Journal of Electrical Engineering, Electronics, Control and Computer Science – JEEECCS, Volume 10, Issue 36, pages 9-18, 2024

A Lecture Reminder System for Enhancing Students' Engagement

1st Usiobaifo Agharese Rosemary
Department of Computer Science, Faculty of Physical Sciences
University of Benin, P.M.B 1154,
Benin City, Nigeria.
rosemary.usiobaifo@uniben.edu

2nd Osaseri Oghogho Roseline
Department of Computer Science, Faculty of Physical Sciences
University of Benin, P.M.B 1154,
Benin City, Nigeria.
roseline.osaseri@uniben.edu

Abstract – **Students often face challenges in remembering the stipulated time or schedule of lectures due to one reason or the other, which often results in missed classes and falling behind on the material covered by the lecturer. Hence, the need to design a lecture reminder system to mitigate this issue by providing notifications to students and lecturers about upcoming lectures. The Lecture Reminder System is a mobile application developed on smartphones for lecturers and 400level students of Computer Science, University of Benin, Benin City, with the service of reminding them about their lectures. In this paper, the major development tools used are JavaScript, ReactNative and NoSQL, creating a robust and adaptable platform.**

Keywords- undergraduate; web development; students' engagement; academic performance

I. INTRODUCTION

In today's fast-paced academic environment, students often struggle to manage their academic schedules and personal responsibilities. This can lead to missed lectures and decreased engagement, which ultimately affects academic performance. To address this challenge, we propose a Lecture Reminder System designed to enhance student engagement within the university.

As the current lifestyle is getting busier, people find it difficult to keep track of the many deadlines and due dates that they need to meet in daily life (Adibah, 2010). Modern mobile gadget such as smartphones have become increasingly powerful and also part of day-to-day lives of the people. As these devices become more like personal computers, they will come to replace objects we carry around such as planners, to-do lists aiding us in completing our daily activities (Divya et al, 2021).

This system is an innovative solution that keeps that keeps students informed and organized by delivering timely notifications about upcoming lectures. By leveraging automated reminders through mobile app notifications, the system ensures that students are always aware of their lecture schedules, enabling them to plan their time effectively.

The system's primary objective is to reduce missed lectures and encourage regular attendance, thereby promoting a more engaged and proactive student community. It also offers features like personalized reminders, integration with academic calendars, and ability to set customizable alerts for specific courses or events making it a versatile tool that caters to individual student needs. By improving communication between the university and its students, the lecture reminder system not only enhances student engagement but also supports academic success by minimizing disruptions and keeping students on track.

A Lecture Reminder System is a type of time management application or software that is designed to notify the user of important events that they have input to the program. Majority of programs offer a calendar or list of views for events, along with a reminder technique.

II. RELATED WORK

[1] developed a mobile-based academic reminder application (MARA) to assist users, and focuses on students in Emilio Aguinaldo College-Manila in managing their class schedules and assignment deadlines conveniently on their Android smartphones. Once users log into the system, they only need to update their schedules into the system and list any important date for their assignment deadline to the system, and the app will send alerts to remind them of the upcoming class and task deadline.

[2] (2019) investigated the impact of such systems on undergraduate students and found a clear positive correlation. Their research indicated that students who actively utilized reminder systems achieved higher grades, suggesting that these tools contribute significantly to improving learning outcomes.

[3] (2020) built an Academic Reminder System that helps students in UiTM Tapah in managing their class schedules and assignment deadlines via their android smartphones. Users login to update their schedules into the system and list any important date of their assignment deadline to the system. Then, the system will give an alert to remind the student of the upcoming class and task deadline. The project was developed using the waterfall model of the System

Development Life Cycle (SDLC), consisting of five phases. The outcome of the discussion suggests that the Academic Reminder System receives positive responses through the usability testing it underwent. A total of 30 students completed the usability survey and 46.7% strongly agreed that this application satisfies them in terms of interface, usability, and performance. Future improvements include adding a sharing application tool in the system so that the user can simply share the application to their friends. Additionally, the application can engage with new technology, which is using near-field communication (NFC) to simplify login processes for students.

[4] developed an Android based student reminder system. The paper proposes the development of an android application designed to meet the needs of students. A number of features are available in the application with the goal of improving organizational and academic aspects. For example, it reminds students of the number of classes needed to achieve certain attendance criteria, notifies students about attendance percentages, and informs them of the deadline for renewing library books. Notifications regarding forthcoming events, including exams, workshops, and student activities, are also provided by the application. Notably, everyone can install and use this user-friendly application with simplicity because it works with any Android smartphone.

