentication to provide more secure user management, Cloud Firestore that allows real-time message synchronization, and Firebase Storage for media usage. The Firebase SDK is used to connect these services to the application.

- Firebase
    - Authentication: User login, sign-up, and secure authentication.
    - Cloud Firestore: A NoSQL database for storing and syncing chat messages and user data.
    - Firebase Storage: Stores media files. (Profile pictures or shared images in chat.)
    - Firebase SDK: Connects the application with Firebase services for backend functionality.

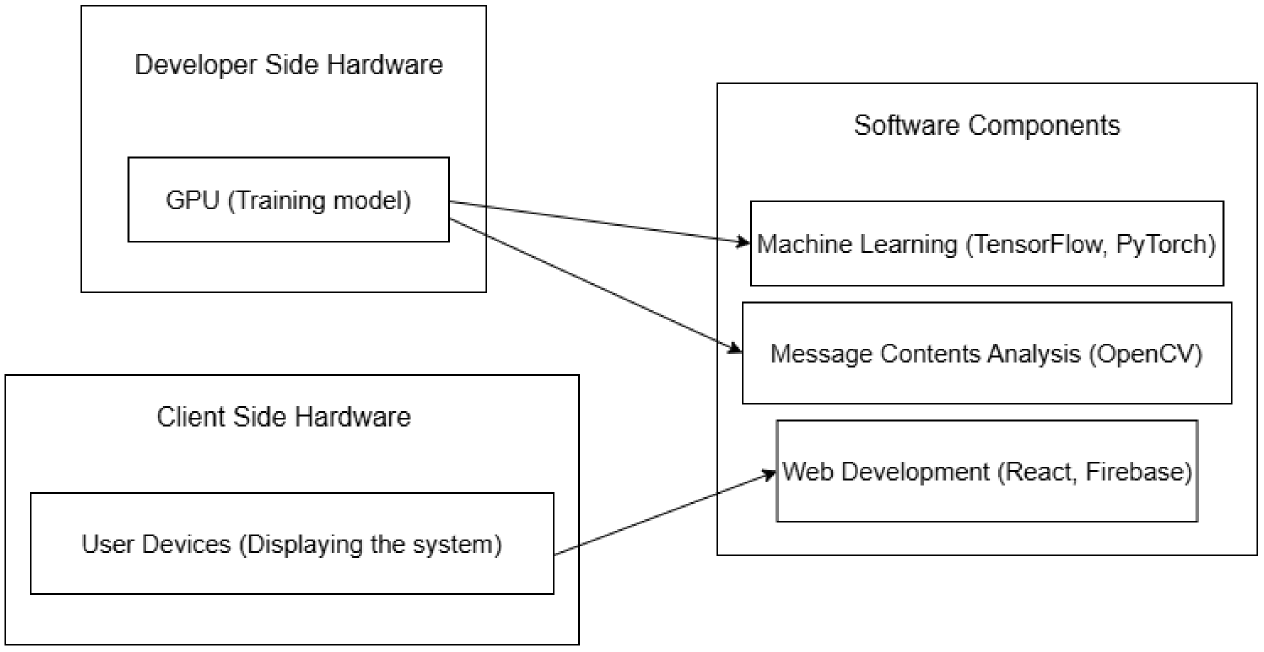
Developer Side Hardware
GPU (Training model)
Software Components
Machine Learning (TensorFlow, PyTorch)
Message Contents Analysis (OpenCV)
Client Side Hardware
Web Development (React, Firebase)
User Devices (Displaying the system)

Figure 1: Hardware/Software Mapping

3.4 Persistent data management

The application offers reliable, secure and consistent data storage. It also has algorithms to access the elements from their table faster, increasing the performance. To achieve this, the database is constructed with Firebase software. It has tables for user profiles, their information, chatlogs and dataset of the AI. While indexing this information, the GDPR rules are ensured to be implemented. Because of that, users are allowed to access and remove the chat history data. Also, partitioning the rows such as chat room or user IDs logically increases the performance. Personal data is encrypted with encryption algorithms.(with hashed passwords and security responses) On top of that, the data is stored in the cloud-base environment. In the end, the tables that the Generative AI uses has frequent inspections to prevent unethical actions.

3.5 Access control and security

To provide users with a safe authorization and protect their data, there are some key implementations. First, the application blocks unauthorized access to different people's dashboards. Also, the information of users is stored securely, inaccessible from third parties. The dataset of Gen-AI also considers this statement, not reachable to prevent harmful actions against users. Besides that, the app also follows the GDPR laws and orders by allowing users to delete their chat history, or halt sending data to the dataset of the Gen-AI etc. Finally, the system suggests to their users to create a strong password to make the account more secure.

3.6 Global software control

The global software control with "message" is to provide the management of requests and communication between subsystems. In web applications, the actions taken by users cause requests to be realized. An example is button clicks. These actions are checked and responses are returned about the status of these actions. These responses are asynchronous responses, including success and failure. The subsystems in our system perform synchronization using these responses. In this way, an effective functionality flow is expected.

Timeout mechanisms are used in communication between different parts of the system. These mechanisms contribute to general synchronization and also increase the security of the system. For example, when the user wants to log in, there is a short-term wait for the completion of the login and backend processes, and after this wait, the user reaches the interface. Similarly, when the responses created using Ai are expected, the system waits for the request to be successful, and after it is successful, it performs the synchronization process to display the suggestions. Thanks to this approach, while a flawless communication between subsystems is aimed, user experience and system stability are increased.

