Hermes Chat

Software Engineering Lab

Group Members

Alan Tony - 191CS207

Ashutosh Anand - 191CS111

Lakshmi Aashish Prateek Janaswamy - 191CS225

Sudarshan Sundarrajan - 191CS255

Table of Contents

Introduction

Introduction

Hermes is a highly scalable real-time chat application available to many users around the world. It offers a medium of connection between people and creates an online platform to have casual conversations as well as manage industrial teams.

Hermes offers a slew of features that enables an environment where groups and individuals can have formal or informal conversations with increased productivity.

Features of Hermes

Features of Hermes

Hermes offers a rich set of features that enables conversations between people and brings in increased productivity for organizations.

The list of features are as follows:

1. User dashboard
2. Groups and One-to-one messaging
3. Channels
4. Permissions
5. Roles
6. Event scheduler & Calendar

Tech Stack and Requirements

Tech Stack

The application is built using the following technologies:

1. Front-end: Flutter, Dart
2. Back-end: NodeJS
3. Database: MongoDB
4. Socket.io to implement web-sockets in Dart and Node.JS for real-time communication.

Software Requirements

The user needs to have:

OS (Android 8 and above, iOS 9 and above)

Hardware Requirements

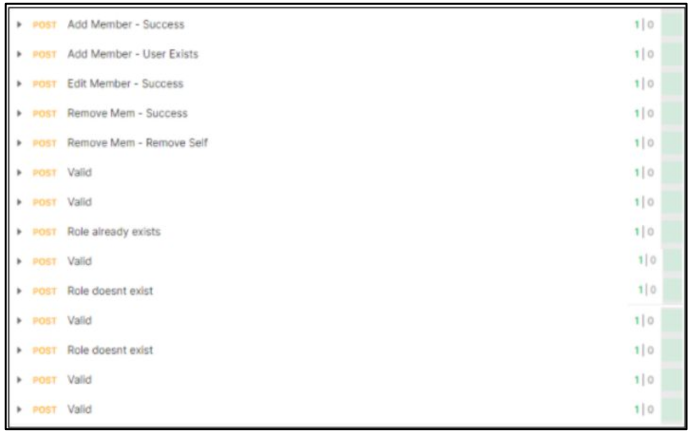
User device hardware requirements

1. 2GB+ RAM
2. Quad-core Mobile Processor (1.5GHz+ Clock Rate)
3. 200MB+ storage
4. Mobile interface (touchscreen)
5. Working internet connectivity (1 Mbps+ connections)

