PROJECT NAME

RoutinePlus

Team Members

Arman Hossain Dipu (190042105)
Raiyan Noor (190042116)
Tasfia Tahsin (190042121)
Azmayen Fayek Sabil (190042122)

==========

INTRODUCTION

This document details the project plan for the development of “Routine-Plus”.
It is intended for developers, designers, and testers working on “Routine-Plus” as well as project investors. This plan will include a summary of:

- The user base.
- How the system will function.
- the scope of the project from the development viewpoint
- the technology used to develop the project, and the metrics used to determine the project’s progress.
- Overall Description.

Now that educational institutes are starting to open once again, students are having to become more active. They are required to attend early morning classes, classes in multiple classrooms and buildings. That’s why they have to plan their day according to the routine. So a change in routine can be hazardous to a planned routine school day. If the students aren’t properly made aware of schedule changes they might fall into misunderstanding and end up wasting a lot of time and energy. “Routine Plus” will solve this problem by multiple features like dynamic routine, change notification, classworks, announcements and student chat.

Targeted Audience:

The targeted customer will be an educational institution of any level which deals with scheduling classes and other academic events.
The end users will be the students, teachers and administrative personnel of the educational institution.

Functionality:

- End users should be able to sign up with categorised accounts.
- Users should be able to get different features, interfaces and access based on their category.
- The application should dynamically update changes in the routine made by permitted users.
- The application should send reminders and notification about any change to concerned end users.
- Users should be able to create ad hoc chat groups compromising sets of users and send.

PLATFORM:

The application will be developed in React Native to enable the creation of a web-based application, an iOS mobile app, and an Android mobile app.

The server side will use Node and Express js to carry out functionalities and will retrieve data from MongoDB database.

Development Responsibilities

The developers on the “Routine-Plus” team will be responsible for writing all the code for the application, developing the database, and managing releases.

User Class and Characteristics:

Primarily there will be three classes of users, Student, Teacher and Administrative officer. The Student class will have 2 different sub-classes, General student and Class representative.

The Teacher class will have the following access:
- Add or remove own class from the routine.
- Insert class in a vacant slot in the routine.
- Ask for permission from other teachers to swap their own class with another class.

The Student class will have the following features:
- View routine online.
- Receive notification related to his classes and courses.
- Group chat with teachers and other students.

The Class Representative sub-class will have the following additional accesses:
- Send requests for any change in the routine.
- Make changes in the routine with the permission of concerned teachers.
- Have direct private messaging options with teachers and administrative personnel.
- Permission to post notices.

The administrative Officer Class will have the following features:
- Oversee all the routines of different departments and sections.
- Post notices and announcements.
- Private messages with Student (Class Representative) and teacher classes.

SYSTEM FEATURES:

Functional Requirements:

- Users should be able to create ad hoc chat groups comprising sets of users and send messages to each other.
- Users will get push notification about the classes and more.
- Certain groups of users or admin panel can update class routines.
- Users can create accounts and enroll into groups.

User Interfaces

- Front-end software: React Native
- Back-end software: Node JS
- Database software: MongoDB

Hardware Interfaces

- Both Mac and Windows operating systems through their default web browser
- Android

Performance Requirements

- The application should load and be usable within 3 seconds
- The application should update the interface on interaction within 2 seconds
- The database should be normalized to prevent redundant data and improve performance
- The database should be distributed to prevent outages

Safety Requirements

- Databases should use sharding to be redundant to prevent loss of data.
- Backups of the databases should be done hourly and be kept for one week.

Software Quality Attributes

- Availability: Because this application is critical to business communication, we will have a goal of four nines (99.99%) availability.
- Correctness: The application should never allow anyone to read messages or discussions not intended for that person and also be allowed to change anything without admin's approval.
- Maintainability: The application should use continuous integration so that features and bug fixes can be deployed quickly without downtime.
- Usability: The interface should be easy to learn without a tutorial and allow users to accomplish their goals without errors.

==========