TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

A
# FINAL REPORT
ON
IOE APP

**SUBMITTED BY:**

NISHA SHARMA (PUL075BCT056)
SAGAR TIMALSINA (PUL075BCT071)
SANDIP PURI (PUL075BCT077)
UDESHYA DHUNGANA (PUL075BCT095)

**SUBMITTED TO:**

DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING

May, 2023

# Page of Approval

TRIBHUVAN UNIVERSIY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS
DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

The undersigned certifies that they have read and recommended to the Institute of Engineering for acceptance of a project report entitled **"IOE App"** submitted by **Nisha Sharma**, **Sagar Timalsina**, **Sandip Puri**, **Udeshya Dhungana** in partial fulfilment of the requirements for the Bachelor's degree in Electronics & Computer Engineering.

.............................
Supervisor
**Prof. Dr. Sashidhar Ram Joshi**
Professor
Department of Electronics and Computer
Engineering,
Pulchowk Campus, IOE, TU.

.............................
Internal examiner

.............................
External examiner

Date of approval:

# Copyright

# Acknowledgments

The completion of this project owes its status to all the individuals and organizations who have provided their valuable guidance and support at different stages of the project. It is with great pleasure that we express our sincere gratitude to all those who have contributed towards the successful completion of our project.

Firstly, we would like to extend our heartfelt appreciation to the Department of Electronics and Computer Engineering at Pulchowk Campus for providing us with the opportunity to apply our knowledge in a real-world setting and for facilitating the development of our teamwork skills.

Additionally, we would like to express our deepest gratitude to our esteemed supervisor, Prof. Dr Shashidhar Ram Joshi, for his invaluable guidance and support throughout the project.

We would also like to acknowledge the support and mentorship provided by Rara Labs, and in particular, Prayash Koirala, Flutter developer at Rara Labs, for his insightful suggestions, constructive feedback and a generous allocation of time and resources towards the successful completion of our project.

We consider ourselves fortunate to have had such incredible support from all these individuals, and we sincerely thank them for their contributions towards our project.

Sincerely,
Nisha Sharma
Sagar Timalsina
Sandip Puri
Udeshya Dhungana

# Abstract

The Institute of Engineering (IOE) in Nepal is utilizing various platforms for providing services to its students and staff. IOE APP is being developed as a comprehensive platform with a user-friendly interface integrating all the services in a place. This platform aims to integrate existing services while providing additional features through the use of plugin-like technology. The IOE APP strives to provide seamless services to its users and enhance the efficiency and centralization of college activities. The Agile methodology approach has been adopted to ensure that the IOE APP meets the requirements of its target audience. Furthermore, the IOE APP's microservice architecture allows for scalability and extensibility, which can pave the way for similar applications to be developed for other organizations. This framework represents a shift away from manual-dependent systems and towards a completely application-based service.

Keywords: *IOE, Nepal, platform, Agile methodology, microservice architecture, scalability, EMS*

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **API** | Application Programming Interface |
| **App** | Application |
| **CDN** | Content Delivery Network |
| **CGPA** | Cumulative Grade Point Average |
| **CSS** | Cascading Style Sheets |
| **DOM** | Document Object Model |
| **HTTP** | Hyper Text Transfer Protocol |
| **ID** | Identity Document |
| **IOE** | Institute of Engineering |
| **JSON** | Javascript Object Notation |
| **JWT** | JSON Web Token |
| **MVC** | Model View Controller |
| **ORM** | Object Relational Mapping |
| **PHP** | Hypertext Preprocessor |
| **REST** | Representational State Transfer |
| **SDK** | Software Development Kit |
| **UMS** | University Management System |
| **UI** | User Interface |
| **URL** | Uniform Resource Locator |

# 1. Introduction

## 1.1 Background

Institute of Engineering (IOE) is the leading technical education institution in Nepal and offers undergraduate and graduate programs in various fields of engineering, such as civil engineering, mechanical engineering, electrical engineering, electronics and communication engineering, computer engineering, and architecture. It currently carries out administrative and academic tasks both offline and online. While some services have been shifted to online platforms, others such as attendance systems and semester result publication still to be shifted online. Moreover, there is a lack of a centralized website or application to access all IOE facilities. The Coronavirus pandemic has highlighted the need to expedite the process of moving services online. To address this, we propose the development of a mobile application that can integrate all IOE services and is flexible for future enhancements through plugin-like features. Our aim is to provide a convenient and efficient platform for IOE users while ensuring easy integration of existing offline processes. We anticipate that this application will be highly advantageous to the Institute of Engineering.

## 1.2 Problem statements

Some of the challenges faced in current IOE services are highlighted as:

- The current services of IOE are mostly manual and offline, leading to delays and inefficiencies.
- Online services such as entrance registration and library management are not integrated and are run separately.
- The current system lacks real-time updates and fails to provide timely and relevant information to students.
- A modernized system, such as a mobile app, is needed to provide a centralized platform for communication, access to course materials, and other relevant resources.

## 1.3 Objectives

Our project aims to develop an app that is easily scalable and integrable, allowing for easy access to all activities offered by IOE. The objectives of our project are highlighted below:

- To create an easily scalable and integrable IOE APP.
- To improve online access to the educational system and mitigate current problems by using microservice architecture.

## 1.4 Scope

The Institute of Engineering (IOE) currently lacks an application for conducting its activities or communicating with stakeholders. A proposed app aims to serve as an external interface for IOE and its users, facilitating activities such as result publication and online attendance. The app will be designed with a suitable architecture to integrate existing systems, enabling IOE to better serve its stakeholders by providing an efficient and effective way to access important information and engage with the institution.

## 1.5 Motivation

The idea for the IOE app was born out of the students' own need for a comprehensive and easily accessible platform for IOE-related information. With the transition to virtual classes due to COVID-19, this need became even more apparent. As students of IOE, our team behind the app project saw an opportunity to give back to the institution by proposing a solution that would benefit both the students and the institute. By creating an app that facilitates activities like online attendance and result publication, IOE can improve its services and communication with its stakeholders. Ultimately, the motivation behind the IOE app is to create a more seamless and efficient experience for students and other users, while also helping IOE to better manage its resources and serve its community.

# 2. Literature Review

Mobile application development has been one of the most emerging fields in IT in the last decade. With the growing field, various technologies have been studied and tested along with many frameworks being introduced time and again. Two widely used frameworks are React Native and Flutter -a widget-based system. Due to the standardization by Google and high security, Flutter can revolutionize React native design with simpler syntax neatness and SDK[1].

A system has a microservice architecture when that system is composed of many collaborating microservices; typically without centralized control [2]. Microservice architecture has been used for scalability, agility and reliability in various applications[3]. Since its introduction in a white paper by Martin Fowler and James Lewis it has since become a de-facto standard for developing large-scale applications.

## 2.1 Related work

Android App for Kathmandu University displays information about the university, university calendar, course registration, applying for transcripts, news, notice and student activities. It gives information about student welfare, the central library, publication and research of the university. This App brings all the major components of their website into a single dashboard with added benefits of Personalization.[4]

An application for Islamic University Gaza(Student Portal) displaying the courses' schedule and exams schedule for students from anywhere and anytime, also notifying the students of student lectures' schedule and exams automatically, viewing the academic information and grades report (marks transcript) for the students, providing silence schedule because most of the students forget their mobile phones in normal mode during the lectures, but by the intended application, mobile phones will be automatically switched to silent mode during the lectures. But this system depends on the university Web site directly[5].

An application for Stanford University, Stanford Mobile allows users to explore campus dining options, upcoming events, featured news stories, campus and shuttle maps etc. Stanford community members with a SUNet ID can log in and access the Stanford Health Check and Mobile ID features. Health Check gives you quick and convenient access to the self-reporting tool. Mobile ID displays the information that you find on your physical Stanford ID card: your name, photo, affiliation, University ID, and barcode number.

Jose De Souza created a mobile application, Gestation[6] using microservice architecture which is capable of meeting requests from pregnant women via mobile application and allowing health professionals to monitor the pregnancy evolution of a health unit. The architecture was designed to offer satisfactory performance, scalability and simplicity.

# 3. Related theory

## 3.1 Microservice architecture

Microservice architecture is a software development approach that involves building an application as a collection of small, independently deployable services. Every service has its own functionality which when combined makes a full application. Due to the features like scalability and flexibility, it has gained popularity in designing the complex system.

In a microservice architecture, each service is built independently that communicates with each other using the API calls or messaging system. This design known as modular design helps for easy maintenance, as each service can be updated, replaced or deleted without affecting the whole system. Also, the independent modules greatly add to the parallelism, improving productivity and allowing for faster development.

Microservice architecture support scalability. Services can be scaled up or down as needed as services are built independently without affecting other services and the rest of the system. This allows to better use of resources as only necessary systems are scaled up to handle the increase in demand. Furthermore, vertical scaling is also possible enabling service to scale up or down.

Another important benefit of microservice architecture is its flexibility. Since the services are built as an independent module, they can be built using any technology stack according to the need of not only the system but also the services. This means that different services can be built using different programming languages, databases, or frameworks. Furthermore, this flexibility enables developers to choose the best tools for each task or function, without being to choose a single stack for the whole system.

Microservice Architecture also makes the complex system resilient. As the services are independent modules, failure in service won't affect the rest of the system. Since a microservice has unique features it is easy to monitor which significantly helps to improve system performance and search for errors.

However, implementing microservice architecture also has its challenges. One of the main challenges is the complexity involved in designing and coordinating the interactions between services. This requires careful planning and design, as well as the implementation of robust testing and monitoring procedures. Additionally, because each service is responsible for a specific function, it can be difficult to ensure consistency across the entire system, leading to issues such as data inconsistencies or duplicate functionality.

## 3.2 HTTP

HTTP is a Hypertext Transfer Protocol which is used for the transfer of data between servers and browsers. It is the foundation of the World Wide Web. Being the primary way of transmitting and receiving data over the net, it is designed to be a simple and extensible protocol. HTTP is a request-response protocol referring to the client-originated data transfer. The client requests the data or information from the server and the server responds with the same. The request can be of many components with the HTTP method, the requested URL and any other additional header. Server with responses with data and predefined status code. HTTP is simple and extremely easy to use. Because it is a text-based protocol, it is understood easily. Additionally, its extensibility allows for the creation of new functionality or features without disrupting the existing protocol.

HTTP is a stateless protocol i.e. it doesn't store the state of the client in the server machine and responds only to the requested data. Since HTTP itself don't encrypt the data it relies on other higher-level protocol for encryption. Also, HTTPS was developed for the same purpose.

## 3.3 MVC Pattern

MVC Pattern is a software design pattern in which the applications are divided into three distinct components: model, view, and controller. It separates the program into the user interface, business logic and application data.