[5] (2017) investigated how digital reminder systems could facilitate the process of remembering in a more diverse range of situations. Time and location are often used as triggers by digital reminder systems to remind people to perform activities. They report findings from a survey and one-week diary research, which reveal that individuals want to remember to carry out a broad spectrum of activities in the future, many of which are too complex to be supported by simple time- and location-based reminders. Along with these examples of prospective memory, or "remembering intentions", they also discovered that people require assistance in "retrieving" information and details, particularly when those details are encountered in social interactions or are meant for use in discussions with others. They derived implications for the development of smart reminder systems such as Microsoft's Cortana and smart speaker systems like Amazon Echo based on their analysis of what people want to remember and how they attempt to support this. They also highlighted the possibilities afforded by drawing on conversation and giving digital reminders material form.

[6] (2016) introduces an activity reminder system that leverages time and location. According to their research, the rise of human activities has led to the development of an activity reminder system and presently, a mobile device has become a daily communication device. An activity reminder system that runs on a mobile device offers distinct advantages, eliminating the need for additional hardware. Mobile devices are accessible anytime and anywhere. The reminder system uses the venue data from platforms like Foursquare and Google Maps.

According to [7] (2014), modern individuals rely heavily on personal task reminders in order to remind them heavily of their tasks at specific circumstances. Traditional paper-based reminders are still useful, but they lack efficient organization. Electronic reminders based on the calendar in cell phones are increasingly popular due to their efficiency, but such reminders are mostly triggered by time. In many situations, tasks are only relevant when executed at particular locations, making it useful for reminders to activate solely when the person is in close proximity to that location. Therefore, in their research, they developed a location-based personal task reminder for android smartphones. In contrast to existing solutions that depend solely on GPS, they took advantage of the widespread IEEE 802.11 WLAN infrastructure to fill the gaps in GPS location sensing and ensure effectiveness both indoors and outdoors. Additionally, they proposed two operational models for the personal task reminder to enhance the application's usability. Furthermore, as long as the WLAN infrastructure is available, our work serves as a foundational platform for various location-based services, such as public transportation guidance, tourism, location-based learning, and even dementia care.

[8] (2022) conducted a research that is motivated by the large number of student activities that occasionally make students forget or overlook the tasks they have to complete on time. The purpose of their study is to design an android-based task reminder system that will enable students to be reminded about lecture assignments, the assignment collection deadline, and other information regarding lecture activities.

The proposed system surpasses the related works because it provides a specific and practical solution tailored to the challenges faced by students and lecturers in maintaining consistent attendance and engagement with scheduled lectures. Unlike the related works that might primarily focus on theoretical frameworks or generic reminder systems, this paper emphasizes a focused approach that directly addressed the gaps in student engagement through timely notifications and user-friendly interface.

Moreover, my work stands out by incorporating feedback from actual users - both students and lecturers – during development, ensuring that the system addresses their specific needs. Furthermore, the use of modern technologies like JavaScript, ReactNative and NoSQL ensures that the system is scalable, adaptable and accessible to different educational settings, making it a more robust and innovative contribution to the field.

III. System Analysis and Design

The system analysis phase delves into understanding the requirements and constraints of the Lecture Reminder System. This encompasses the collection of users' needs, identifying functionalities and analyzing the existing systems for potential improvements.

During the design phase, the focus shifts towards conceptualizing the architecture and layout of the lecture reminder system. This involves formulating high-level designs that illustrate system components, interactions and data flows. Additionally, detailed designs are developed, encompassing database schemas, user interface mockups, and system behavior diagrams. The objective of the design phase is to provide a clear blueprint for the implementation of the system.

A. Architecture of the Lecture Reminder System

The proposed Lecture Reminder System architecture in figure 1 composed of several interconnected layers that provide smooth operation for both students and lecturers. At the client layer, there are two different mobile applications- one for students and one for lecturers. The students' app allows them to sign up, add their courses, view their timetable, and receive notifications. The lecturers' app enables lecturers to manage lecture schedules by cancelling classes, with notifications sent to students whenever changes are made. The API Gateway serves as a middle layer, directing requests from mobile apps to the appropriate backend services. In the application layer, the user management module handles the student and lecturer registration and authentication. The scheduling module manages the class timetable, while the notification service is responsible for sending reminders and updates. The course management module stores course data for both lecturers and students.