3.7 Boundary conditions

The system is designed to stay reliable despite unexpected circumstances. For instances of login errors, users can securely reset their login information using methods like email verification or security questions. In the event of system downtime, operations will resume after server recovery, with users being notified through maintenance messages. To reduce risks of data corruption or loss, the system uses backups and does periodic controls to ensure the system is working and data is in good condition. Additionally, unexpected user actions are handled through user friendly error messages and error pop-ups and user guidance, ensuring a smooth and reliable experience.

4. Subsystem services

The proposed system is consisting of several subsystems, each having specific functionalities important for the application's usage. These subsystems are designed to work flawless, creating an innovative and satisfying user experience.

1. **Authentication Subsystem:**
   The authentication subsystem ensures secure access to the application, it is first step for user to access application whether it is Logging in or Registering for the first time. It consists:
   - User Registration: When a user provides their email and password, the system saves the password using a secure algorithm before storing it in the database.
     1. Accept user input for email and password.
     2. Ensure the information and save it to database.
     3. Return success or failure response.
   - User Login: Checks user provided information by comparing the password with the stored information.
     1. Retrieve user details based on the entered email.
     2. Compare the provided password with the stored password using a comparison function.
     3. If valid, authenticate user to enter their account.

2. **User Profile Management Subsystem:**
   This subsystem allows users to personalize and control their profiles Adding profile pictures controlling their privacy or setting their security settings. Features include:
   - Profile Setup: Users upload a profile picture and provide details such as a display name and status, which are validated and stored securely.
     1. Receive input data and check image format and text content.

2. Store the information in the database associated with the user ID.

- Privacy Controls: Provides customizable options for managing who can view user details and securing users privacy.
    1. Retrieve privacy options decided by user.
    2. Apply filters based on the user's relationship with the other user.
    3. Accept or deny access to the requested information.
- Security Settings: Allows users to manage blocking or unblocking accounts.
    1. Decide the account to block or unblock.
    2. Update the "blocked users" field in the database.
    3. Make sure no conversation is happening between blocked accounts.
- Friend Management: Handles sending friend requests and managing the friend list.
    1. Check the email address of the target user.
    2. Create a friend request entry in the database.
    3. If accepted, update the relationship status to "friends."

3. Chat Management Subsystem:

The messaging subsystem provides communication functions messaging setting up group chat or sending photos, videos and supported multimedia types, such as:

- One on One Chat: Provides direct communication between two users in chat room.
    1. Identify participants and create a chat room entry in the database if it doesn't exist.
    2. Store each message with data (time, sender, status).
    3. Send messages with message sending protocols.
- Group Chat**:** Supports multiple users in a single chat room.

        1. Create a group chat entry with an list of participants.
        2. Sent message to all participants with the help of a multicast system.
- Multimedia Support: Uploads and shares multimedia files.
    1. Check file type and size.
    2. Store files in a cloud storage.
    3. Send a link to participants, to access sent multimedia.
- Message Status Indicators: Shows message status as sent or delivered.
    1. Attach a status to each message.
    2. Update the status based on server message.

4. **Generative AI Subsystem:**

   The generative AI subsystem is the main subsystem of this project it is a subsystem that provides usable replies to user that they can choose from, it consists of:

   - Text Analysis: Examine message context to understand user interaction.
       1. Send the text and extract words to AI.
       2. Use analysis to classify and understand meaning of the content.
   - Response Generation: Generates responses using a language model.
       1. Input chat history and context into the model.
       2. Generate multiple responses and filter irrelevant content.
   - Ordering and Filtering: Prioritizes responses based on relevance.
       1. Rank responses based on relevance.

Display the top ranked responses to the user interface

5. **Database Management Subsystem:**

This subsystem handles the storage and retrieval of all application data, system uses Firebase as its database system and handles all sorts of storage, including:

- User Data: Stores user information and settings in Firebase.
- Chat History: Stores conversations with timestamps and involved participants.
- AI Data: Stores data of AI usage of users in the app in order to learn to give more suitable answers.

5. Glossary

- **Authentication:** The process of verifying the identity of a user, typically using information such as a username and password.
- **Generative AI (Gen-AI):** A artificial intelligence type used to generate content, such as text, based on learned patterns.
- **General Data Protection Regulation (GDPR):** A European Union regulation that decides the rules of the collection and use of personal data, ensuring privacy and security.
- **User Interface (UI):** The graphical layout through which users interact with the application, including buttons, menus, and forms.
- **Firebase**: A cloud-based platform providing tools such as authentication, real-time databases, and cloud storage to support application development.
- **Machine Learning:** An artificial intelligence type focused on training systems to learn and make predictions based on data.
- **Two-Factor Authentication (2FA):** An additional layer of security requiring users to provide two forms of identification for login.

6. References

1. GDPR.eu. (n.d.). *General Data Protection Regulation (GDPR)*. Retrieved from https://gdpr.eu/
2. Firebase. (n.d.). *Firebase: Build apps fast, without managing infrastructure*. Retrieved from https://firebase.google.com/
3. Auth0. (n.d.). *Hashing passwords: One-way road to security*. Retrieved December 27, 2024, from https://auth0.com/blog/hashing-passwords-one-way-road-to-security/