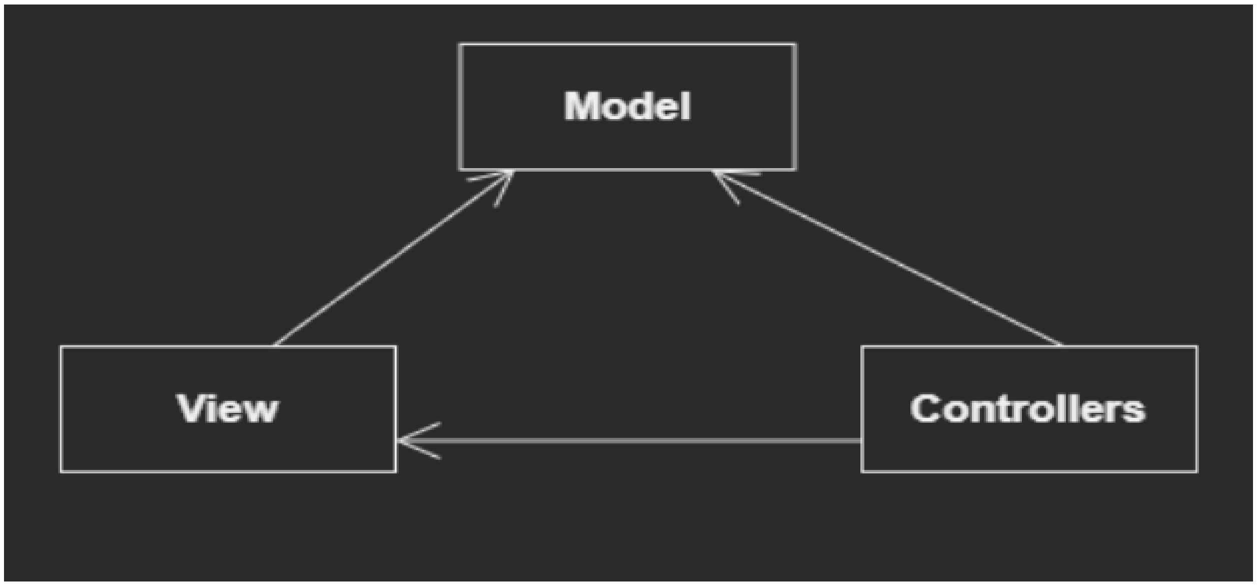


Figure 3.1: MVC pattern

The Model component consists of the data and business logic of the application. It contains the data and provides access to the database for the purpose of manipulating it. The model can be tested independently as it doesn't depend on the view or controller.

The view components are the way of presenting data to the user i.e. it consists of a user interface. It is a visual representation of the models' data on a platform like a web, mobile application or desktop application window. The views receive updates from the model.

The Controller component acts in the middle between the view and the model. It receives input from the user from the view and updates the model accordingly. It also updates the view based on changes in the model. It binds together the view and the model and eases data communication between these two.

MVC has many benefits. To include a few it separates the application's concerns into three distinct components, each component can be developed and tested independently of the others. This can lead to more modular and maintainable code and can make it easier to add new features or make changes to existing ones.

## 3.4 REST API

REST is an architectural approach for developing web services. A RESTful API (Application Programming Interface) is built using REST (Representational State Transfer)concepts. RESTful API transfers a representational state of the resource to the endpoint or requester. The information can be of any format like JSON, HTML, PHP, XML etc.

REST APIs are used for the data exchange between client and server. They communicate with the server via HTTP (Hypertext Transfer Protocol) techniques. GET, POST, PUT, PATCH, and DELETE are the most often used HTTP methods in RESTful APIs.

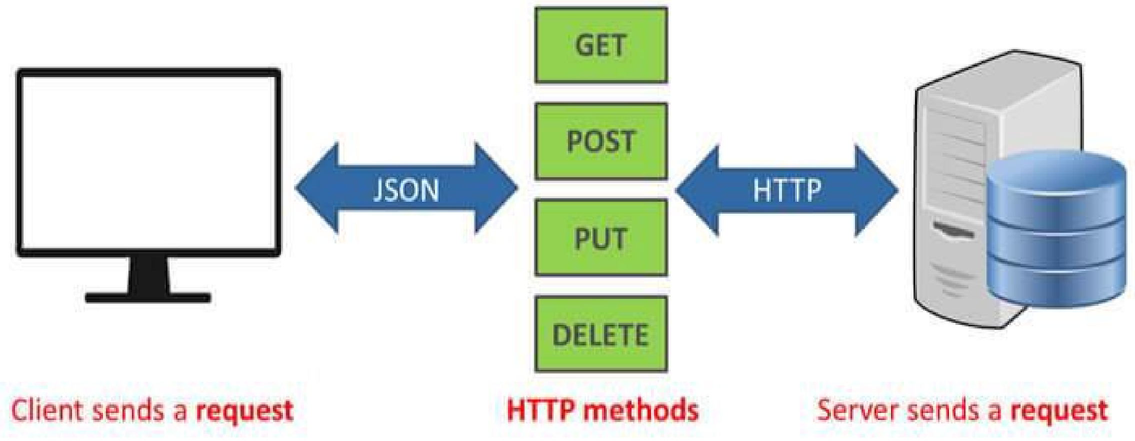


Figure 3.2: REST Api

### 3.4.1 GET

The GET method is used when data is needed to retrieve from the server. It reads the data from the server. It is used to retrieve data like user profiles, product information, or other resources.

### 3.4.2 POST

THE POST method is used to send the data to the server. It is useful for creating new resources to be retrieved later in the server.

### 3.4.3 PUT

The PUT method is used to update an existing resource on the server. It replaces the existing resource with the new one that is sent in the request.

### 3.4.4 PATCH

The PATCH method is similar to the PUT method, but it only updates the specified fields of an existing resource. It is used when only a portion of a resource needs to be updated, rather than replacing the entire resource.

### 3.4.5 DELETE

The DELETE method is used to delete a resource from the server. It removes the resource from the server and makes it unavailable for further use.

## 3.5 Wireframes

Wireframes help to visualize the app's basic structure and layout, enabling the designer and developer to spot the problems early on in development. It can also be useful for interacting with the user and getting feedback from the user . Besides aiding the design process, wireframes are also useful for user testing to get feedback on the app's usability and identify areas for improvement. Including wireframes in a mobile and web app-based project report provides readers with a tangible example of the app's look and feel, as well as insights into the design process. It allows them to better understand the app's functionality, usability, and design decisions.

Overall, wireframes are a crucial tool for designing successful mobile and web applications. They can help define the app's overall structure, prioritize essential features, and improve communication between designers and developers.

## 3.6 Quantum Computing

A quantum computer is a computer that uses various phenomena of quantum mechanics. At small scales, in the order of nanometers, the phenomena of physical matter exhibiting properties of particles and waves become significant, and quantum computing uses special devices to take advantage of these properties. Due to their quantum nature, they can act exponentially faster than classical computers, as demonstrated by the breaking of symmetric cryptography algorithms in $O(n)$ time complexity. [7]

### 3.6.1 Grover's Algorithm

Grover's Algorithm is a quantum computing algorithm that is used to search data in an unsorted database in $O(\sqrt{n})$ time complexity. Quantum mechanical systems can be in a superposition of states and simultaneously examine multiple names. By properly adjusting the phases of various operations, successful computations reinforce each other while others interfere randomly. [8]

# 4. Methodology

Our project is a full-stack web as well as a mobile app project. The backend part of the project was built using a microservice-based architecture, which is a software development technique that focuses on creating small, modular, and independently deployable services that can work together to form a larger system.

The front for the admin panel was built using a web-based React app which is a popular JavaScript library for building user interfaces and is widely used for developing web applications.

The mobile app which we made for the teachers and students was built using Flutter.

The detailed description of each of these components is explained in the following sections.

## 4.1 Backend

The IOE App is built using a microservices architecture, which allows for the development of a complex application by breaking it down into smaller, independent services that can be developed, tested, and deployed independently. The app is comprised of several microservices, including the campus microservice, which provides information about the campus and its facilities; the teacher microservice, which handles teacher-related functionalities such as managing course content and assessments; the student microservice, which manages student profiles and enrollment information; the curriculum microservice, which manages course and curriculum information; the authentication microservice, which handles user authentication and authorization; the attendance microservice, which tracks attendance for classes; the internal mark submission microservice, which allows teachers to submit internal marks for students; and the result microservice, which calculates and publishes student results. Each microservice is designed to perform a specific set of tasks and can communicate with other microservices through well-defined APIs, allowing for efficient and scalable development of the application.

Below is a brief overview of each microservice and its functionality:

### 4.1.1 Campus Microservice

The campus microservice architecture contains a database that houses data on the campuses provided by IOE, the departments provided by each campus, and the programs offered by each department and campus. A primary key is given to each campus, which is utilized for accessing other database tables. This method enables effective data retrieval and connectivity between the various tables in the campus microservice. For instance, the primary key for that campus is utilized as a foreign key in the linked tables to retrieve the relevant data to retrieve information about the departments offered by that campus. The models are listed below, along with a description of each one:

This microservice is built using Golang, with Gin framework for routing, and Gorm library for ORM.

**Models**

- Campus
- Department
- Programme
- CampusDepartment
- CampusProgramme
- Batch
- Notice

**Campus**

This table is used to store information on campuses of the IOE. The columns of this table are:

Table 4.1: **Columns of Campus table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| name | Name of the campus |
| location | Location of the campus |

**Department**

It includes information on all the generic departments in the IOE.

Table 4.2: **Columns of Department table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| name | Name of the department |

**Programme**

This table contains information about the programs offered by the IOE.

Table 4.3: **Columns of Programme table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| name | Shorthand description, eg BCT |
| full_name | Full name of the programme |
| department_id | Foreign key specifying the offering department |

**CampusDepartment**

This table contains the list of associations specifying the departments present on each campus.

Table 4.4: **Columns of CampusDepartment table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| department_id | Foreign key specifying the department |
| campus_id | Foreign key specifying the campus |

**CampusProgramme**

This table contains the list of associations specifying the list of programmes offered by departments of campuses.

Table 4.5: **Columns of CampusProgramme table and their description**

| Column | Description |
|---|---|
| id | Primary key |
| campusdepartment_id | Foreign key specifying the row in CampusDepartment table |
| programme_id | Foreign key specifying the related programme |

**Batch**

This table contains the list of batches in IOE.

Table 4.6: **Columns of Batch table and their description**

| Column | Description |
|---|---|
| id | Primary key |
| year | Year of that batch |

**Notice**

This is for the publication of any new notices for all the campuses. For example, the notice regarding the entrance examination, exam form, admission procedure, etc.

Table 4.7: **Columns of Notice table and their description**

| Column | Description |
|---|---|
| id | Primary key |
| title | The title of the notice |
| description | Description about the notice |

As an example, let's consider Pulchowk Campus, which is one of the campuses under IOE and is assigned a primary key of 1 in the campus table of the database. Using this primary key, we can retrieve information about the departments offered by Pulchowk by querying the department's table with the primary key for Pulchowk. The departments of the Pulchowk Campus are the Department of Architecture, Department of Civil Engineering, Department of Electronics and Computer Engineering, and so on. Similarly, the campus microservice includes data about departments offered by each campus of IOE, as well as data about the programs offered by each department and campus. This data is stored in separate tables in the database and can be queried and retrieved using the primary keys assigned to each campus, department, and program.

Overall, the use of primary keys to link related data in the campus microservice database ensures that data can be efficiently retrieved and linked across the various tables in the database, making it easier to query and retrieve information about departments, programs, and other data associated with a particular campus or department.

### 4.1.2 Student Microservice

The student microservice is a crucial component of the IOE app architecture as it handles data related to the students enrolled in the institute, including information about their personal details, academic records, attendance, and other related information. It is designed to communicate with other microservices, including the campus microservice, to store and retrieve student data efficiently. The student microservice also includes sections where students belonging to different sections are kept, making it easier for the app to manage and organize student data.