When a user opens the app, they are approached with three different screens depending on their authentication state:

i) **SignIn -> Home**:
- When the user opens the app, they are directed to the SignIn screen.
- If the user successfully signs in with their credentials (email/password), they are redirected to the Home page.
- Firebase Authentication in figure 2 handles the sign-in process and persists the user's authentication token locally.

ii) **SignUp -> AddCourses -> Home**:
- If the user is new and doesn't have an account, they can choose to sign up.
- During the sign-up process, the user provides their email address, password, and username. This data is then stored in the users collection in firebase.
- After successful signup, they are directed to the Add Courses screen where they can add the courses they are offering for the semester.
- Once they finish adding courses, they are redirected to the Home page (figure 3).

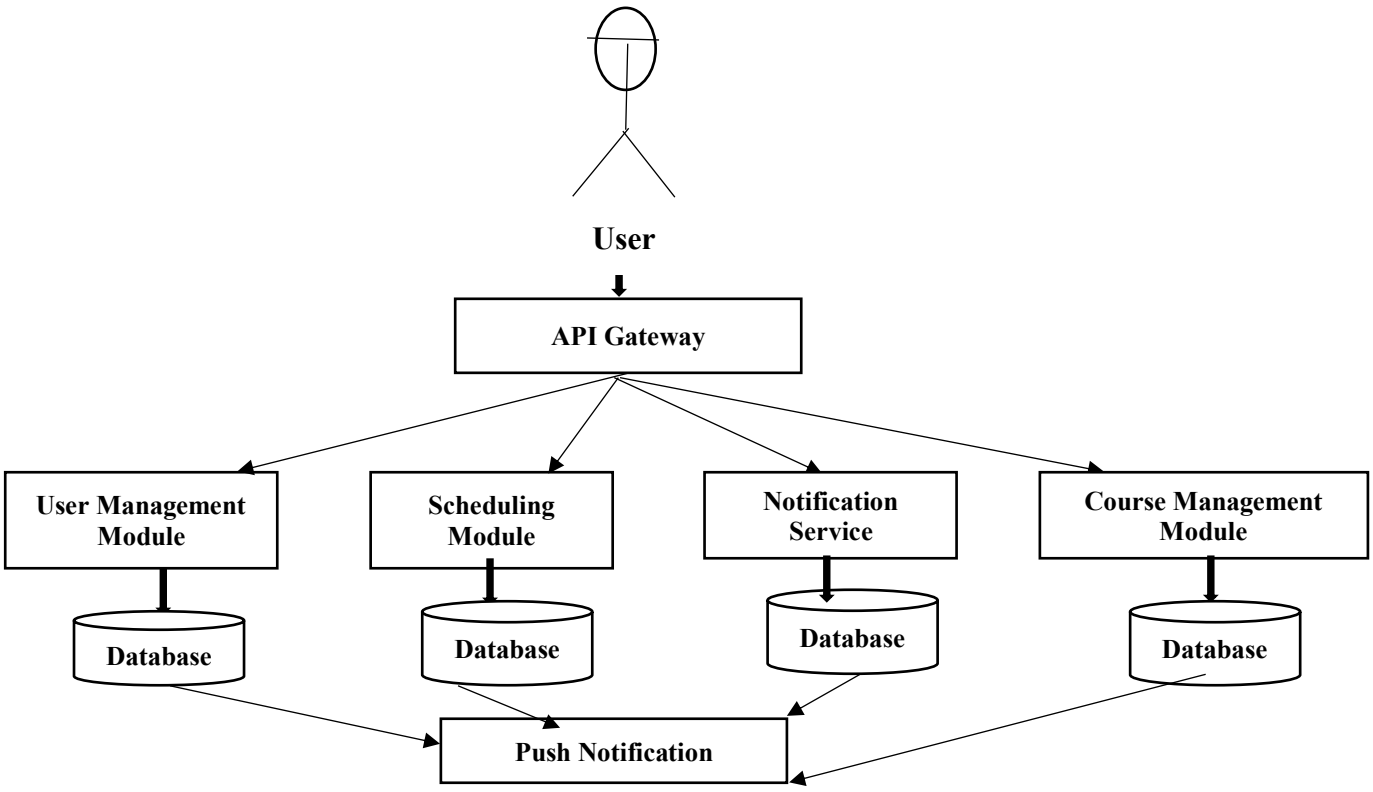

Fig 1: Architecture of a Lecture Reminder System

USER AUTHENTICATION (Sign In)

Enter Valid Email and Password	
Email	
Password	
Don't have an account?	**Sign up**
	SIGN IN

Fig 2: Sign In page

USER AUTHENTICATION (Sign UP)

Create New User Account	
User Name	
Email	
Password	
Already have an account?	**Sign in**
	SIGN UP

Fig 3: Sign Up Page

iii) **Home**:
- If the user is already signed in (authenticated), they are directly taken to the Home page when they open the app.
- Firebase Authentication in Figure 3 automatically persists the user's authentication state, so returning users don't need to sign in again.
- The Home page serves as the main dashboard or landing page for authenticated users. It contains the course dashboard to see courses holding for a given day and a timetable component to show the timetable

iv) **Logout**:
- On the Home page, there is a drawer option that allows the user to log out.
- When the user chooses to log out, Firebase Authentication handles the sign-out process.
- Upon successful logout, Firebase removes the user's authentication token from local storage, effectively logging them out of the app.
- After logging out, the user may be redirected back to the Sign In screen, depending on the app's design.