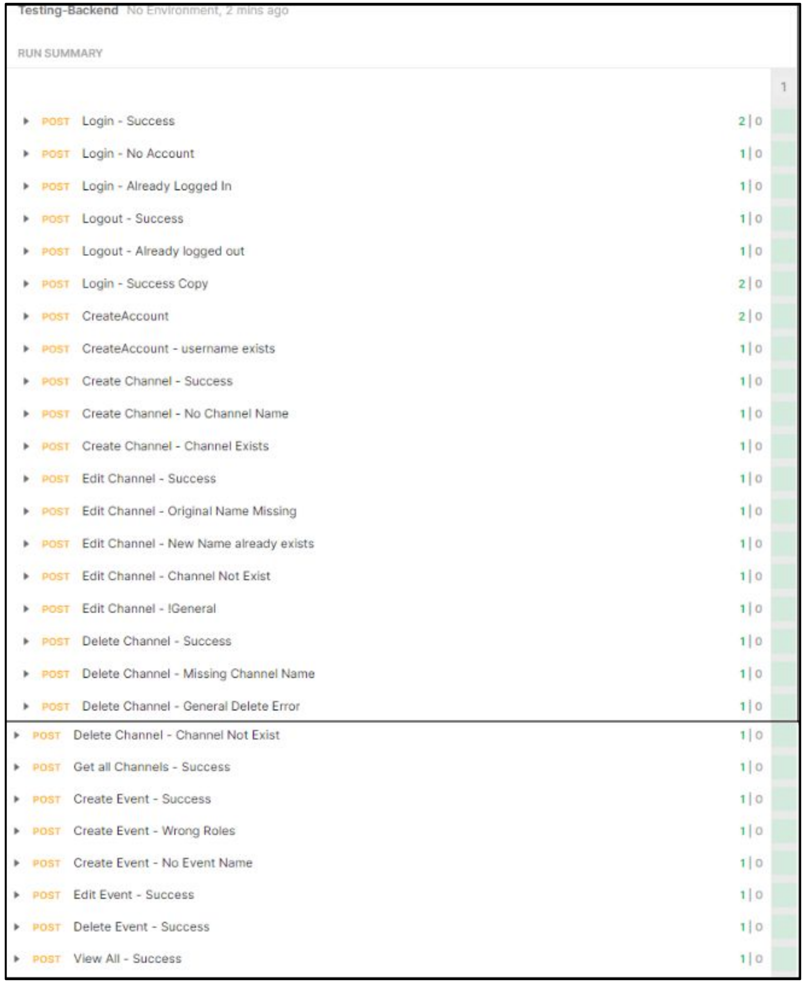
Testing

Backend

Both unit testing and integration testing for the backend was automated using Postman scripts.



POST	Add Member - Success	1 | 0
POST	Add Member - User Exists	1 | 0
POST	Edit Member - Success	1 | 0
POST	Remove Mem - Success	1 | 0
POST	Remove Mem - Remove Self	1 | 0
POST	Valid	1 | 0
POST	Valid	1 | 0
POST	Role already exists	1 | 0
POST	Valid	1 | 0
POST	Role doesnt exist	1 | 0
POST	Valid	1 | 0
POST	Role doesnt exist	1 | 0
POST	Valid	1 | 0
POST	Valid	1 | 0