The student microservice is built using the Node.js platform. The microservice uses Node.js to handle requests from other microservices and respond with the appropriate data. The student microservice database includes tables for storing student details, for example, the history of academic records.

The tables of this microservice are:

- Section
- Student

**Section**

It is used to store the sections to which students belong. Additionally, this table references the CampusProgramme and Batch table from Campus Microservice.

Table 4.8: **Columns of Section table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| name | Name of the section |
| batch | batch from the Batch table |
| campusprogramme_id | Id specifying the unique row in CampusProgramme table |

**Student**

It is used to store the detailed information of the students in the IOE.

Table 4.9: **Columns of Student table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| name | Name of the section |
| section_id | Foreign key specifying the section |
| citizenshipNumber | The unique identification of the student |
| dateOfBirth | Date of birth of the student |

As of now, further columns need to be added to the table to store the student, for example, the past academic record, admission type, and entrance examination result.

For example, suppose Smriti is enrolled in IOE in the Electrical Engineering department, Section A. The student microservice would store Smriti's personal details.

### 4.1.3 Teacher Microservice

The teacher microservice in the IOE App is built using the Django web framework, which is a high-level Python web framework that allows developers to build web applications quickly and efficiently. The teacher microservice contains information such as the name, contact details, email address, and other relevant details of the teachers in the institute.

As of now, there is only one table in this microservice, Teacher.

**Teacher**

It is used to store the detailed information of the teachers in the IOE.

Table 4.10: **Columns of teacher table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| name | Name of the teacher |
| post | Post designated to them |
| title | Title designated to them |
| campusDepartmentId | Id referencing the CampusDepartment to which they belong to |

This microservice is utilized in various other fields such as attendance management, internal marks management, and other related services within the IOE App. The teacher microservice communicates with other microservices such as the attendance microservice and internal marks submission microservice to retrieve and store the data relevant to each teacher. By utilizing the Django web framework, we were able to build a robust and scalable microservice that can handle a large amount of data and provide efficient responses to client requests.

We have also implemented a function to search for a teacher's record using Grover's algorithm for fast retrieval. Since there is no currently fully functioning quantum ORM software or library for any databases, we have implemented this by following the steps.

1. Retrieving all the database items into a python list
2. Encoding the id of teacher to be searched in quantum state
3. Using Grover's algorithm to search through the list using quiskit library

### 4.1.4 Authentication Microservice

The authentication microservice plays a critical role in the IOE app architecture as it provides a secure and efficient way to manage user authentication and authorization. This microservice includes features such as login, register, and verification, which allow users to create and manage their accounts within the system. The authentication microservice is designed to provide three different types of accounts, namely admin, student, and teacher, each with their specific roles and permissions.

This microservice contains only the account information about the users of the IOE app.

**User**

It is used to store the information on user accounts of the IOE app.

Table 4.11: **Columns of User table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| username | Unique username of each user |
| passwordHash | Hashed password of the user |
| role | Role can be either TEACHER, STUDENT, ADMIN |
| studentId | Id referencing the CampusDepartment to which they belong to |
| teacherId | Id referencing the CampusDepartment to which they belong to |

To authenticate users and verify their roles, the authentication microservice provides a token that is generated based on the user's credentials. This token is used to identify and authorize users, ensuring that only authorized users can access the protected resources within the system. The authentication microservice is built using the Node.js platform, which is well-suited for creating scalable, high-performance applications. Node.js allows the authentication microservice to handle a large number of requests, ensuring that user authentication is fast and reliable.

The authentication microservice stores a secret key in each microservice, which is used to verify the authenticity of the requests. When a user tries to access a protected resource, the authentication microservice checks the user's credentials and generates a token that contains information about the user's role and permissions. This token is then passed to the other microservices, allowing them to verify the user's identity and ensure that only authorized users can access the resources.

### 4.1.5 Curriculum Microservice

The curriculum microservice is a critical component of the IOE app architecture, as it contains information about the courses offered by the institute. This microservice comprises data related to the courses, such as the course code, name, department offering the course, whether it is practical or theoretical, and other relevant information. By using this microservice, students and teachers can easily access course-related information and plan their academic schedules accordingly.

The curriculum microservice is built using Node.js, which provides several benefits in terms of performance, scalability, and reliability. It uses Node.js to handle requests from other microservices and respond with the appropriate course data. The microservice database includes tables for storing course details, such as the course code, name, department offering the course, and other relevant information.

The tables of this microservice are:

- Subject
- Practical

The detailed information on each table is provided below.

**Subject**

It is used to store the information on the subjects offered by the IOE.

Table 4.12: **Columns of User Subject and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| courseId | Course code in the IOE registry |
| fullName | Full name of the subject |
| marks | Full marks of the subject |
| passPercent | Minimum percentage of marks required for passing |

**Practical**

It is used to store the information the practical subject associated with each subject.

Table 4.13: **Columns of Practical table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| marks | Full marks on the practical |
| passPercent | Minimum percent of marks required for passing |
| subjectId | Foreign key indicating which subject it belongs to |

This microservice can be extended to include the course contents of each subject. Further tables can be created for storing the chapters under each subject and also providing their notes in markdown format which can be hosted in a CDN.

### 4.1.6 Attendance Microservice

The attendance microservice is responsible for managing attendance data for students enrolled in courses at the IOE. This microservice allows teachers to create courses by merging curriculum information with student section information. Once a course has been created, teachers can evaluate the daily attendance of students and record their presence or absence.

The attendance microservice is built using the Node.js platform, which allows for the efficient handling of large amounts of data. The microservice database includes tables for storing course information, student section information, and attendance records. Teachers can create courses by selecting the curriculum for the course and merging it with the section information for the students enrolled in the course.

The tables included in this microservice are as follows

- Course
- Lecture
- AttendanceRecord

The details on these tables are as follows:

**Course**

A course is an educational offering of a particular subject to a particular section. It is used by teacher to create the registry of attendance of students.

Table 4.14: **Columns of Course table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| subjectId | Id specifying which subject is offered in the course |
| sectionId | Id specifying to which section it is offered |
| teacherId | Id specifying by which teacher it is offered |
| name | Name given to the course |

### Lecture

A course usually has multiple lectures, taking place at different dates. This table is used to store the lecture conducted in a certain course.

Table 4.15: **Columns of Lecture table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| courseId | Foreign key specifying to which course the lecture belongs to |

### AttendanceRecord

To store the presence of students in a certain lecture, this table is used.

Table 4.16: **Columns of AttendanceRecord table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| studentId | Id specifying which student is present in the lecture |
| lectureId | Foreign key referencing the belonging lecture |

Once a course has been created, teachers can use the attendance microservice to record the daily attendance of students. The microservice allows teachers to mark students as present or absent for each lecture of the course. Attendance data is stored in the attendance records table.

For example, let's say a teacher named John wants to record the attendance of students in a course called "Introduction to Computer Science." John would use the attendance microservice to create the course by selecting the curriculum information for the course and merging it with the section information for the students enrolled in the course. Once the course has been created, John can use the attendance microservice to mark students as present or absent for each day of the course.

Overall, the attendance microservice is an essential component of the IOE app architecture, allowing teachers to efficiently manage attendance data for students enrolled in courses at the institute. By using Node.js, the microservice is scalable and can handle large amounts of data, ensuring that attendance records are always available when needed.

### 4.1.7 Internal Marks Submission Microservice

The Internal Marks Submission microservice is a critical component of the IOE app architecture as it handles the submission and storage of students' internal marks for each course. The microservice is designed to be used by teachers, who can access the platform to submit the internal marks for each of their students.

The microservice is built using the Node.js platform, which is ideal for building scalable, high-performance applications. The microservice interacts with other microservices, such as the authentication microservice and the teacher microservice, to ensure that only authorized teachers can access the platform and submit internal marks for their students. The tables of this microservice are:

- Course
- InternalMarkRecords

The description of each table is provided below:

**Course**

This table is used to store the course offered. This table is kept separate from the course table of attendance record table because the same teacher may not be responsible for submitting internal marks as the ones who are taking attendance.

Table 4.17: **Columns of Course table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| subjectId | Id specifying which subject is offered in the course |
| sectionId | Id specifying to which section it is offered |
| teacherId | Id specifying by which teacher it is offered |
| name | Name given to the course |

**InternalMarkRecord**

To store the internal mark record of each student in a course, this table is used.

Table 4.18: **Columns of InternalMarkRecord table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| studentId | Id specifying which student is present in the lecture |
| courseId | Foreign key referencing the belonging lecture |
| marksObtained | The marks obtained by the student in the course |
| remarks | Any related remarks about the student |

To use the Internal Marks Submission microservice, a teacher must first create a course by merging the section and curriculum. Once the course has been created, the teacher can access the platform to submit the internal marks for each student. The microservice includes a table for storing the internal marks for each course, along with the student's name, student ID, and other relevant information.

So, the Internal Marks Submission microservice is a crucial component of the IOE app architecture, providing an efficient and secure way for teachers to submit internal marks for their students.

### 4.1.8 Result Publication Microservice

The Result Publication microservice is a critical component of the IOE app architecture as it handles the submission and storage of students' mark sheets. The microservice is used by students to find out the marks and mark sheet on the day the result is published.

The tables used in this microservice are as follows:

- SymbolNumber
- Gradesheet

**Gradesheet**

It is used to store the grade of a symbol number.

Table 4.19: **Columns of Gradesheet table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| symbolnumber | Symbol number of a student |
| subjectcode | Foreign key specifying the id of the subject |
| internal | Marks obtained in internal |
| theory | Marks obtained in theory |
| practical | Marks obtained in practical |

**SymbolNumber**

It is used to relate symbolnumber and student roll number.

Table 4.20: **Columns of SymbolNumber table and their description**

| **Column** | **Description** |
|---|---|
| id | Primary key |
| rollnumber | Roll number of a student |
| symbolnumber | Symbol number of a student |

To use the Result Publication Submission microservice,an administrator will provide the ledger sheet of marks of all student and a record matching symbol number and roll number of students in csv format which would then map the ledger to roll number and provides the result.

Hence, the Result Publication microservice is a crucial component of the IOE app architecture, providing an efficient and secure way for students to get the result on time and at the time of publication. By using Node.js, the microservice is highly scalable and can handle a large number of requests.

## 4.2 Front-End

Our IOE App project provides users with the option of using either a web or mobile app. We've made sure to include several features to make the app easy to use and enjoyable for our users. Whether one is using the web or mobile app, we've designed the IOE App to be user-friendly and interactive.

### 4.2.1 Web App

The web app for IOE App is a browser-based version of our mobile app that runs on desktop and laptop devices. It offers the same features and functionalities as the mobile app, such as attendance tracking, marks management, result viewing, event notifications, and notices. The web app is built using React and CSS, which are powerful web development technologies that allow for fast and seamless user experiences. The user interface is designed to be responsive, intuitive, and optimized for different screen sizes and resolutions, providing a smooth user experience across various web browsers and operating systems.

Our web app comprises of several features including :

**React Framework**

React is a popular JavaScript library for building user interfaces. It allows for the creation of dynamic and responsive user interfaces by breaking down the UI into small, reusable components. React is widely used in web development because of its efficiency, flexibility, and ease of use.

