M5: Final Report and Delivery

Introducing our app, MangoChat! This chat application is designed for communication and collaboration between users. New users can easily register, while existing users can log in to access their accounts. Once in, users can join and exit channels, and engage in group discussions. Users can also create new channels, becoming admins. Admins have additional privileges, including channel management, user removal, and the ability to appoint other users as channel admins. Our system ensures privacy and security by hiding channels that users are not a part of, maintaining a comprehensive database of channels, messages, and admin roles.

User Groups:

1. **Professional Collaborator** – Uses the app for work-related collaboration, meetings, and project discussions.
2. **Gaming Enthusiast** – Utilizes the app primarily for gaming, connecting with friends for multiplayer sessions, and discussing gaming strategies.
3. **Social Butterfly** – Engages with the app to stay connected with friends, share updates, and participate in various community discussions.
4. **Study Buddy** – Relies on the app for study groups, sharing educational resources, and discussing academic topics with peers**.**

Scenario Stories:

1. **Professional Collaborator**

    - **Scenario:** Mark, a project manager, relies on MangoChat for work-related collaboration with his team members who are spread across different locations.
    - **Story:** Mark logs into MangoChat and joins the “Top Secret Corporate Project” channel where his team discusses updates and tasks for their ongoing project. He schedules a meeting in the “Weekly Meetings” channel and invites all project members. During the meeting, they review progress and brainstorm ideas. Mark also utilizes the app to distribute new tasks and receive feedback from team members in real-time.

2. **Gaming Enthusiast**

    - **Scenario:** Sarah is an avid gamer who uses MangoChat to connect with her friends for multiplayer gaming sessions.
    - **Story:** Sarah opens MangoChat and sees messages from her gaming friends in the "Gaming Crew" channel. They are planning to play a popular multiplayer game later in the evening. She quickly joins the chat to discuss strategies and coordinate their gameplay. Sarah also checks the "Game News" channel to stay updated on the latest patches and updates for her favorite games. After an intense gaming session, they share highlights and memorable moments in the "Sweat Lodge" channel.

3. **Social Butterfly**

    - **Scenario:** Alex is a social butterfly who loves connecting with friends and engaging in community discussions on MangoChat.
    - **Story:** Alex logs into MangoChat and scrolls through the "General Chat" channel to catch up on the latest news and updates from friends. He shares stories from his recent hiking trip in the "Adventure Lovers" channel and receives enthusiastic responses from fellow adventurers. Alex also participates in a debate about a favorite TV show in the "Pop Culture" channel, sharing his opinions and engaging in friendly banter with other members.

4. **Study Buddy**

    - **Scenario:** Emily is a student who relies on MangoChat for study groups and academic discussions with classmates.
    - **Story:** Emily opens MangoChat and joins the "Biology Study Group" channel where her classmates are discussing an upcoming exam. They share study resources and quiz each other on key concepts, such as how the mitochondria is the powerhouse of the cell. Emily also posts a question in the "Math Help" channel and receives helpful explanations from other students. She finds a study partner in the "Peer Tutoring" channel and schedules a study session to prepare for their next test.

Requirements:

1. **Allow users to log into account.**
2. **Users can only view channels and friends when logged in.**
3. **Allow new users to register for an account.**
4. **Allow users to create/delete channels.**
5. **Allow users to join/leave channels.**
6. **Allow users to send messages to channels.**
7. **Allow users to add/remove friends.**
8. **Invite other users to channels.**

Admin Requirements:

1. **Admins should be able to remove members of channels.**
2. **Admins should be able to appoint other users as admins of channels.**

System Requirements:

1. **Database containing all channels, including messages, connected users, banned users and admins.**
2. **Store channel IDs to invite other users to the channel.**
3. **System should store a list of admins and users.**
4. **System should store IDs of user friends.**
5. **System should store IDs of blocked users.**
6. **Hide channels user is not in.**

Non-Functional Requirements:

1. **System should store the user's personal info.**
2. **System should not disclose login information with other users.**

Status of Software Implementation:

Our team embarked on the development journey of MangoChat with a clear set of requirements aimed at delivering a functional communication platform. Here is a detailed overview of the status of our software implementation:

Checklist of Initial Requirements

Requirement	Status
Allow Users to Log into Account	Completed
Allow new Users to Register for an Account	Completed
Allow Users to Create/Delete Channels	Completed
Allow Users to Send Messages to Channels	Completed
Admins should be able to Remove members of Channel	Completed
Admins should be able to Appoint other users as Admins of Channel	Completed
Database containing all Channels, including Messages, Connected Users, Banned Users and Admins	Completed
System should store a list of Admins and Users	Completed
System should store IDs of User Friends	Completed
System should store IDs of Blocked Users	Completed
Hide Channels User is not in	Completed
System should store User's personal info	Completed
System should not disclose login information with other users	Completed
Allow Users to Join/Leave Channels	Partially Completed
Allow Users to Add/Remove Friends	Not Completed
Invite other Users to Channel	Not Completed

Analysis of Implemented Requirements

- **Completion:** We are pleased to report that our team has successfully implemented most of the initial requirements outlined for MangoChat.
- **Requirement Details:** The initial requirements sufficiently captured the necessary details for the project. They provided a solid foundation for development, guiding our team through each stage of implementation.

