Name: Asish Addanki

Asu ID: 1230916442

Alignment with guardian angel:

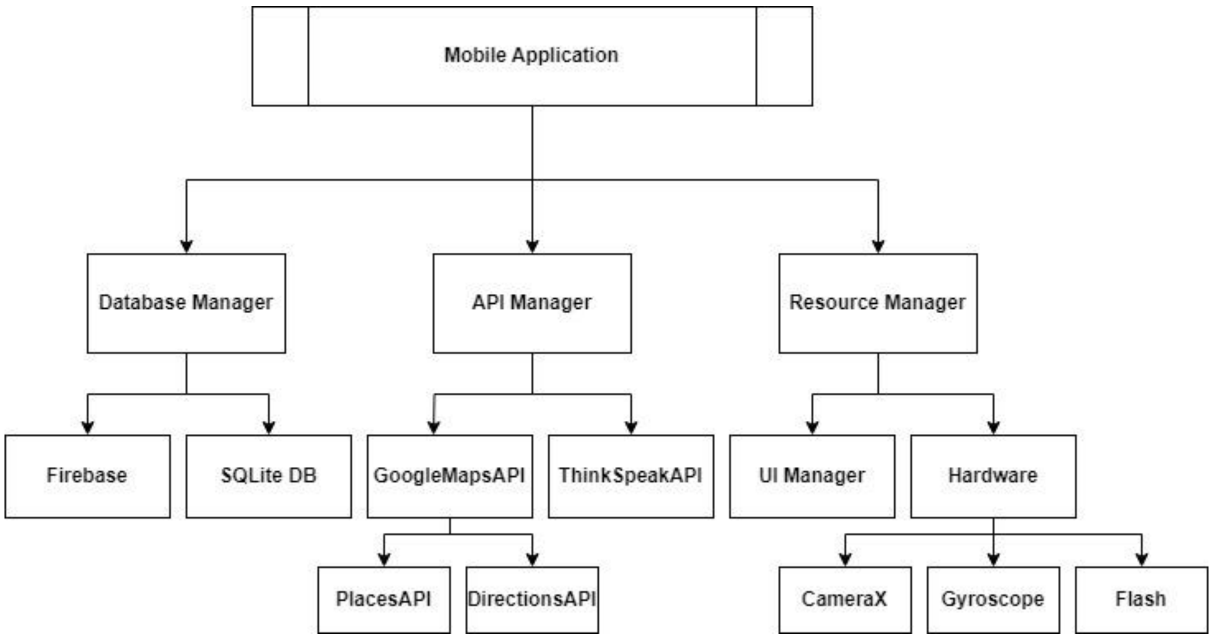
The initial login page is equipped with fields for an avatar, user's Gmail address, and password. Users also have the option to sign in using their Google account, and this feature is seamlessly integrated with the Firebase database. When users opt for the "sign in with Google" option, a popup window appears, prompting them to select their Gmail ID for logging into the Guardian Angel application.

Moving to the second page, which focuses on vehicle information, the collected data encompasses both vehicle and travel details. Specifically, vehicle information includes metrics such as the distance between two cars and the speed of each individual car. Meanwhile, travel information comprises the source and destination of the journey, coupled with insights into road conditions.

Notably, the vehicle page incorporates the Google Maps API, providing users with a geographical map of their city. The map dynamically displays real-time traffic conditions, categorizing them as high, medium, or low. This valuable information is subsequently shared with MATLAB, contributing to its decision-making process.

Ensuring consistent and accurate data persistence in a database is crucial for maintaining the integrity and reliability of an application. This process involves several considerations and practices to guarantee that every piece of entered data is appropriately and reliably written to the database.

Specifications:



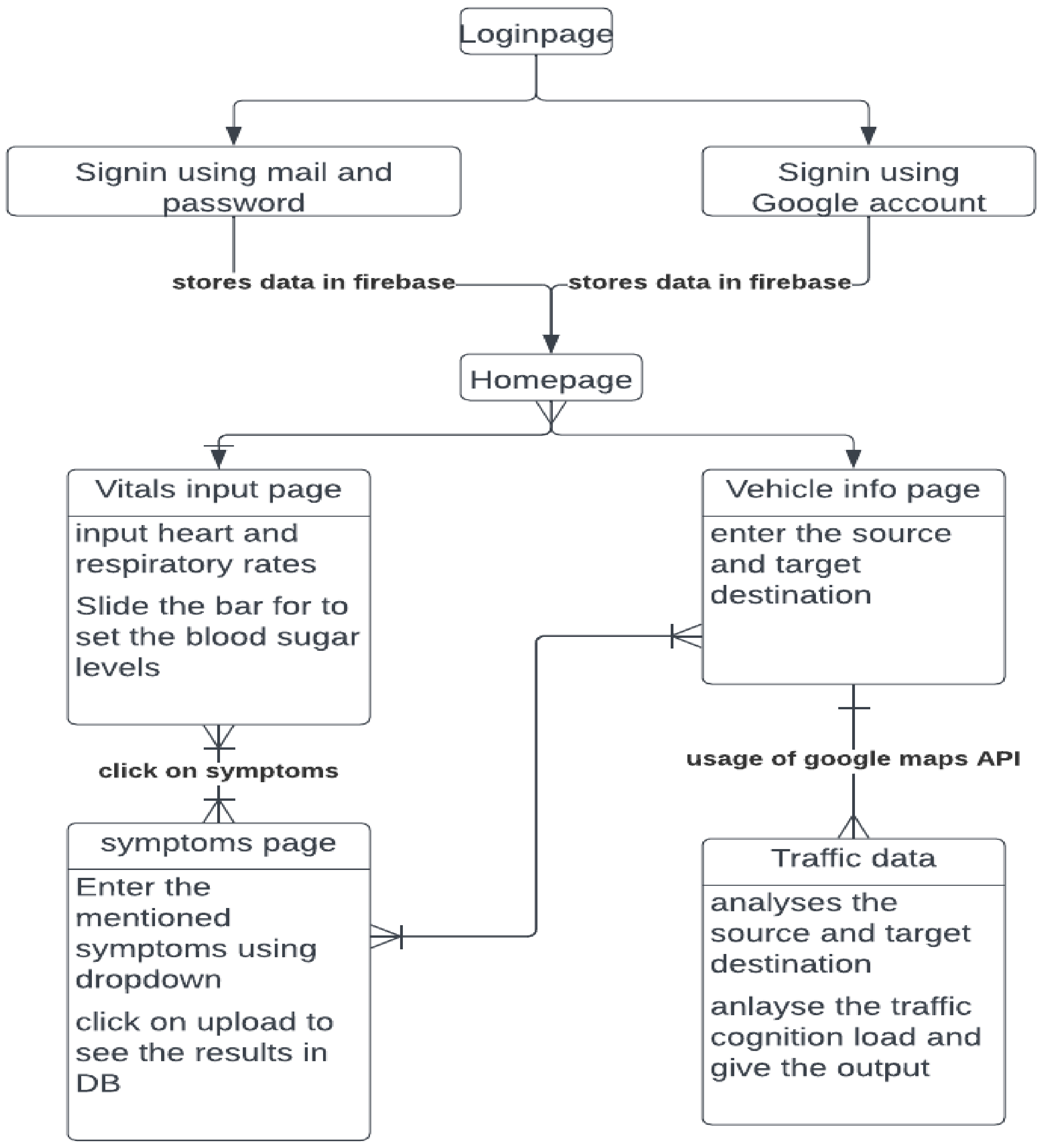
Database manager:

The database manger is integrated with the mobile application to make sure that the data that the user enters is being loaded into the DB without any issues. It makes use of Firebase for recording and maintain the login details and SQLite DB is used to store the data that the user enters in the application such as heartrate, resp rate, blood sugar levels and the symptoms.

UI manager:

The UI manager includes all the UI app pages that are developed and it includes the pages of login page and the vehicle information page that gets navigated from the symptoms page where it contains the search sections to enter the source and target destinations and the Google maps API uses the inputs to determine the traffic conditions and pass on the data to MATLAB determined if the traffic cognition load is high or medium or low.

Design:


Loginpage
Signin using mail and password
Signin using Google account
stores data in firebase
stores data in firebase
Homepage
Vitals input page
input heart and respiratory rates
Slide the bar for to set the blood sugar levels
Vehicle info page
enter the source and target destination
click on symptoms
usage of google maps API
symptoms page
Enter the mentioned symptoms using dropdown
click on upload to see the results in DB
Traffic data
analyses the source and target destination
anlayse the traffic cognition load and give the output

Testing strategies:

The authentication module warrants meticulous attention, involving rigorous testing of Gmail authentication and password handling mechanisms. This includes scrutinizing the seamless integration with Google accounts, testing various scenarios such as selecting different Gmail IDs, and ensuring effective handling of authentication errors.

crucial to confirm that data traverses accurately between different stages, particularly from user input to Firebase and vice versa. This involves validating the correctness of data formatting for Firebase, guaranteeing a seamless exchange of information within the application's ecosystem Via the usage of Firebase.

Ensuring the accuracy of displayed road conditions within the application is crucial for providing users with reliable and up-to-date information. To achieve this, the application employs a real-time data retrieval mechanism, actively fetching information from the Google Maps API to reflect the current traffic conditions. Frequent updates further guarantee that the displayed road conditions remain synchronized with the dynamic nature of traffic scenarios, minimizing the risk of presenting outdated information to users.

To make sure the app can handle many users without slowing down or crashing, we conduct load testing. This means we simulate different levels of people using the app to see how well it copes. By doing this, we can figure out if the system can handle the expected number of users smoothly and identify any areas that might cause problems or slow things down. It's like checking if the app stays fast and reliable even when a lot of people are using it at the same time.

Navigating challenges:

As I delved into the development work, integrating the Google Maps API with the application and with the DB emerged as a significant challenge. Ensuring a seamless connection between the application the DB and the API posed complexities, particularly in fetching real-time traffic conditions, displaying geographical maps, and efficiently updating road conditions for users. A solution approach for this is the regular updates and monitoring of the API's performance played a pivotal role in maintaining a reliable connection between the application and the Google Maps API. This proactive approach allowed me to identify and address issues promptly, ensuring a consistently smooth user experience.

While immersed in development, ensuring accurate data traversal between different stages, especially from user input to Firebase and vice versa, emerged as a crucial challenge. The integrity of data formatting for Firebase played a central role in guaranteeing a seamless exchange of information within the application's ecosystem. Implementing clear data formatting standards and error-handling mechanisms became essential components of the solution. Any discrepancies or errors in data formatting were identified and rectified promptly. Documentation detailing the expected data formats and exchange processes played a key role in guiding the development process, fostering consistency, and minimizing potential pitfalls.