**Reusable components**

React's component-based architecture allows developers to create reusable UI components. This means that developers can create components once and use them multiple times throughout the app. This makes the app more modular and maintainable because changes made to a component are reflected throughout the app.

**Database**

The database is the heart of the web app, where data is stored and organized in a way that is easy to search and retrieve. The database can include information about students, teachers, courses, schedules, events, and other relevant data. This information can be used to generate reports, analyze trends, and make data-driven decisions.

**Table Display**

The database is presented to users in the form of a table, which allows users to easily view and sort the data. The table can also include filtering and search options, which make it easier for users to find the information they need. This can be especially helpful for administrators who need to manage large amounts of data.

### API Calls

The web app uses API calls to communicate with external resources and services. This allows the app to interact with the database, add new data, modify existing entries, and perform other actions as needed. API calls are an important part of the app's functionality because they allow the app to connect with other systems and services.

### Admin Panel

The admin panel provides a secure area where authorized administrators can manage and control various aspects of the app. This can include adding new users, modifying data, managing notifications and other important updates. An admin panel is an important tool for administrators because it allows them to manage the app and its data with ease.

### Dashboard

The dashboard is a central hub where users can view important data and metrics at a glance. The dashboard can include information such as upcoming events, recent notifications, and other important updates. This can help users stay informed and up-to-date with the latest information.

### User Profiles

The web app allows students and teachers to create profiles that include their personal information, course schedules, and other relevant data. User profiles can help users stay organized and keep track of important information such as class schedules, grades, and upcoming assignments.

### Notifications and Announcements

The web app allows users to view notifications, read notices and announcements, and manage their own profile information. Notifications and announcements can be an important way to keep users informed about important updates and events.

### Customization

The web app can be customized to meet the needs of specific schools or universities. This can include customizing the colour scheme, logo, and other visual elements to match the branding of the institution. Customization can help to create a cohesive and personalized experience for users. These features can enhance the user experience, improve data management, and provide valuable insights to help institutions make data-driven decisions.

### 4.2.2 Mobile App

The IOE mobile app is an innovative solution designed to meet the academic needs of students and teachers at the Institution of Engineering (IOE). The app provides a seamless platform for managing academic records, staying up-to-date with events, and receiving important notices. It is built using the Flutter framework, which is a popular cross-platform development tool that enables the efficient creation of high-quality mobile apps for both iOS and Android platforms. The Flutter framework offers numerous benefits, including rapid development, hot reload, and easy customization, making it an ideal choice for mobile app development.

We used the Dart programming language, which is an object-oriented, client-optimized language designed for building user interfaces and applications. Dart offers a range of features, such as static typing, garbage collection, and a just-in-time compiler, that make it ideal for building high-performance mobile apps. Additionally, Dart has a robust community of developers who contribute to the open-source language and its ecosystem, providing a wealth of resources, documentation, and libraries to support developers. By leveraging the power of Flutter and Dart, the IOE mobile app has been developed to provide an intuitive, user-friendly platform that enables students and teachers to manage their academic records and stay informed about important events and notices.

The development of the IOE mobile app using Flutter followed a structured methodology to ensure the app met the needs and requirements of its users. The methodology can be broken down into the following stages:

**Requirements Gathering**

The first stage of the methodology involved gathering requirements from the stakeholders, including students, teachers, and staff members, to understand the features and functionality required in the app. This involved conducting surveys, focus group discussions, and one-on-one interviews to gather feedback and suggestions.

**Design**

Once the requirements were identified, we used a user-centred design approach to create wireframes and prototypes of the app. This involved creating mockups of the app screens and testing them to ensure that the design met our needs and preferences.

**Development**

The development stage involved building the app using Flutter, a powerful framework that allowed for rapid development and testing of the app. We followed agile development practices, which involved building the app in small increments and testing it regularly to identify and fix bugs and issues.

# 5. System design

We first present the ERD diagrams of all the microservices present in our project, and then we present the communication pattern between the microservices.