B. Architecture of the Lecture Reminder System

A use case diagram in a Lecture Reminder System visually represents the interactions between users and the system itself. It shows the functionalities (use cases) that the system provides and the users who perform them. The diagrams control primary actors and use cases. Actors are entities (students and lecturers) that will interact with the application while use cases are functions that actors can perform as illustrated in the diagrams below (figures 4 – 6).

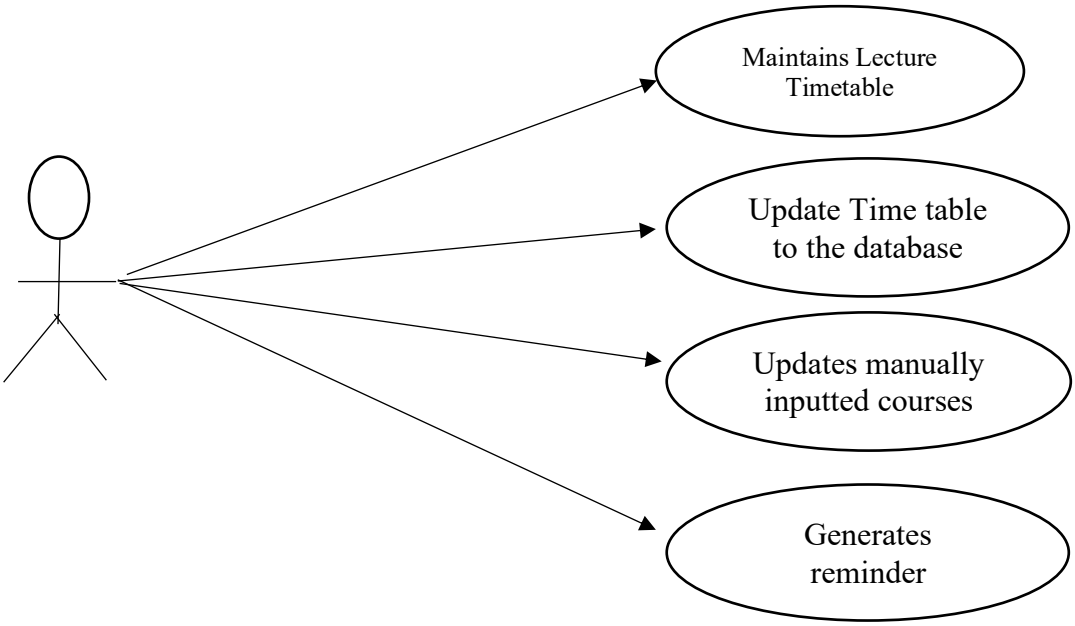

Fig 4: Time Table Entry Operator

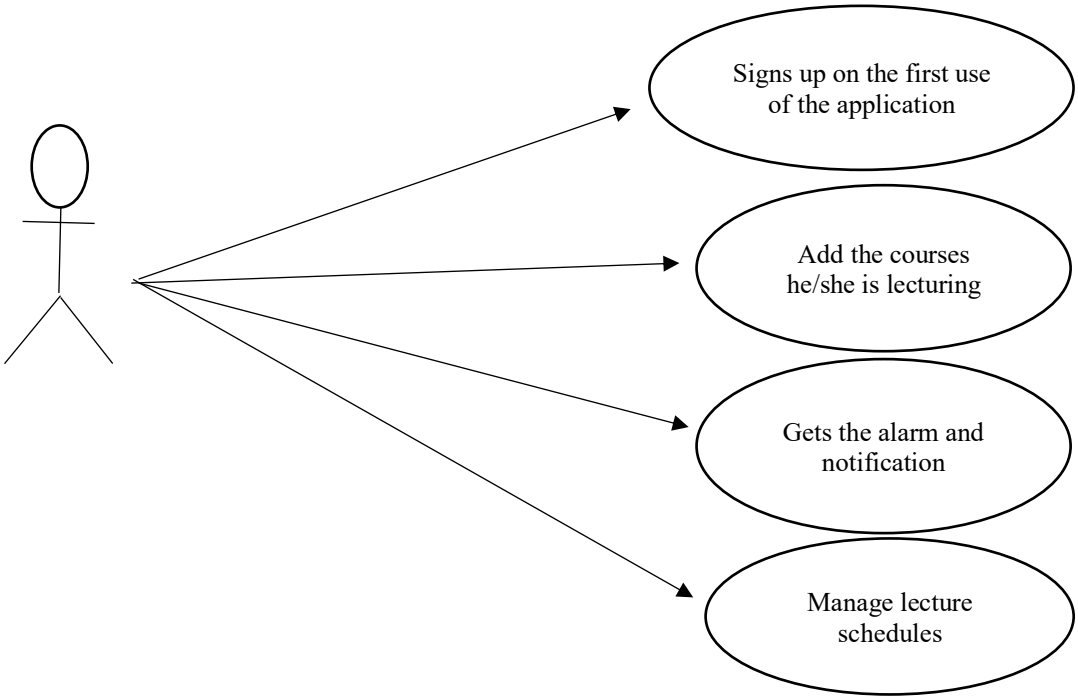

Fig. 5: Lecturer or staff

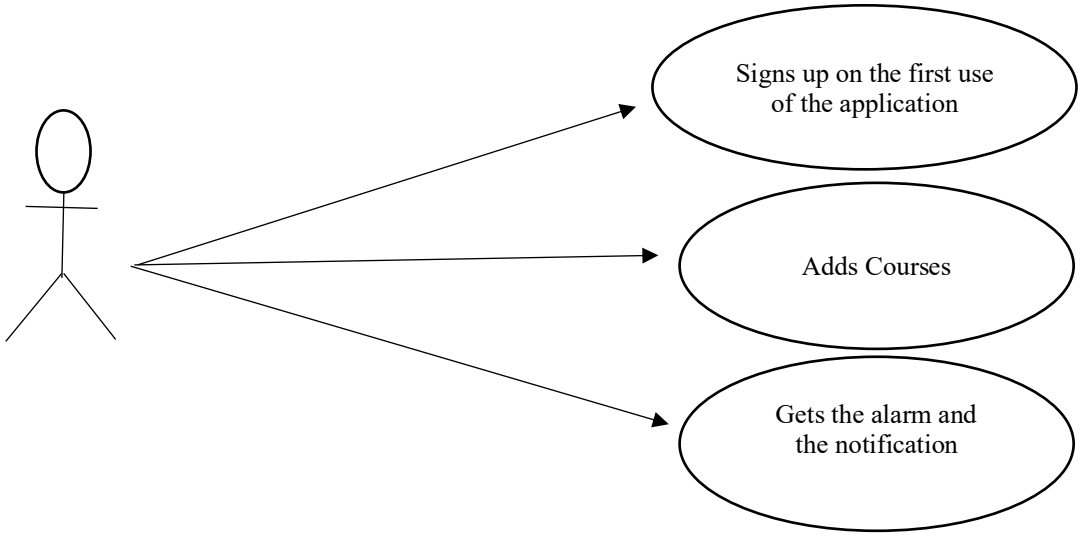

Fig. 6: Student

IV. Implementation

In this paper, JavaScript was the programming language used, the framework used was ReactNative (ReactNative and NoSQL ensures that the system is scalable, adaptable and accessible to different educational settings, making it a more robust and innovative contribution to the field).

The implementation platform being used was Visual Studio Code which is an outstanding choice for building web applications. Firebase was developed for making the web application available to all users. The firebase is used for backend and it handles the authentication and push notifications. The database makes use of firestore backed by firebase which is a NoSQL database.

NoSQL allows for flexible data modeling unlike traditional SQL databases. SQL databases are best used for structured data, whereas NoSQL are suitable for structured, semi-structured and unstructured data. Consequently, unlike SQL databases that adhere to a fixed schema, NoSQL databases don't follow a rigid schema but instead have more flexible structures to accommodate their data-types.

The following figures (figures 7 – 11) are the different screens of the Lecture Timetable Reminder System. At launching the application as shown in figure 7 is the image of the home screen of the application where new users can sign up for the application and the user must be connected to the internet at this point. The next screenshot (figure 8), shows the user adding or selecting the courses he or she is offering by clicking on "Select Course". On this

page, users are allowed to add multiple courses at a time. The course information which includes the course title, location, date and time is depicted in figure 9. Figure 10 shows the student's home screen immediately after the student successfully sign up. It shows the timetable, the number of courses he or she have for that day and it also tells if a class has been held. Figure 11 shows the list of course codes, course title, venue and the date and time it holds every week. Figure 12 shows the home screen of the lecturer's app immediately after signing up. It shows the timetable and the button where he or she clicks to cancel a course that was scheduled to hold at a particular time due to one reason or the other and thereafter it notifies the students that the course scheduled for that time has been cancelled by the course lecturer. Figure 13 shows the notifications received. The notification uses the default sound of the mobile phone and the notification comes few minutes before the class starts.

