Github link- https://github.com/tarun8334/mess_portal
Loom video link - https://drive.google.com/drive/folders/1SIHFCAk3oywKBnW-WguAXl37PMTP6rnt?usp=sharing

Title: "Mess Registration Reminder System"

Introduction:

In this project, my aim is to address the common issue of students forgetting which mess they've registered for, especially when multiple messes exist within a college. To mitigate this problem, I have developed a portal that enables students to effortlessly select their preferred mess for breakfast, lunch, snacks, and dinner. The system provides timely reminders, sending notifications **30 minutes** prior to the scheduled mealtime. These reminders include vital details such as the mess name and the day's menu.

Moreover, I've incorporated a feature that alerts students in the morning if they haven't chosen a mess for the day, ensuring that no student misses out on their meals. Additionally, I offer a client-side interface for mess administrators to communicate important messages to registered students. This includes notifications of any delays in the mess starting time or information about special menu items for the day.

Project Overview

Problem statement: Harness your creativity to craft a unique product by leveraging the capabilities of Plivo APIs.

Main Features and Functionality

1. User Registration and Login:
- Users can create accounts and log in securely to access the system through **Google sign** in.

2. Meal Selection:
- Registered users can select their preferred mess for breakfast, lunch, snacks, and dinner.

3. Scheduled Reminders:
- Users receive automated reminders 30 minutes before their chosen mealtime, containing the mess name and the day's menu.

4. Missed Meal Alerts:
- If a user fails to select a mess for a specific day, the system sends an SMS alert in the morning, notifying them that no mess is registered for the day.

5. Admin Messaging System:
- Mess administrators have the ability to send important messages to registered students.
- Messages may include information about delays in meal service, special menu items, or other relevant updates.

6. Interactive User Dashboard:
- Users have access to a user-friendly dashboard where they can manage their mess selections, view upcoming meal reminders, and check for messages from administrators.

7. Meal History Tracking:
- Users can review their meal selection history, helping them keep track of their dining choices.

8. SMS Integration:
- The system utilizes SMS alerts to ensure that users are notified even if they don't have access to the internet.

9. Menu Display:
- The system displays the daily menu for each mess, helping users make informed meal selections.

10. Mobile Responsiveness:
- The user interface is optimized for mobile devices, making it convenient for students to use on smartphones and tablets.

12. Security and Authentication:
- Robust security measures are in place to protect user data and ensure secure login and registration processes.

These features and functionalities collectively aim to simplify the process of managing mess registrations, enhance communication between students and mess administrators, and reduce the likelihood of missed meals or mealtime confusion.

Technologies Used

Javascript, Nodejs, HTML, Bootstrap CSS, Firebase, **Plivo Apis**

Project Setup:

Follow these steps to set up and run the project on your local machine:

- Clone the Repository:
  Start by cloning the project repository to your local system using the following command:

  git clone https://github.com/tarun8334/mess_portal.git

- Install Node Modules:
  Navigate to the project's root directory and install the necessary Node.js modules by running:

  npm install

- Navigate to the Backend Folder:
  Change your working directory to the backend folder using:

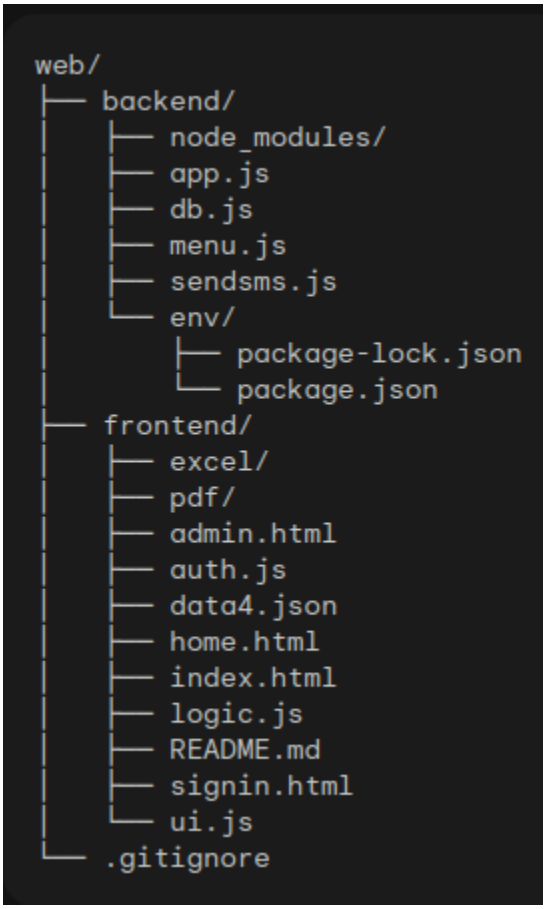
  cd backend

- Start the Backend Server:
  Once the dependencies are installed, start the backend server with:
  npm run start

- This command will launch the Node.js server, allowing the frontend to interact with the backend APIs.