## 5.1 Campus Microservice ERD

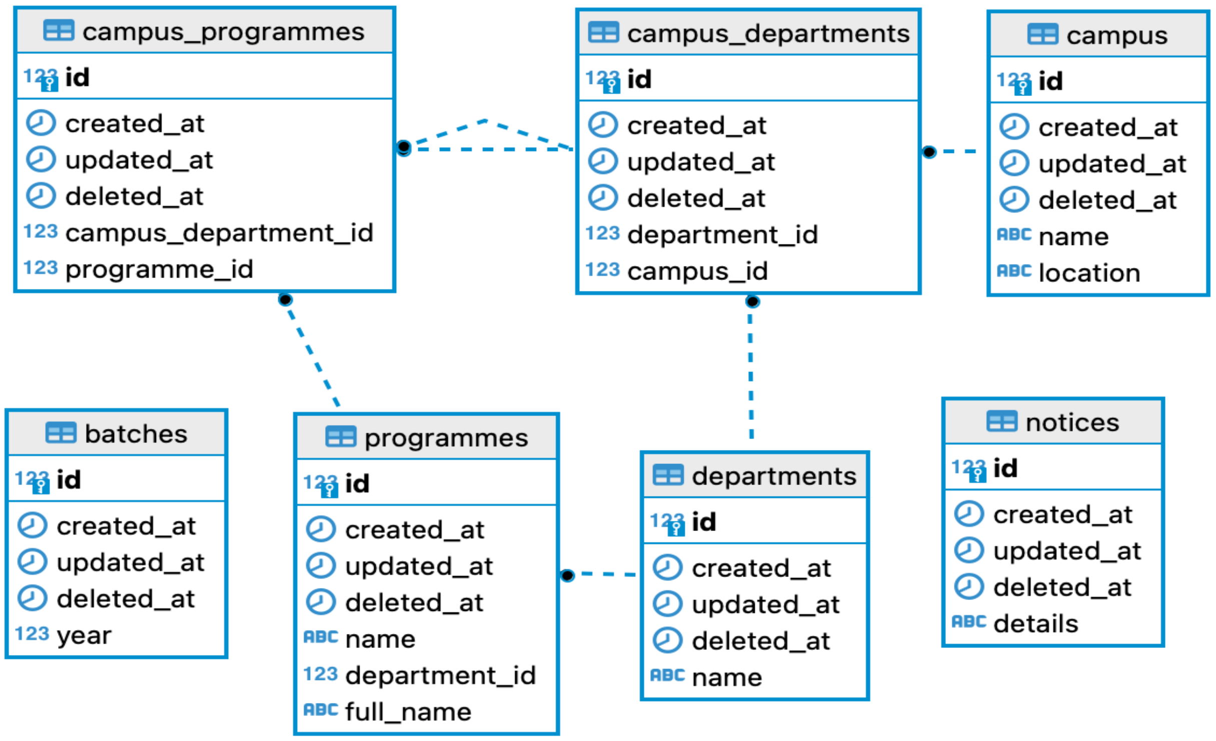


Figure 5.1: ER Diagram of Campus Microservice

## 5.2 Authentication Microservice ERD

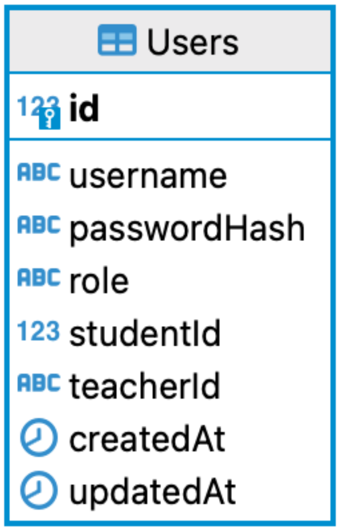


Figure 5.2: ER Diagram of Authentication Microservice

## 5.3 Student Microservice ERD

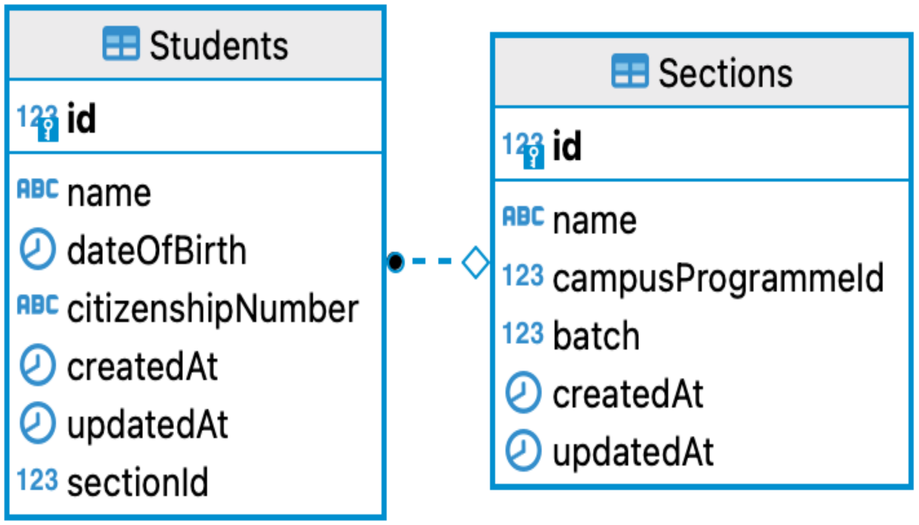


Figure 5.3: ER Diagram of Student Microservice

## 5.4 Curriculum Microservice ERD

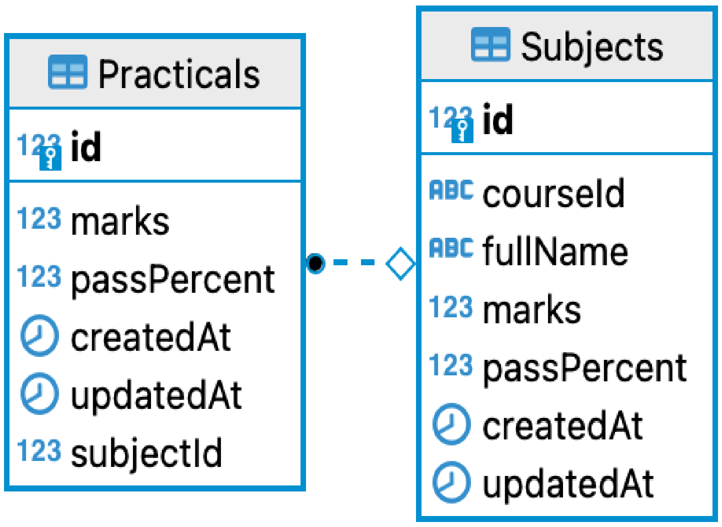


Figure 5.4: ER Diagram of Curriculum Microservice

## 5.5 Teacher Microservice ERD

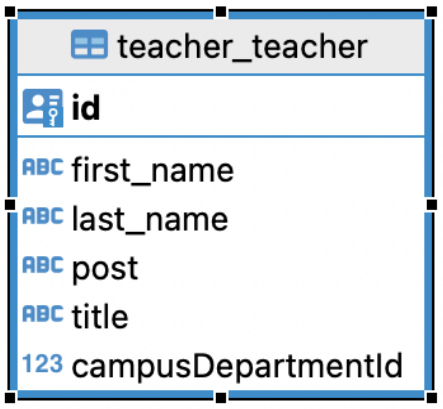


Figure 5.5: ER Diagram of Teacher Microservice

## 5.6 Attendance Microservice ERD

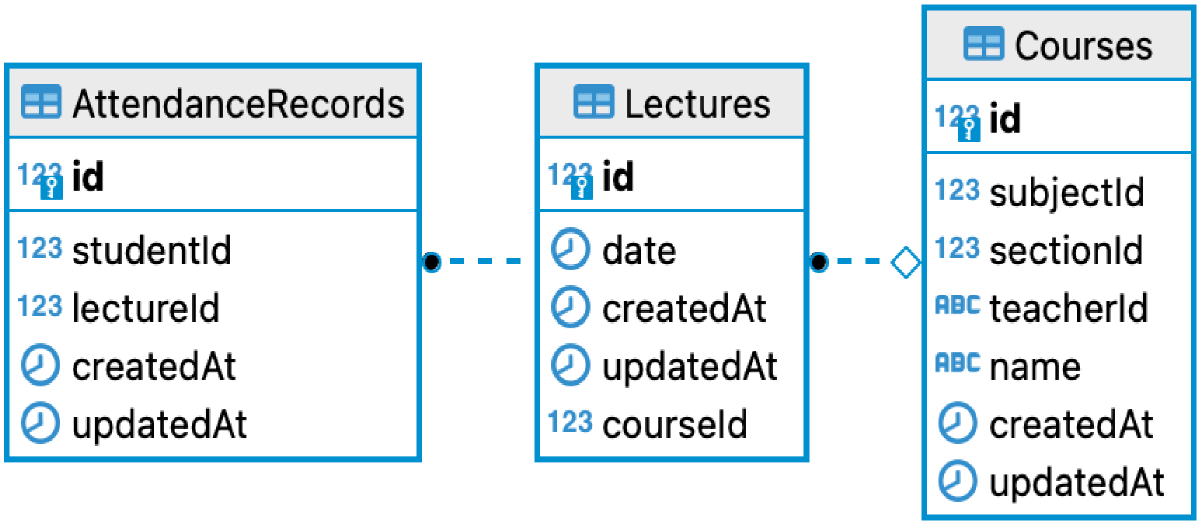


Figure 5.6: ER Diagram of Attendance Microservice

## 5.7 Internal Marks Microservice ERD

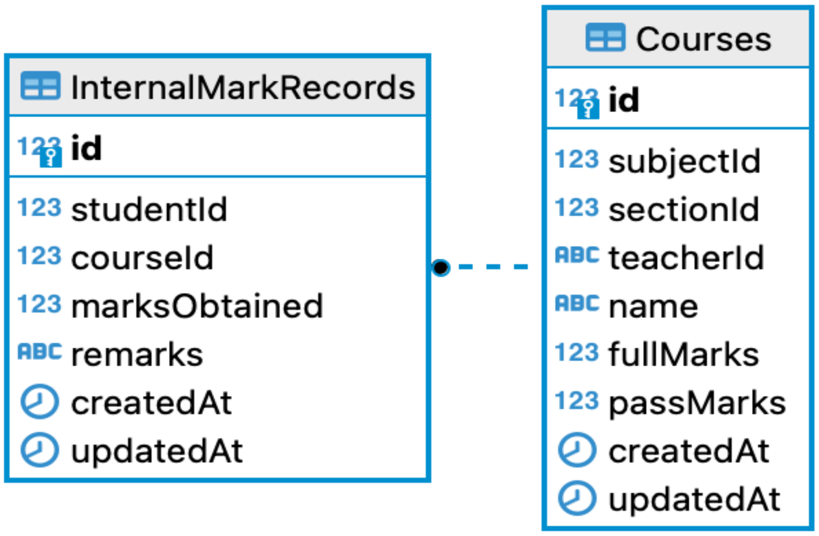


Figure 5.7: ER Diagram of Internal Marks Microservice

## 5.8 Result Publication Microservice ERD

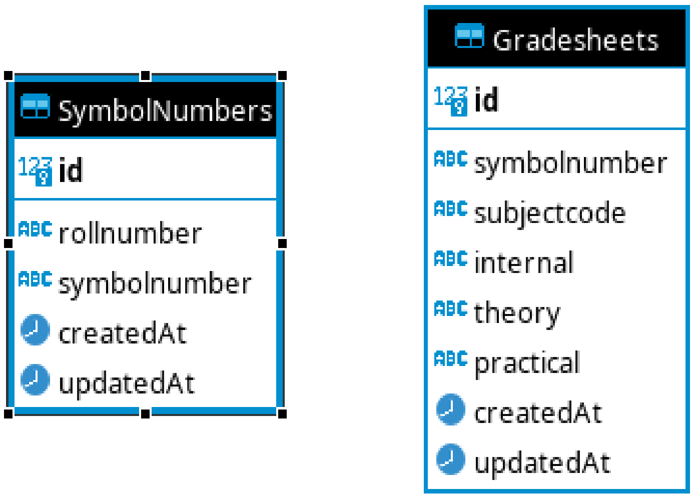


Figure 5.8: ER Diagram of Result Publication Microservice

## 5.9 Communication Between Microservices

The first point of communication is from teachers microservice to the campus microservice. The teacher microservice queries the CampusDepartment table from Campus microservice for the verification of existence of a CampusDepartment instance, since a teacher belongs to a department of a campus.

Another communication point is from student microservice. Student microservice references CampusProgramme instance and Batch instance from Campus microservice, this requires communication with campus microservice for verification of existence of entities with given id.

Similarly, attendance microservice, internal marks submission microservice and result publication microservice needs cross-microservice references to store references to curriculum microservice and student microservice. Attendance microservice references subject from subject microservice, and student as well as section from student microservice. The communication pattern of internal marks microservice and result publication microservice is exactly same as that of attendance microservice.

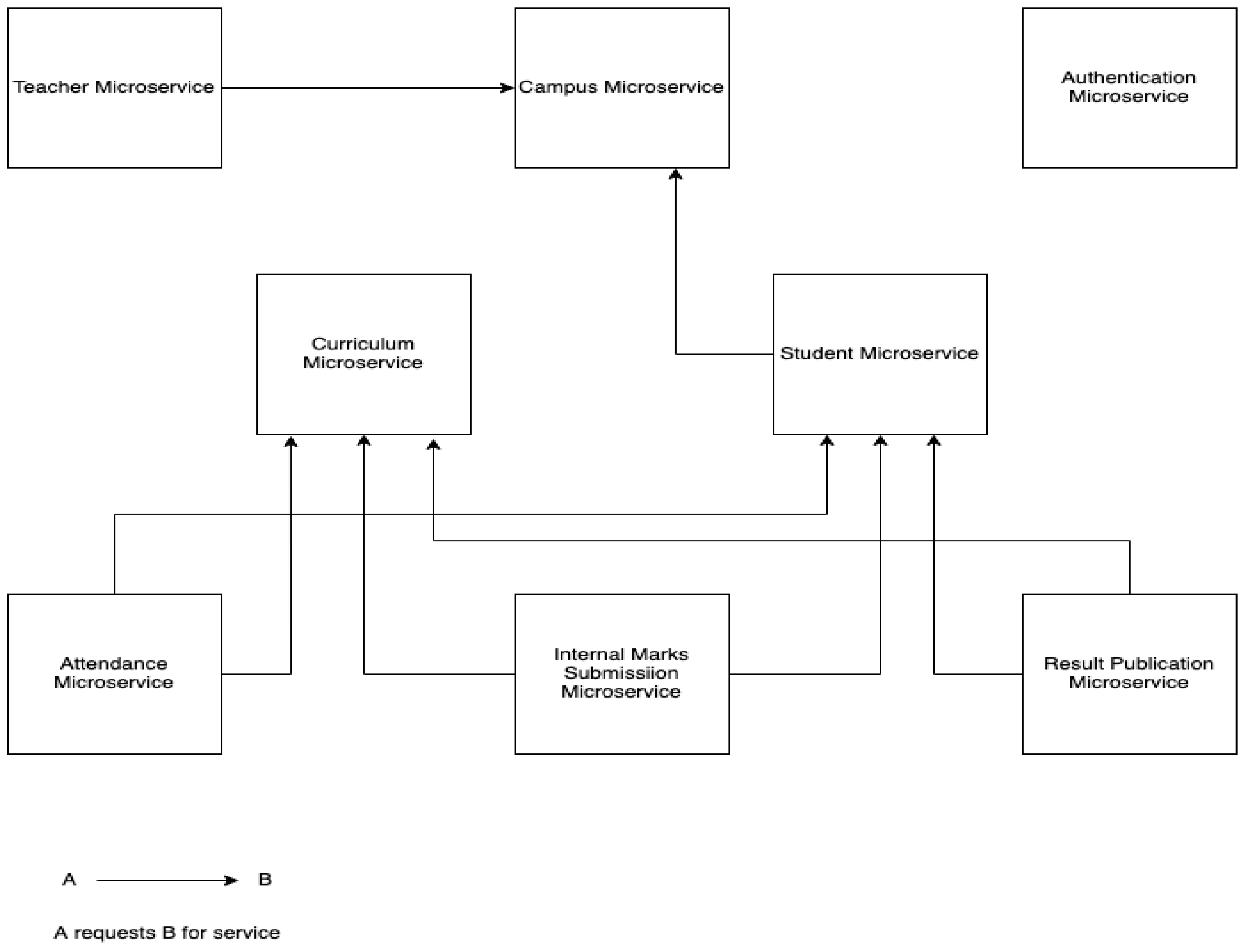


Figure 5.9: Android Dashboard

# 6. Technologies used

## 6.1 Front-End

### 6.1.1 React

React, an open-source project developed by Facebook is a JavaScript library that permits developers to create an interactive user interface for web applications. It helps in simplifying the building of complex user interfaces.

The major work of React involves breaking down the UI into reusable components that are then utilized to construct the more complex UI elements. It supports code reusability and makes the use of state changes easier within the application. In React, there is the virtual DOM whose implementation helps to improve the time and performance for rendering the content in DOM.

### 6.1.2 Typescript

To add the optional static typing to the language, a superset of JavaScript is used which is known as TyeScript. It is better known to improve code quality and maintainability by catching the bugs at the time of compiling. It comprises several features like auto-completion, code navigation, and helping to prevent common errors during programming. With the help of TypeScript, the developers write more maintainable and scalable code creating a better user experience for their applications.

### 6.1.3 Tailwind CSS

Tailwind CSS provides developers with a set of pre-defined CSS classes in order to style their web applications, making it a utility-first CSS framework. It helps to simplify the development process with a predictable as well as consistent set of CSS styles that the user can change as per his/her requirements. Also, the developers don't have to write the custom CSS code because of which they can fully focus on building the functionality for the applications.

Reducing the amount of CSS code to improve the performance of the application is the next big usability of Tailwind CSS.

### 6.1.4 Figma

Figma is a design tool that is used by designers and developers for creating user interfaces which are responsive in nature. It comprises features like vector editing tools, design templates, and interactive prototyping capabilities that enable the users for bringing their ideas in a user-friendly way. It supports real-time collaboration as well as feedback features by which the work in the team becomes more effective. Also, it has an extensive plugin ecosystem that allows it to be integrated with other tools.