Testing-Backend No Environment, 2 mins ago

RUN SUMMARY

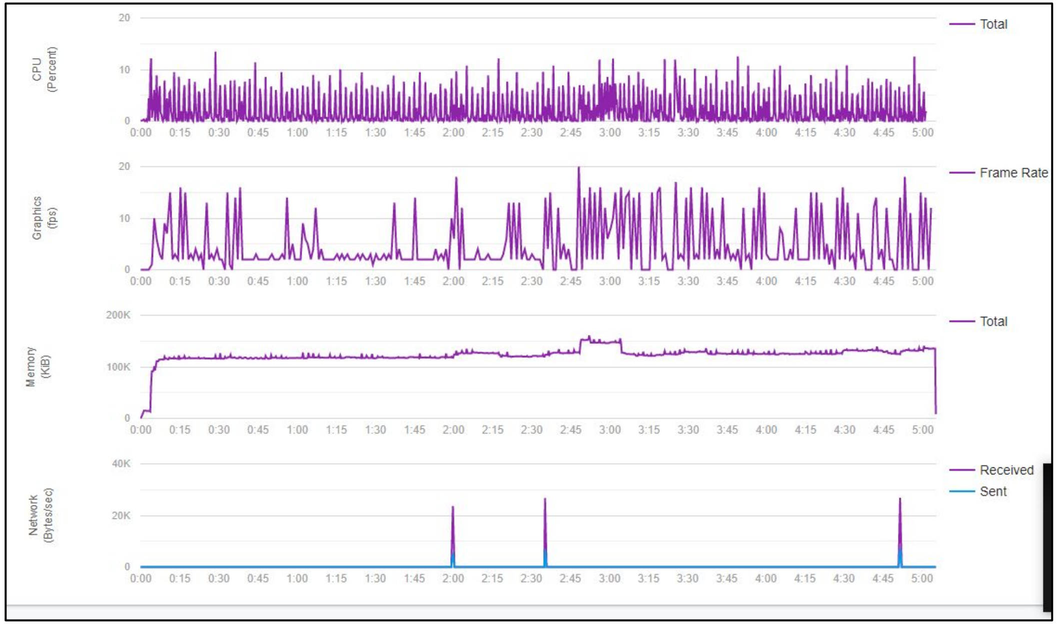
1

POST	Login - Success	2 | 0
POST	Login - No Account	1 | 0
POST	Login - Already Logged In	1 | 0
POST	Logout - Success	1 | 0
POST	Logout - Already logged out	1 | 0
POST	Login - Success Copy	2 | 0
POST	CreateAccount	2 | 0
POST	CreateAccount - username exists	1 | 0
POST	Create Channel - Success	1 | 0
POST	Create Channel - No Channel Name	1 | 0
POST	Create Channel - Channel Exists	1 | 0
POST	Edit Channel - Success	1 | 0
POST	Edit Channel - Original Name Missing	1 | 0
POST	Edit Channel - New Name already exists	1 | 0
POST	Edit Channel - Channel Not Exist	1 | 0
POST	Edit Channel - !General	1 | 0
POST	Delete Channel - Success	1 | 0
POST	Delete Channel - Missing Channel Name	1 | 0
POST	Delete Channel - General Delete Error	1 | 0
POST	Delete Channel - Channel Not Exist	1 | 0
POST	Get all Channels - Success	1 | 0
POST	Create Event - Success	1 | 0
POST	Create Event - Wrong Roles	1 | 0
POST	Create Event - No Event Name	1 | 0
POST	Edit Event - Success	1 | 0
POST	Delete Event - Success	1 | 0
POST	View All - Success	1 | 0

Frontend

We manually tested various edge cases both on the Android Studio emulator and on the app installed on our personal devices.

Firebase Test Lab was also used to obtain performance result data by performing the automated Robo Test.



Maintenance

Maintenance

1. To prevent possibilities of leaking passwords from the database, we stored the salted and hashed password.
2. We use JWT Tokens to authorize users to make the browsing along the application seamless while maintaining high security and privacy for the user
3. To maintain secure connection and communication between client-server for text messaging and authorization, we use HTTPS.
4. Each group in the application has various roles that its members can define.
5. We deployed our application to Heroku Cloud and are currently maintaining a Node web server and a MongoDB Atlas cloud database.
6. The client side software is to be installed as an apk from the client side on Android.
7. DNS server is maintained by Heroku Cloud platform.

Implementation

Features Implemented

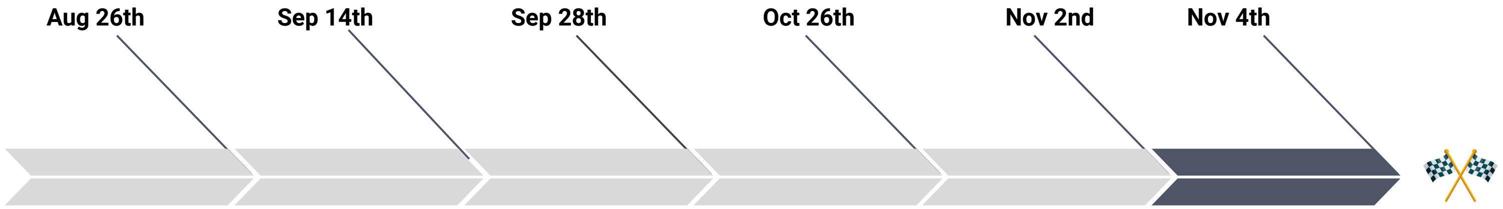
- Login/Registration
- Direct Messaging
- Groups
- Channels (with messaging)
- Roles
- Event Scheduler

Progress and Future Work

Progress

- As of now, all the functionalities have been implemented
- Testing has been done automatically on the backend and manually for frontend on emulators and on our personal Android devices.
- Security requirements are met (but there is scope for further improvements in security)
- The server has been deployed using Heroku
- The database has been ported from the local machine to MongoDB Atlas for cloud storage
- We have a working release APK

Progress Chart



Idea Inception

Brainstorming for Hermes

Requirements Engineering

Gather requirements from client

Modelling and Design

Model the application in different ways in the form of DFD, AD, & SA

Implementation

Implement all features in backend and frontend.

Testing

Automated testing for backend and manual testing for frontend

Deployment

Deployed server on Heroku, generated working APK

Future Work

- Optimize the working of the application (by introducing caching and message queues)
- Scaling up the application if and when the user base grows
- Try working on end-to-end encryption to further secure messages
- Include support for file transfer
- Implementing a web application as frontend to enhance portability
- Get a signed APK to get recognized by Play Services and the Google Play Store.