AUST Junction

Group: 1 (B2)

Group Members:

Member 1: 17.01.04.082
Member 2: 17.01.04.086

Project Overview:

Aust Junction is composed of a backend that is responsible for storing AUST bus locations,previous year study materials , sending notification about bus and new notice on aust.edu website.,and a frontend that allows the user to query their bus locations,query their desired study materials etc.

Core Features:

1. User authentication through mobile number.
2. User Profile
3. Bus schedule and routes
4. Real-time locations of all of the Bus of AUST.
5. Easy query of previous semester lectures and questions of all departments of AUST.
6. Request section for location contribution or lecture contribution.
7. Email notification of bus locations after a particular time.
8. Email notification of new notice on official website of AUST

Technology Stack:

Client side language & framework - HTML,CSS,Bootstrap,Javascript

Server side application language & framework - Javascript & Nodejs

Database for the server - Firebase cloud firestore

Hosting for the server - Heroku

Client Side

index.js:

1)signoutChecker(): Used to check if a user is signed out or not from the website.
2) setUserDetailsOnHome(): Used to set user details on the home page if the user is logged in.

profile.js:

1)setUserDetailsOnProfile(): Used to set user details on profile page.

loginStateChecker.js:

1)checkLogin(): Used to check if the user logged in or not.

authentication.js:

1)sendCode(): Used to send code in the user’s phone no by which the user tried to sign in.
2)requestStoreUserData(): Used to send requests to store data.

MapController.js

/*
This class used to do mapbox related tasks like marker update,user location marker update
*/

class MapController{}:

1) getUserPosition(): This method used to track user location.

2) listenForBusLocationChanges(): used to set and update the bus marker position on the mapbox layout

locationThread.js*:*

initLocationFetchThread(): used to start a background thread that will fetch bus location from server.

worker.onmessage() : EventListener method that is used to receive messages from worker thread to main thread.

Web-work.js:

self.onmessage(): EventListener method that is used to receive messages from the main thread to the worker thread.

requestFetchLocationData(): this method used to send requests to fetch real-time location of the selected university bus. this method will be called from an event emitter that will be triggered after entering into the Mapbox interface.

HttpClientService.js

1)requestStoreUserData(): Used to send requests to store user data while signing up.
2)requestEditUserData(): Used to send requests to edit the existing user data from the Profile page.
3)requestFetchUserData(): Used to send requests to read a particular user data. It will be called when no user data is available in the browser's local cache.
4)requestStoreContactData(): Used to send requests to store contact data to the server.
5)requestFetchLectures(): Used to send requests to fetch all the lectures according to the selected department & semester.

Lecture1.html

1)storeSemester(): Used to store the selected semester in the local storage api.

2)storeDepartment(): Used to store the selected department in the local storage api.

Server Side

Model:

This class used to store user data to server,fetch user data from the server and handle coin transactions with server and database.

Class AuthenticaltionModel{}

```
      //this variable used to hold the information of all the users
including coin
       AllUsersData={};
```

storeUserData(): used for storing user data while signing up to the server and database.

editUserData(): used for storing edited user data to the server and database.

readUserDataFromDb(): For loading all users data to server from cloud firestore on start of a server

readUserData() : Used to fetch a particular user data from server

readAllUserData(): used to fetch all user data from server

BusLocationData Module

```
    //this json object variable holds the information of all the university bus
      var BusLocationData={};
```

storeBusLocationToDb(): used to backup all bus location information in cloud firestore

fetchAllBusLocationInfo() : used to fetch all bus location information from the server

fetchBusLocationInfo(): used to fetch location ,address,velocity and other information of a particular bus from the server in real time.

storeBusLocationData(): used to update the location information of all the university buses in real time .

/*This class used to handle the data that are coming from the contact section.
*/

Class ContactModel{}

```
        AllContactData{}: this variable used to store all contact information of all users
```

storeContactDetails() : used to store contact information to the server variables

storeContactDetailsToDb(): used to store contact data to firestore

//this class used to send bus location status through email notification to the students

Class EmailNotifier{}

monitorBus():
used to continuously check if a bus has arrived/left the varsity and notify the students based on it.

sendEmailNotification(): used to send email notification to the signed up users

checkUniversityRadius(): it checks the distance between varsity and bus using haversine formula.

/* This class handles lecture related queries with backend and database */

class LectureModel{}

fetchAllLecturesFromDb(): used to fetch all the data of lectures of the varsity

fetchAllLectures(dept,semester):used to query the lectures of all sessions of particular department and semester.

```
/**
 *  A web scrapping bot  will be implemented in this module that will listen for new
 *  notice board update in http://aust.edu/news_events.htm page
 */
```

NoticeBoardListener module

listenNoticeUpdate(): used to notify the students about the new notice on the official website of AUST.

Members Contribution:

ID: 170104086 contributed all the client side JS modules and design,DOM manipulation in HTML except the mapbox and thread related tasks.

ID: 170104082 contributed all the server side JS modules,API call and mapbox , thread related tasks on the client side.Also, contributed in database design.