### 6.1.5 Flutter

Flutter is an open-source mobile application development framework that is created by Google. Building visually appealing, interactive and high-quality applications for ios, Android as well as the web by the help of a single codebase. There are several distinct features of Flutter as built-in tooling, and architecture based on a widget that allows the developers for creating complex user interfaces and delivering an interactive user experience. It also has hot- a reloading feature to streamline the development process because of which the developers can quickly iterate and make choices for the developers with all the skill levels.

## 6.2 Back-End

### 6.2.1 Gorilla MUX (Golang)

Gorilla MUX is a Golang package providing URL matcher as well as HTTP router. Handling HTTP requests and routing them to the correct handler functions based on the requested URL path and HTTP method. It is better known for its performance, flexibility as well as easy-to-use feature and provides a variety of features like middleware support for authentication and logging, as well as helping to build RESTful APIs by the robust APIs. It is preferred by many developers to use Gorilla Mux for the building of reliable as well as scalable web applications.

### 6.2.2 Node.js

Node.js is a popular and powerful backend framework that enables developers to build fast, scalable, and event-driven applications using JavaScript. It's known for its efficient I/O operations, non-blocking I/O model, and a vast library of modules and packages via its package manager, npm. Node.js is also flexible and easy to learn, thanks to its JavaScript syntax and supportive community. With Node.js, developers can build robust and high-performance backend systems for web and mobile applications alike.

Node.js is a powerful framework for the back end that enables developers for building scalable, faster as well as event-driven applications with the usage of JavaScript. It is better known for its non-blocking I/O model, efficient input-output operations as well as a vast library of modules and packages via npm. It is also quite easier to learn Node.js and build robust as well as high-performance back-end systems for web as well as mobile applications.

### 6.2.3 Django

Django is a well-known Python web framework for simplifying web development. It provides built-in components for the common needs of web development. Following the Model-View-Controller(MVC) pattern and including features such as Object-Relational Mapping makes Django more reactive. The developers can interact with databases and handle their applications in a more efficient manner. It is also modular and scalable and has become the ideal choice for building complex web applications with ease.

### 6.2.4 PostgreSQL

PostgreSQL is widely used by all sized organizations for storing as well as managing their data. This is also a powerful, free and open-source database management system. It has the ability to handle complex transactions and robust features of security. Developers can extend as well as customize PostgreSQL for matching their specific needs. This is a reliable database solution which is a boon for any organization that are intending to store as well as manage their data efficiently.

### 6.2.5 JWT Token

JWT Token stands for JSON Web Token which is an extensively used standard for transmitting information between the parties securely as a JSON object. There are three main parts in JWT Token: a header, a payload, and a signature. The header comprises metadata about the token, the payload consists of the transmitted data like scopes of authorization or the information of the user, and the signature is created using a secret key, allowing the receiver for verifying the token's integrity. These are used generally for authentication as well as authorization in APIs, microservices as well as web applications, since they provide a stateless way ti transmit information between systems.

### 6.2.6 Sequelize

Sequelize is known as an Object-Relational Mapping (ORM) library for NODE.js. It offers an easy-to-use interface for interacting with relational databases like MySQL, PostgreSQL etc. It also allows the developers for writing the database queries using JavaScript Syntax which makes it more readable and manageable. Sequelize provides several powerful features as validation of data, database schema creation as well as modification, and query optimization. It even supports advanced concepts such as transactions and associations, that allows developers to easily model complex relationships between entities. It is widely used in Node.js web applications and consists of a large as well as active circle of users as well as contributors.

# 7. Output

A short description of each of the tabs and pages in the mobile application is provided below.

**Attendance Tab**

The attendance service is a crucial feature of the app that allows students to view their attendance records and teachers to take attendance for their classes. The app should allow teachers to take attendance through their mobile devices, and students to view their attendance records in real-time.

**Internal Marks Submission Tab**

The internal mark submission feature allows students to submit their marks for internal assessment, and teachers to access and manage the marks of their students. The app should have a user-friendly interface that allows students and teachers to upload and view the marks easily.

**Result Section**

The result section of the app allows students to view their examination results. The app should retrieve the result data from the database and display it in an organized way, such as using a table or graph or graph/chart, and provide options for students to filter and search their results based on their courses, semesters, and other relevant criteria.

**Notice Page**

The notice page displays important notices from the institution, such as exam schedules, fee payment deadlines, and other relevant information. The app should allow users to view the notices in a list or grid format, and provide options to filter and search the notices based on their categories and tags.

**Navigation**

The app should have a smooth navigation system that allows users to move from one page to another seamlessly. This can include a navigation drawer, a bottom navigation bar, or any other intuitive navigation system. The app should also have a consistent visual design and layout across all the screens, using appropriate typography, colours, and icons.

Overall, the IOE mobile app using Flutter can offer several features that can help students and teachers stay up-to-date with their attendance, marks, results, events, and notices. The app should have a simple and intuitive interface that allows users to navigate through the app easily, and provide options to customize the app settings and preferences based on their needs and preferences. By leveraging the power of Flutter, the IOE mobile app can provide a seamless and engaging user experience to its users.

The output of our project can be visualized as below:

## 7.1 Teacher App

### 7.1.1 Home

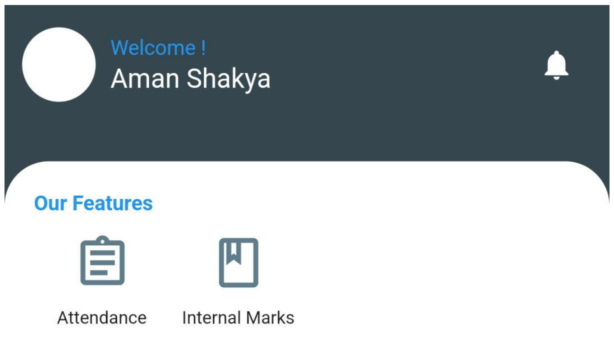
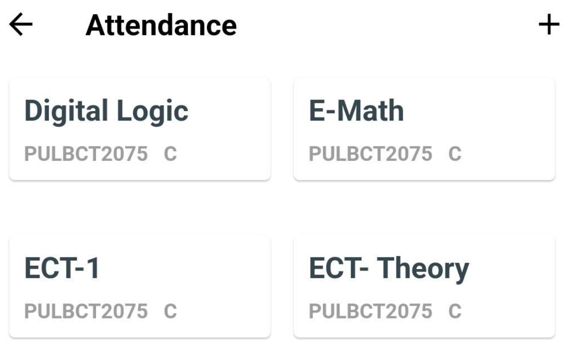


Figure 7.1: Android Homepage

### 7.1.2 Attendance module



Digital Logic
PULBCT2075 C

E-Math
PULBCT2075 C

ECT-1
PULBCT2075 C

ECT- Theory
PULBCT2075 C

Figure 7.2: Attendance page

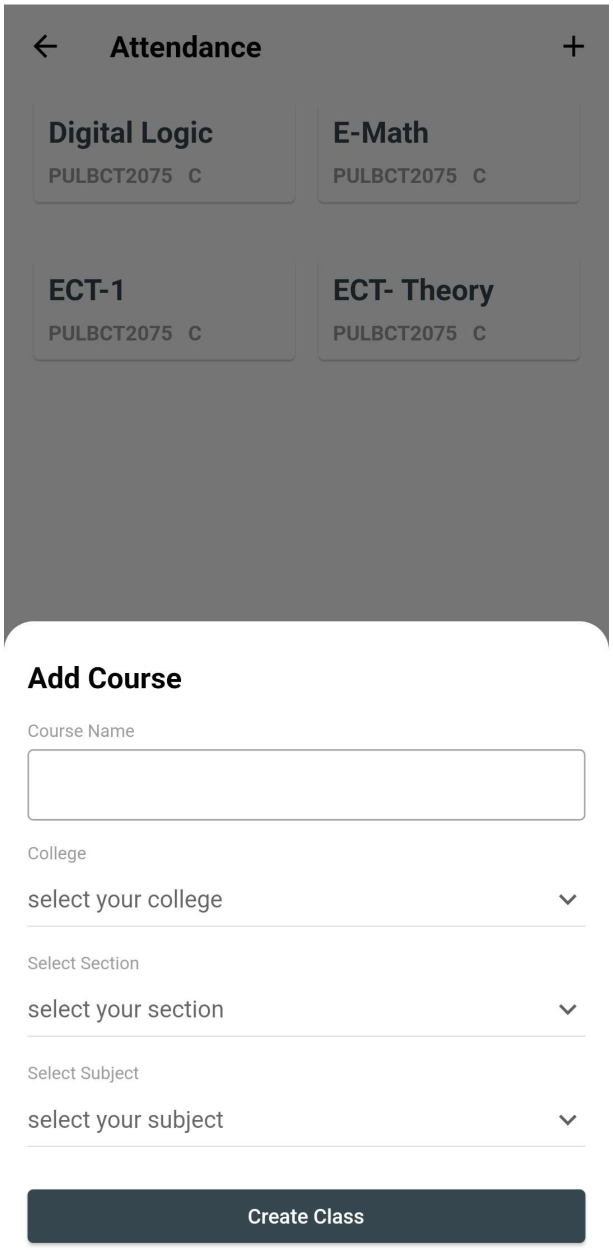
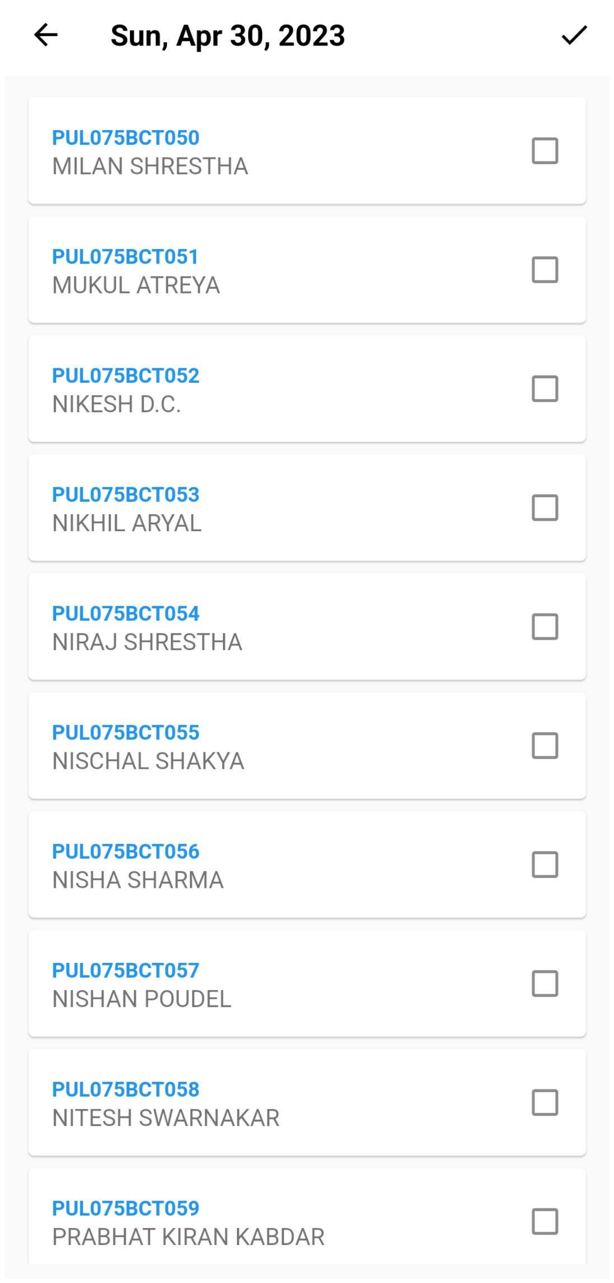


Figure 7.3: Adding a new course


Sun, Apr 30, 2023

PUL075BCT050
MILAN SHRESTHA

PUL075BCT051
MUKUL ATREYA

PUL075BCT052
NIKESH D.C.