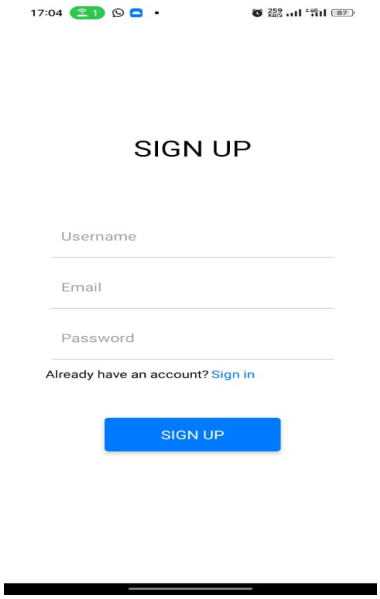

Figure 7: Sign up page

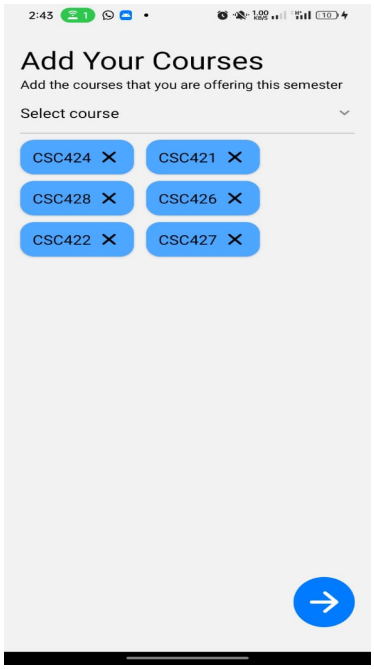

Figure 8: User adds courses

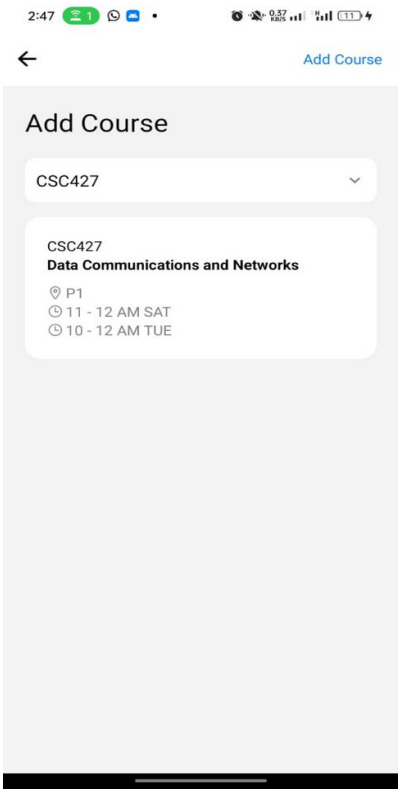

Figure 9: Course Information

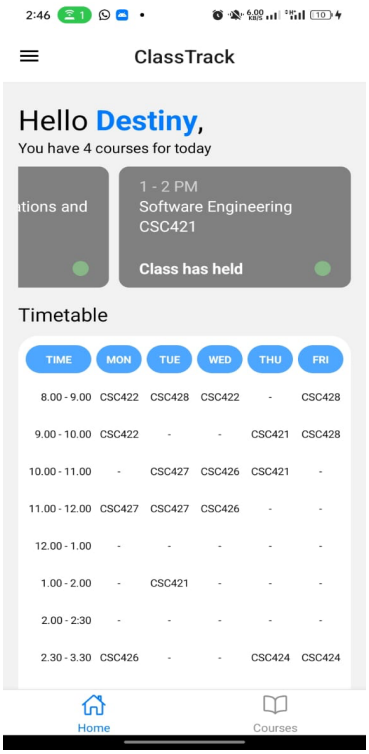

Figure 10: Student home screen

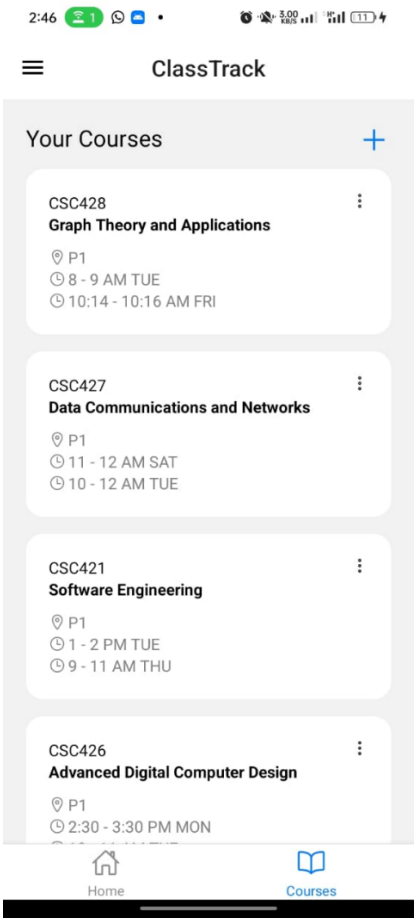

Figure 11: Course list

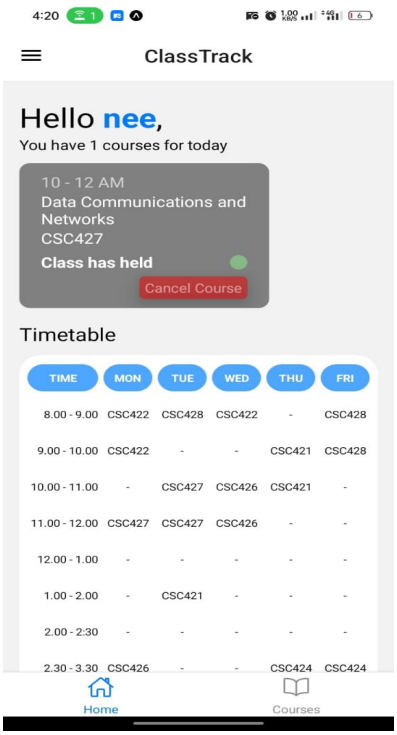

Figure 12: Lecturer home screen

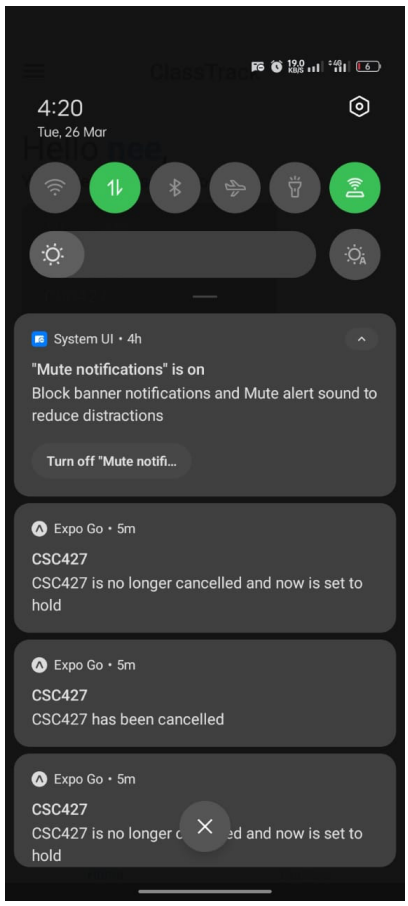

Figure 13: Timely Notifications

Database Schema

Firestore does not enforce a rigid schema but for the sake of documentation, here is an overview of the schemas used to model a user's data:

i) Users collection:

```
{
 "users": [
   {
    "id": "string",
    "email": "string",
    "displayName": "string",
    "password": "string",
    "role": "string",
    "expoPushToken": "string"
   }
 ]
}
```

The user's collection contains information about the user such as their id, email, displayName, password and role. The id is randomly generated. Their passwords are hashed in the database. The role can either be a "LECTURER" or "STUDENT" which specifies the permissions for the users. The expoPushToken is used to send push notifications to the user's device.

ii) Courses collection:

```
{
 "courses": [
  {
   "id": "string",
   "venue": "string",
   "schedule": [
    { "day": "string", "time": "string", cancelled: “boolean” }
   ]
  }
 ]
}
```

The courses collection contains information about courses such as their id (which in this case is the name of the course such as CSC422). The venue which is where the course would hold. The schedule which is which day and for what time period the course would hold. Each schedule also has a cancelled field indicating whether the course would hold for a given week.

iii) User_Courses collection:

```
{
 "user_courses": [
  {
   "id": "string",
   "user_id": "string",
   "course_id": "string"
  }
 ]
}
```

Despite firestore being a NoSql database, we can still benefit from normalization techniques used in traditional SQL databases to reduce redundancy in the data. This allows us to make a separate collection to store the courses a user is offering.