  By following these steps, you'll have both the backend and frontend of the project up and running on your local machine, ready for testing and development.

Folder Structure:



The `backend/` folder contains the Node.js server code, while the `frontend/` folder contains the React code. The `.gitignore` file tells Git which files to ignore when committing changes.

Here is a brief description of each folder:

- `backend/`: This folder contains the Node.js server code. The `node_modules/` folder contains all of the third-party dependencies that are needed to run the server. The `app.js` file is the main entry point for the server.
- `frontend/`: This folder contains the React/HTML code. The `excel/` and `pdf/` folders contain any Excel or PDF files that are used by the frontend. The

`admin.html`, `auth.js`, `data4.json`, `home.html`, `index.html`, `logic.js`, `README.md`, `signin.html`, and `ui.js` files are all part of the React code.

- `.gitignore`: This file tells Git which files to ignore when committing changes. This is useful for preventing unnecessary files from being committed to the repository.

Database Schema:

Firebase Database Structure

The Firebase database for this project is structured as follows:

```
JSON
{
  "users": {
    "[user_id]": {
      "alerts": boolean,
      "favorites": [object, object, ...]
      "Phone no": phone_no,
    }
  }
}
```

The `user's` node is the root node of the database. It contains a child node for each user in the system. The `[user_id]` node is the child node for the user with the ID `user_id`. This node contains two child nodes: `alerts` and `favorites`.

The `alerts` node stores a boolean value indicating whether the user wants to receive alerts. The `favorites` node stores an array of objects, each of which represents a favorite item.

Example

The following is an example of a valid Firebase database structure for a user with the ID `xavSbqRGAN5EqBgsCLKXaAK7XF2`:

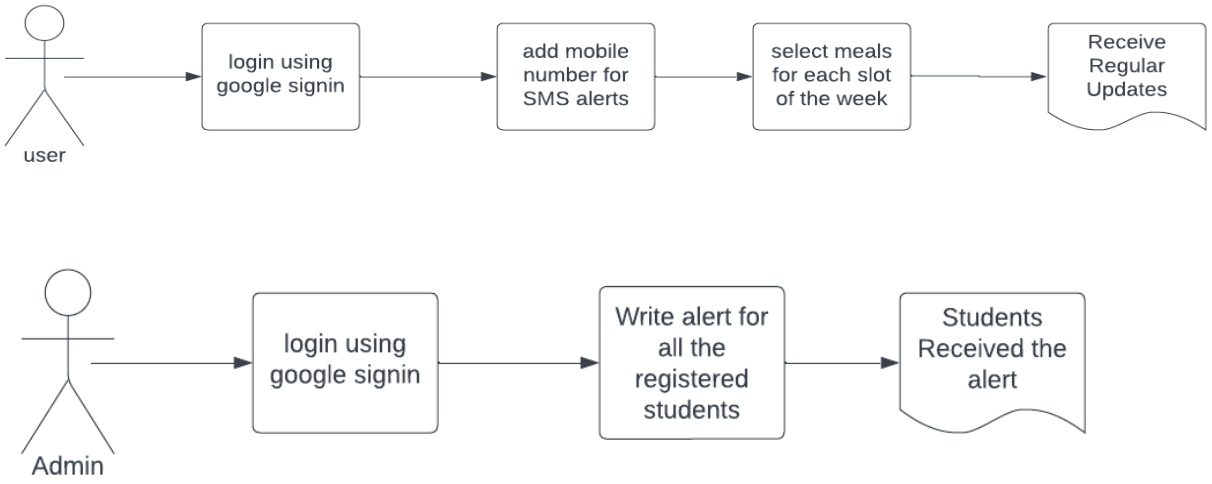
JSON

```
{
  "users": {
    "xavSbqRGAN5EqBgsCLKXaAK7XF2": {
      "alerts": true,
      "favorites": [“Sunday > North > Dinner Sunday > South > Snacks Sunday > South > Lunch”]

       "phone": "+919812160171"


    }
  }
}
```

Flow Diagram



Future Improvements:

- Admin portal can have more features like sending images of any special dish added in today's menu.
- If the mess system is designed for schools then SMS alerts option for parents can also be added so that admin can send SMS to parents if the student missed the meal.
- Calling features can also be added if a student wishes to avail reminder by call.
- Feature of receiving return message from student if they don't wish to avail the meal.

Challenges Faced:

- Choosing the Database.
- Testing the functionality as the SMS were scheduled for a fixed timings.

Sample messages:

Contact Information:

tarun.jindal@students.iiit.ac.in
9812160171