PUL075BCT053
NIKHIL ARYAL

PUL075BCT054
NIRAJ SHRESTHA

PUL075BCT055
NISCHAL SHAKYA

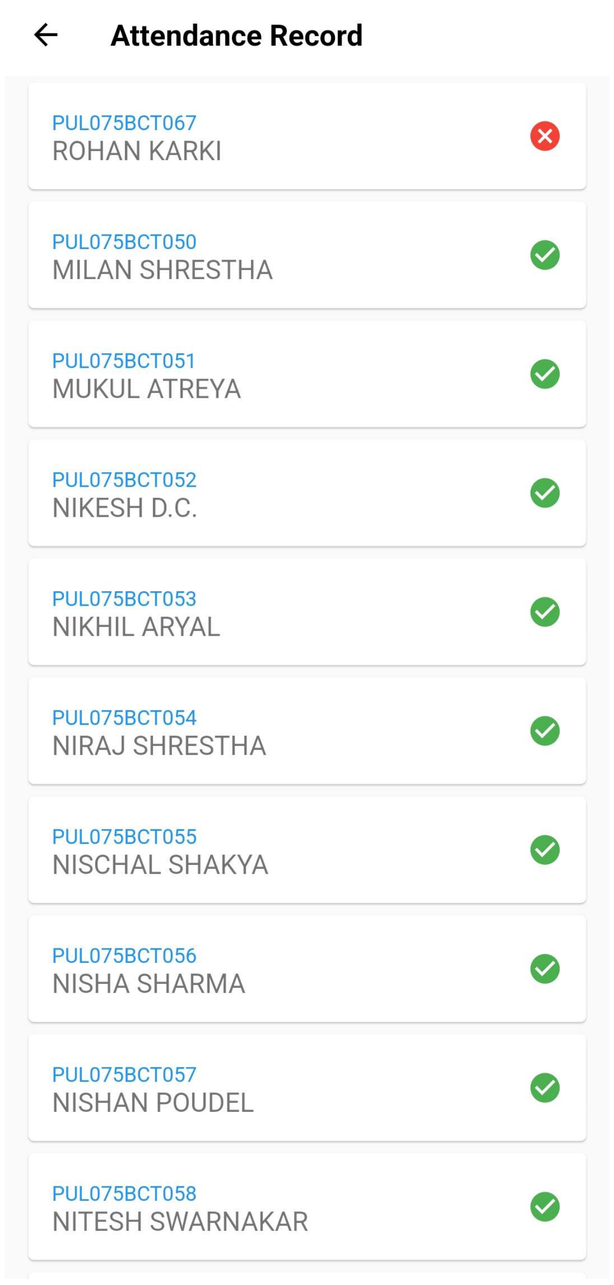
PUL075BCT056
NISHA SHARMA

PUL075BCT057
NISHAN POUDEL

PUL075BCT058
NITESH SWARNAKAR

PUL075BCT059
PRABHAT KIRAN KABDAR

Figure 7.4: Taking attendance



Attendance Record

| | |
|---|---|
| PUL075BCT067 ROHAN KARKI | ✗ |
| PUL075BCT050 MILAN SHRESTHA | ✓ |
| PUL075BCT051 MUKUL ATREYA | ✓ |
| PUL075BCT052 NIKESH D.C. | ✓ |
| PUL075BCT053 NIKHIL ARYAL | ✓ |
| PUL075BCT054 NIRAJ SHRESTHA | ✓ |
| PUL075BCT055 NISCHAL SHAKYA | ✓ |
| PUL075BCT056 NISHA SHARMA | ✓ |
| PUL075BCT057 NISHAN POUDEL | ✓ |
| PUL075BCT058 NITESH SWARNAKAR | ✓ |

Figure 7.5: Viewing Attendance

### 7.1.3 Internal Marks Submission module

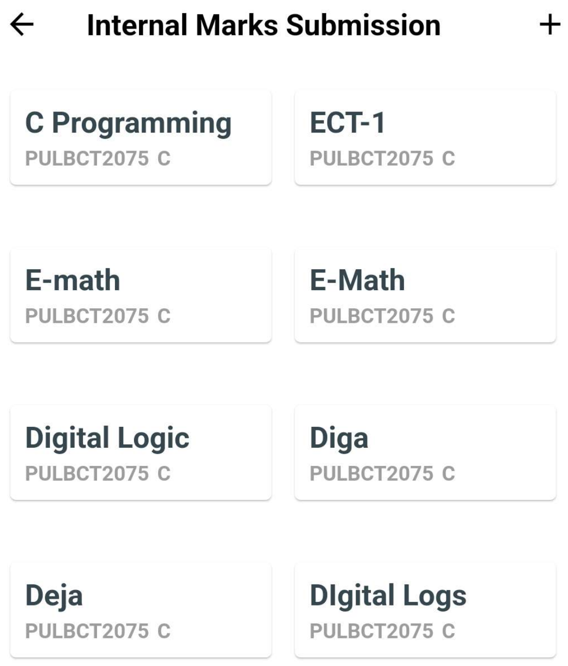


Figure 7.6: Internal Marks Submission Page

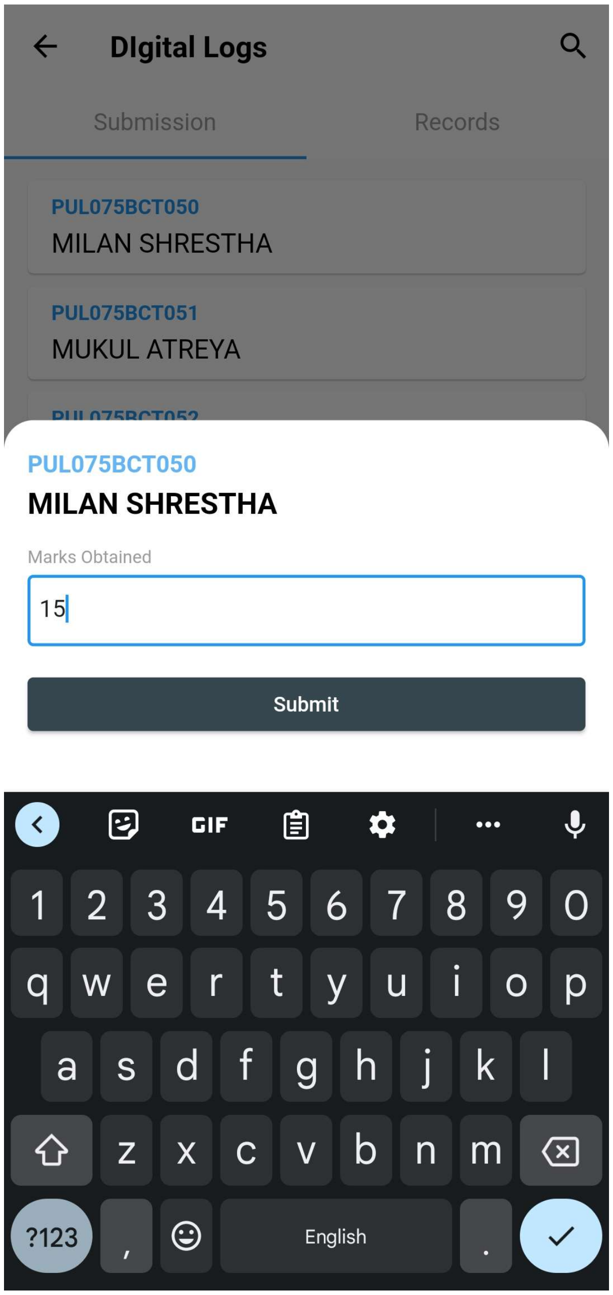


Figure 7.7: Assign Marks

## 7.2 Student App

### 7.2.1 Attendance Page

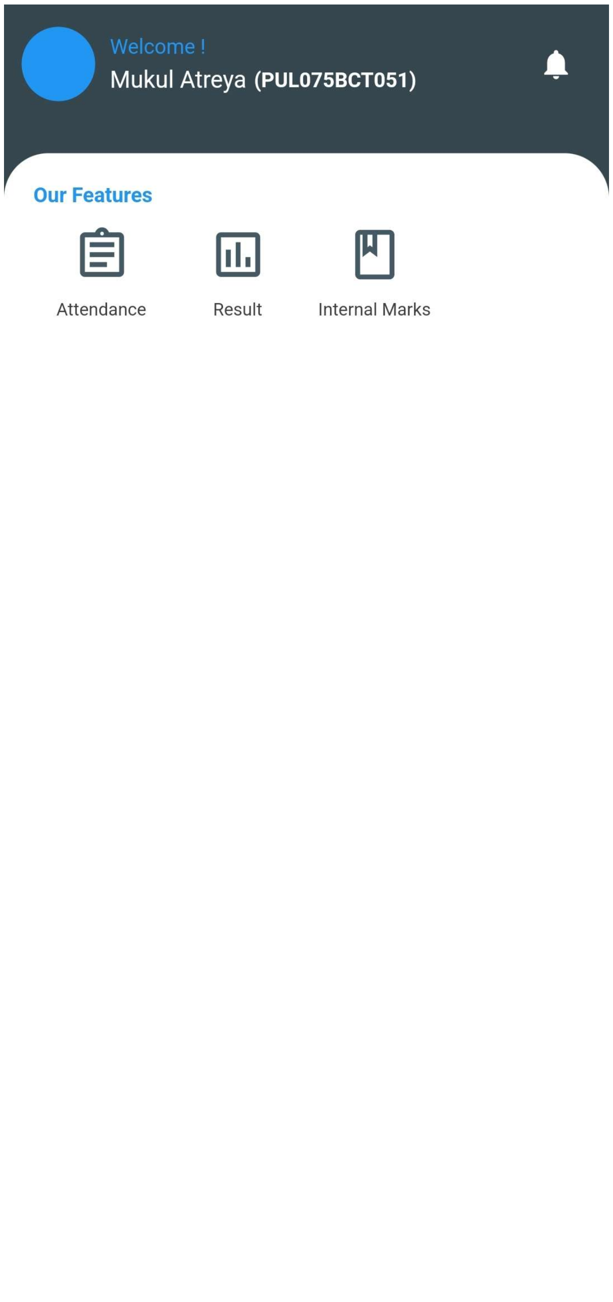


Figure 7.8: Student Home

### 7.2.2 Attendance Page

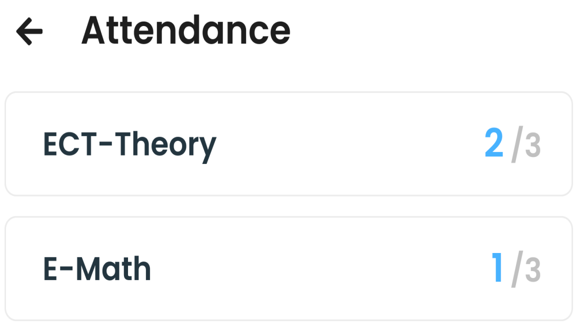
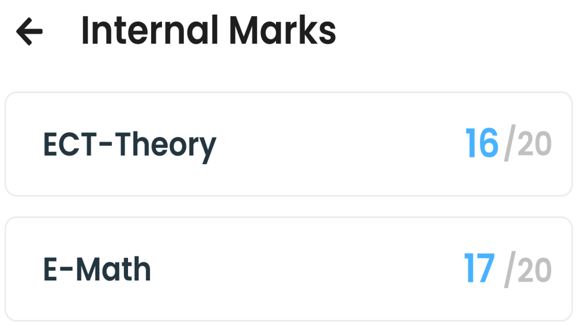