Tasks Remaining in the Backlog

- **Full Integration of Join/Leave Functionality:** Although Users can join and leave channels through the web page, they are currently unable to permanently join the database. The backend functionality has been created to facilitate this but has not been fully integrated with the front end.
- **Add/Delete Friend and Block Users:** The functionality for adding and deleting friends, as well as blocking users, has not been created. By reusing the structure of the previous code, this addition should be relatively easy to implement in the future.
- **Invite Users to Channel:** Similar to the Add/Delete Friend functionality, inviting Users has not been fully integrated into the app.

System Architecture

- **Key Components:**
    - **Frontend:** HTML, CSS, JavaScript
    - **Backend:** Python
    - **Database:** SQL
- **Design Patterns and Principles:**
    - **Singleton Pattern:** The “app” object is essentially a singleton. It is created once and reused throughout the application.
    - **Model-View-Controller Pattern:** Implemented for clear separation of concerns, enhancing maintainability and scalability.
    - **Single Responsibility Principle:** The app is structured to ensure that each function within the app falls under the Single Responsibility Principle of SOLID.
    - **Open/Closed Principle:** All classes and methods in the backend closely follow the Open/Closed Principle of SOLID, being open to extension and closed to modification.

Degree of Reuse

- **Testing Reuse:** We used Pytest to test our backend functionalities. By using test fixtures we were able to create tests for each functionality very quickly.
- **Backend Services:** The structure of the functionalities are coded in very similar ways. Due to our implementation of the MVC Pattern we simply had to create methods that updated the database for most functionalities. By reusing this code structure the implementation of each functionality was easier as the development continued.
- **Database Structure:** Each table in the database follows a similar structure. Users and channels are connected via the UserChannel table. Messages are connected to the Channel table with a foreign key. This structure of the database allows for simple integration of frontend and backend services. It also lowers coupling between frontend and backend, allowing for modification of either without needing to modify the other.

Known Bugs

While MangoChat has undergone rigorous testing, there are a few known bugs that have been identified:

1. **Issue:** If a user writes for more than 10 seconds in the channel's text box their cursor will be removed from the text box and to continue typing they will have to manually click back into the text box.
   Fix: Implement an asynchronous request to refresh the comments rather than the page itself.

2. **Issue:** The current iteration of Join/Leave functionality does not add the user to the UserChannel table in the database so they are only in the channel while they keep the web page open.
   Fix: A Join/Leave Functionality has been created in the backend with python, but the frontend has not yet implemented it. Implementation of this functionality into the frontend needs to be done.

3. **Issue:** Implementing duplicate channels returns an IntegrityError page.
   Fix: Error handling needs to be added to the frontend to give an error message to the user instead of letting the page crash.

Bug Fixing Strategies

- **Testing and Debugging:** Our team, or a future team, can conduct thorough testing, including unit tests and end-to-end tests, to identify and isolate bugs.
- **Issue Tracking:** Teams can utilize the Kanban board on GitHub to prioritize and address reported bugs promptly.
- **Patch Releases:** Regular patch releases can be used to implement bug fixes and enhancements.

Conclusion

In conclusion, MangoChat has successfully delivered all initial requirements, showcasing a functional communication platform. The system architecture, design patterns, and degree of reusability reflect our commitment to building a scalable and efficient application. Despite a few identified bugs, or needed polish, we believe that our team, or a future team, can address these issues promptly to ensure a seamless user experience.

Handover Guide:

Running the website:

Dependencies:

- Docker
- Git Bash or Bash
- Python

Install steps:

1. Open this location in Bash or Git Bash.
2. run `./setup.sh`.
3. Wait until it says “Serving Flask app ‘app’”.
4. To take down the server, hit CTRL-c, and then run `docker-compose down`.

Team Reflection:

1. Our project management for the team worked well. The most difficult thing was keeping each team member up to date with where different parts of the project were coming along. Keeping on top of this aspect better could have allowed for more efficient prioritization of development. To accomplish this in the future we could spend more thorough in our Scrum Meetings.
2. The initial requirements were properly detailed. Our team found that there was no need for significant changes or clarifications to the requirements as the project was developed.
3. Our initial planning was fairly thorough. We feel that there was nothing major that was missed while development ensued.
4. Our testing was very well implemented from the start of the project to the end. We would probably handle it in a similar fashion in the future.
5. The group is mixed on expectations vs. reality for the project. Some of us felt that the project ended up being less work than initially expected, while others found it to be more work than expected.
6. Our group is proud of the simple fact that we were able to get the basics of the app functioning properly. We also feel that we better understand the process of software development, using GitHub as a group, and learning new programming languages on the fly.

Demo Video Link:

https://youtu.be/KWNWH00tz3E