Notification System

The app makes use of expo-notification library which simplifies the implementation of push notifications. Expo Notifications provides a unified approach to handling notifications on both iOS and Android platforms.

This app implements two types of notifications:

1. Local notifications

They are generated and scheduled directly on the user's device by the mobile app itself. They can be used to remind users about tasks, events, or other activities based on specific triggers or conditions set within the app. Unlike push notifications, local notifications are generated and displayed entirely on the device without requiring communication with a server.

The local notifications system schedules courses to the end of a given week. The notifications are triggered when a course is about to start, has started, and has ended.

2. Push notifications

They are notifications remotely sent to the users; they are received by the app, triggering a local notification that is displayed to the user.

The push notifications system in this project can be summarized by the steps given below:

i) Get Push token: When a user installs the app and opts in to receive push notifications, Expo generates a unique Expo Push Token for that device. This push token is then stored in the firestore database in the user collection.

ii) Send Notifications: The notifications are sent using the expo server sdk which is hosted using firebase cloud functions. Expo server handles the communication with FCM (notification system for android) and APNs (notification system for apple) on your behalf. When you send a push notification using the Expo Server API, Expo translates your request into the appropriate format for FCM or APNs based on the recipient's device platform. You can customize the content of your push notifications, including the title, body, and any additional data you want to send along with the notification. An example message object is given below:

```
const message = {
to:
'ExponentPushToken[xxxxxxxxxxxxxxxxxxxxxx]',
sound: 'default',
title: 'New Notification',
body: 'This is a test notification from Expo server SDK',
};
```

iii) Receive and Manage notifications: The devices then receive the notifications, which are presented as local notifications to the user. The user can interact with these notifications in the notification drawer, which can subsequently open the app.

Home Page

The UI displays information to the user about which courses are to be held for a given day. It hooks to notifications triggered by the expo-notification library to update the UI. A course can be in two states:

i) NOT_CANCELLED: A course in the NOT_CANCELLED state can be in three other sub states:

- NOT_HELD
- HOLDING
- HELD

ii) CANCELLED

CONCLUSION

In conclusion, the development of the ClassTrack lecture reminder app is of importance to the school. By providing timely reminders about lectures and schedule changes, ClassTrack aims to reduce the frequency of missed classes and boost overall student engagement. The app's features, including personalized notifications and seamless integration with lecture timetable, offer a tailored solution that addresses the specific needs of students navigating their university journey. The ClassTrack app also encourages a habit of regular attendance and active engagement in learning.

After the full implementation of the system using 400 level, Computer Science Department as a case study, it is therefore recommended that the application can be adopted by all other departments of the University of Benin and all other higher institutions around the world so as to reduce the rate at which students and lecturers forget the time of their various lectures.

REFERENCES

[1] Acuna,A.A, Sabilli M.A.P and Friginal, F.F.S(2024). MARA: AMobile-based Academic Reminder Application. International Journal of Computing Sciences Research, 8, 2734-2748. http://doi-org/10.25147//ijcsr.2017.001.184.

[2] Aung, M. (2019). Mobile Academic Reminder Systems and Academic performance: A correlation Analysis. International Journal of Educational Technology, 34(2), 123-140

[3] Nik Ruslawati Nik Mustapa, Siti Hajar Natasha Mustapa (2020). "Design and development of mobile application for academic reminder system". Journal of Computing Research and Innovation (JCRINN) 5 (4), 18-26.

[4] Y Bhavani, D Sanjay (2017). "Android based Student Reminder System". Oriental Journal of Computer Science and Technology 10 (4), 760-764.

[5] Robin N Brewer, Meredith Ringel Morris, Siân E Lindley (2017). "How to remember what to remember: exploring possibilities for digital reminder systems". Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies 1 (3), 1-20.

[6] Nur Rokhman, Lubab Saifuddin (2016). "Location and time-based reminder system on Android mobile device". 2016 2nd International Conference on Science in Information Technology (ICSITech), 147-151

[7] Kukuh Yulian Santoso, Taufiq Abidin, Slamet Wiyono (2022). "Mobile-based Assignment Reminder Application for Students and Lecturers". Journal of Machine Learning and Artificial Intelligence 1 (2), 167-172.

[8] Victoria Nkemijika Emmanuel, Bright Uzoma Eseoha, Victor Chibunna Enyinnaya. (2024)"Development and execution of a notification system for lecturers in universities". World Journal of Advanced Research and Reviews 23 (1), 1093-1098