Figure 7.9: Attendance Page

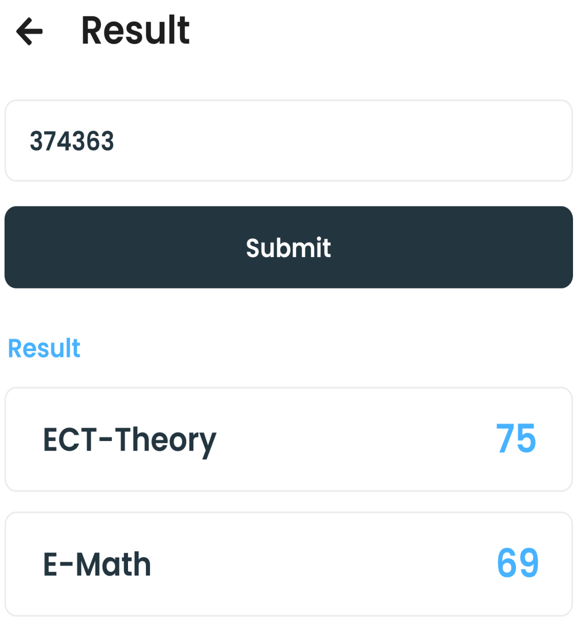
### 7.2.3 Internal Marks Page


Internal Marks

| | |
|---|---|
| ECT-Theory | 16/20 |
| E-Math | 17/20 |

Figure 7.10: Internal Marks Page

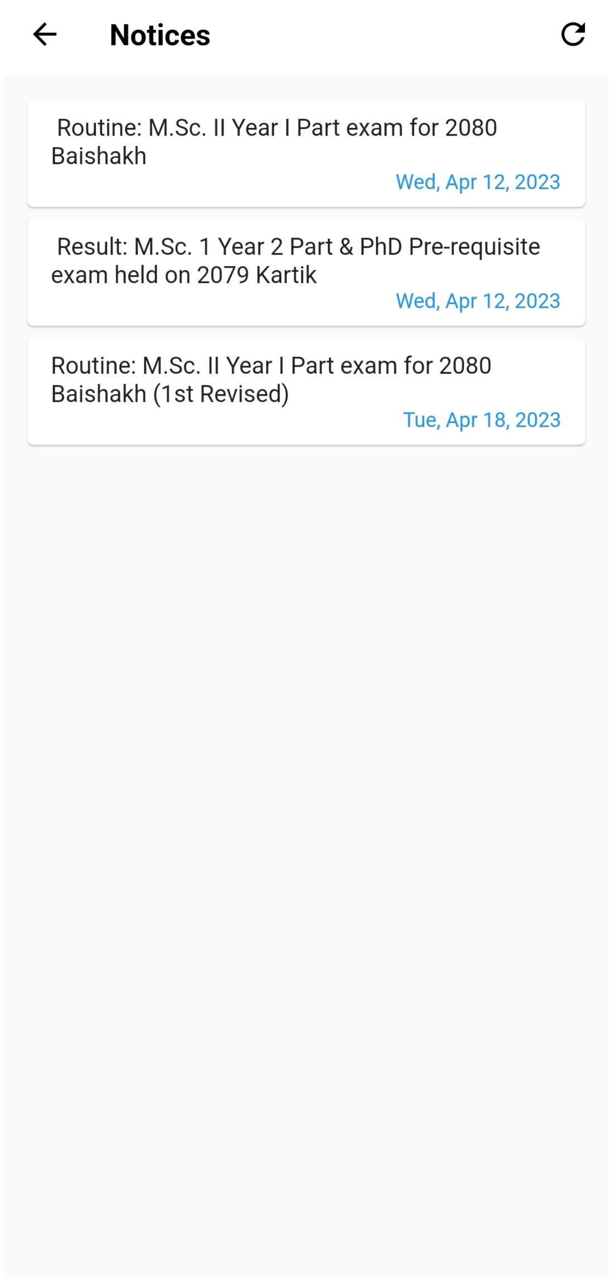
### 7.2.4 Results Page



| | |
|---|---|
| ECT-Theory | 75 |
| E-Math | 69 |

Figure 7.11: Results Page

### 7.2.5 Notice



Routine: M.Sc. II Year I Part exam for 2080 Baishakh
Wed, Apr 12, 2023

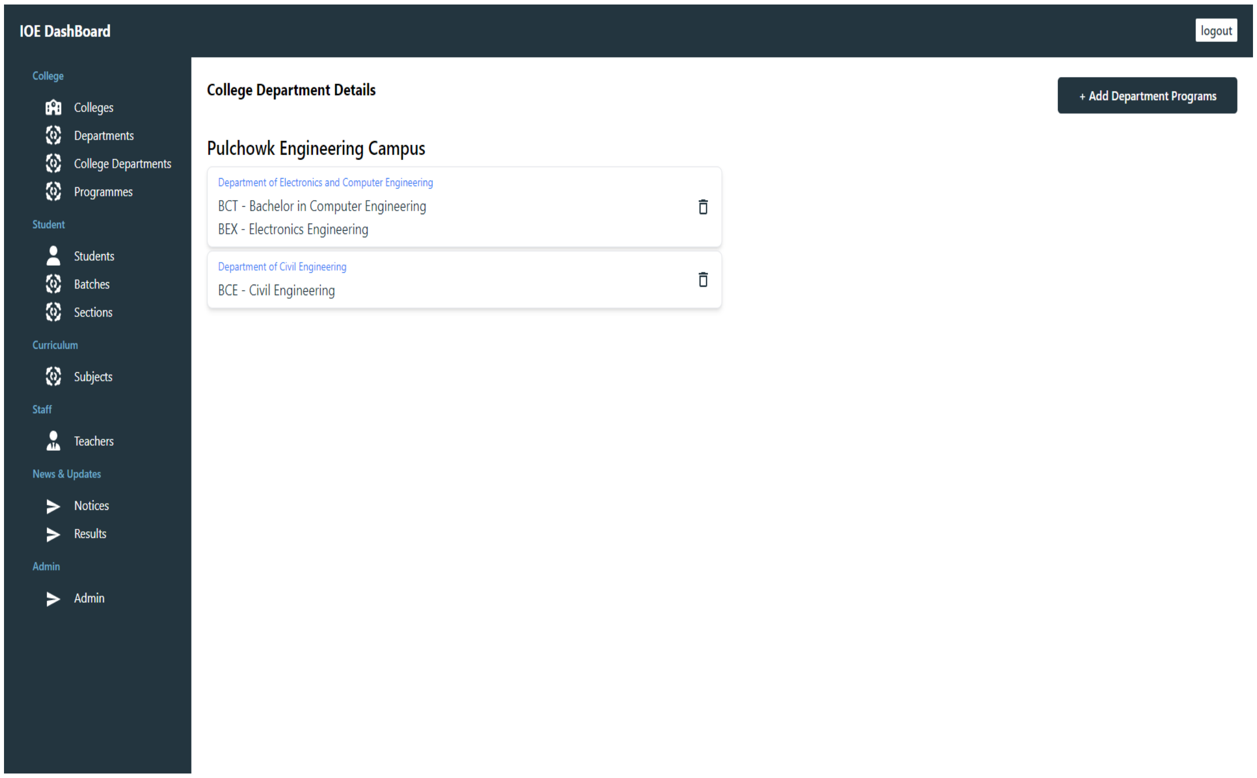
Result: M.Sc. 1 Year 2 Part & PhD Pre-requisite exam held on 2079 Kartik
Wed, Apr 12, 2023

Routine: M.Sc. II Year I Part exam for 2080 Baishakh (1st Revised)
Tue, Apr 18, 2023

Figure 7.12: Notice Page

### 7.2.6 Result Page

## 7.3 Admin Dashboard



College Department Details

Pulchowk Engineering Campus

Department of Electronics and Computer Engineering
BCT - Bachelor in Computer Engineering
BEX - Electronics Engineering

Department of Civil Engineering
BCE - Civil Engineering

Figure 7.13: Android Dashboard

# 8. Conclusion

In conclusion, the IOE App project has been developed and implemented to provide the Institute of Engineering with an integrated application through which all the activities of the IOE can be tracked. It helps students to get easy access to academic information and resources. The present scenario of IOE using various outsourcing applications with separate databases can be easily tackled with this application. The app has a user-friendly interface, which allows students to quickly find important information, such as schedules, exam results, course materials, and announcements. Moreover, the app can reduce the workload of the administrative staff by automating routine tasks. Moving forward, the project team can enhance the app by incorporating additional features based on user feedback and emerging technology trends. The IOE App project is a valuable contribution to the IOE community, and it has the potential to significantly improve the experience of stakeholders of IOE by providing them with easy access to essential information.

# 9. Limitations and Future enhancement

In the course of implementing our project, several limitations have been observed, which, if addressed, could further improve the system's performance and functionality. The limitations include:

Firstly, the communication between microservices is currently in synchronous mode. The introduction of event-based communication could enhance the system's efficiency and responsiveness.

Secondly, a pipeline from attendance to internal marks and result publication could be built to automate the process of result generation and make it more streamlined.

Thirdly, the implementation of identity and access management could significantly improve the system's authorization and flexibility. Additionally, introducing campus-level admin access for admin accounts could provide better control over the system.

Finally, the implementation of artificial intelligence for tasks such as login using an ID card for students could improve the system's usability and security.

It is important to note that these limitations and future enhancements are not exhaustive and that there may be other potential areas of improvement that could be explored. However, addressing the above-mentioned limitations would enhance the functionality and overall performance of the system. Therefore, in future works, we recommend further exploring these areas to improve the system's overall quality and user experience.

# References

[1] Kewal Shah, Harsh Sinha, and Payal Mishra. Analysis of cross-platform mobile app development tools. In *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*, pages 1–7. IEEE, 2019.

[2] Sam Newman. *Building microservices.* " O'Reilly Media, Inc.", 2021.

[3] Wilhelm Hasselbring and Guido Steinacker. Microservice architectures for scalability, agility and reliability in e-commerce. In *2017 IEEE International Conference on Software Architecture Workshops (ICSAW)*, pages 243–246. IEEE, 2017.

[4] Kathmandu university - apps on google play.

[5] Samy S Abu-Naser, Mahmoud Abu Ghosh, and Rasha R Atallah. Mobile cloud computing: Academic services for palestinian higher education institutions (mccas. 2015.

[6] Jose George Dias de Souza and Daniel Scherer. Gestation: A microservice architecture for a prenatal care application. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 683–687. IEEE, 2021.

[7] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Breaking symmetric cryptosystems using quantum period finding. 2016.

[8] Lov K. Grover. A fast quantum mechanical algorithm for database search, 1996.