Classkepper:- The Ultimate Time Saving Solution

A Major Project Report

submitted in partial fulfillment of requirements of the degree

of

Bachelor of Computer Applications

by

Taha

EN21CA301055

under the guidance of

Prof Anil Patidar

Department of Computer Applications

Faculty of Engineering

MEDI-CAPS UNIVERSITY, INDORE- 45333

May / 2024

Classkepper:- The Ultimate Time Saving Solution

A Major Project Report

submitted in partial fulfillment of requirements of the degree

of

Bachelor of Computer Applications

by

Taha
EN21CA301055

under the guidance of

Prof Anil Patidar

Department of Computer Applications
Faculty of Engineering

MEDI-CAPS UNIVERSITY, INDORE- 453331

May / 2024

Report Approval

The project work **“ClassKepper”** is hereby approved as a creditable study of an engineering/computer application subject carried out and presented in a manner satisfactory to warrant its acceptance as prerequisite for the Degree for which it has been submitted.

It is to be understood that by this approval the undersigned do not endorse or approval any statement made, opinion expressed or conclusion drawn there in; but approve the “Project Report” only for the purpose for which it has been submitted.

Examiner(s)

Name(s)	Signature(s)
1. …………………………	…………………………………….
2. …………………………	…………………………………….
3. …………………………	…………………………………….
4. …………………………	…………………………………….

Date: 24/05/2024

<u>Declaration</u>

I hereby declare that the project entitled **“Classkepper”** submitted in partial fulfillment for the award of the degree of Master of Computer Applications in “BCA+MCA INTEGRATED” completed under the supervision of **Prof Anil Patidar HOD Of Master of Computer Application ,** Faculty of Engineering, Medi-Caps University Indore is an authentic work.

Further, I/we declare that the content of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for the award of any degree or diploma.

Signature of the Candidate: ____________________

Name of the Candidate: Taha

CERTIFICATE OF THE SUPERVISOR

This is to certify that the work entitled "Classkepper" is a piece of Project work done by Taha Under my Guidance and Supervision for the degree of BCA from Medi-Caps University Indore (M.P.)

I certify that the candidate has put in an attendance of more than 75% with me.

________________________	________________________
Prof Anil Patidar	**Prof Anil Patidar**
Head of Department	Head of Department
Computer Application	Computer Application
Medi-Caps University	Medi-Caps University,

Acknowledgement

I would like to express my deepest gratitude to our Honorable Chancellor, **Shri R. C. Mittal,** who has provided me with every facility to successfully carry out this project, and my profound indebtedness to **Prof. (Dr.) D. K. Patnaik,** Vice-Chancellor, Medi-Caps University, whose unfailing support and enthusiasm has always boosted up my morale. I also thank **Prof. (Dr.) Pramod S. Nair,** Dean, Faculty of Engineering, Medi-Caps University, for giving me a chance to work on this project. I would also like to thank my Head of the Department **Prof. Anil Patidar** for his continuous encouragement for betterment of the project.

I express my heartfelt gratitude to my **Major Project Guide, Prof Anil Patidar** ,Department of **Master of Computer Application**, for continuous help and support which helped me to complete this project.

Taha EN21CA301055
BCA - MCA (Integrated) VI Semester
Department of Computer Applications
Faculty of Engineering
Medi-Caps University, Indore

Abstract

In the dynamic landscape of higher education, efficient attendance tracking and seamless communication between teachers and students are critical components for academic success. ClassKepper is a revolutionary mobile application designed to streamline the attendance management process and facilitate real-time communication of important notes and updates between teachers and college students. This innovative app serves as a comprehensive solution to bridge the gap between traditional attendance tracking methods and the evolving needs of modern educational environments.

ClassKepper provides an intuitive interface for teachers to take attendance directly from their mobile devices, eliminating the need for manual record-keeping. Through the app, instructors can quickly mark the attendance status of each student with just a few taps, saving valuable class time and reducing the likelihood of errors associated with traditional methods. The real-time synchronization feature ensures that attendance data is instantly reflected in the respective students' portals, providing both teachers and students with up-to-date and accurate records.

For students, ClassKepper offers a dedicated portal where they can access their attendance records in real-time. This transparency promotes accountability and allows students to stay informed about their attendance status throughout the semester. The application also serves as a centralized hub for sharing important course-related information, such as lecture notes, announcements, and supplementary materials. This feature enhances Format for Preparation of Minor Project communication between teachers and students, fostering a collaborative and engaged learning environment

Table of Contents

INTRODUCTION

Description of project

Welcome to ClassKepper, the ultimate solution for seamless attendance management and student note organization. In the dynamic world of education, staying on top of student engagement and progress is crucial, and ClassKepper is designed to simplify these tasks with efficiency and precision.

Gone are the days of manual attendance tracking and scattered student notes. ClassKepper harnesses the power of technology to streamline these processes, providing educators with a user-friendly and intuitive platform. With our app, you can effortlessly take attendance, with just few taps and store the data forever in the database and edit the data according to your needs

And the Student do not need to worry about notes anymore as in this app all the notes are provided in one place arranged in semester wise and subject wise categories

ClassKepper will save a lot of time of students and teacher's by removing the need of physical registers for taking attendance and using third party applications for Notes keeping and student will get all the notes of subjects, unit wise in the same platform

<u>BACKGROUND</u>

Description of the existing system

1. Manual Attendance Tracking: Use physical attendance sheets or registers where individuals can sign in or mark their attendance manually when they arrive at a class, meeting, or event.

2. Paper-Based Note Taking: Provide notebooks or loose sheets of paper for individuals to jot down their notes during lectures, meetings, or other learning sessions.

3. Organizational Structure: Establish a hierarchical system for organizing attendance records and notes. For example, attendance sheets could be sorted by date or event, while notes could be categorized by subject or topic.

4. Storage Solutions: Utilize physical storage solutions such as filing cabinets or folders to store attendance records and notes securely.

5. Manual Data Entry: Assign personnel to manually enter attendance data and notes into centralized records or ledgers. This could involve transcribing information from paper-based documents into handwritten ledgers or typed documents.

6. Verification Process: Implement a verification process where attendance records and notes are reviewed and verified for accuracy by designated individuals, such as supervisors or administrators.

7. Backup Procedures: Establish backup procedures to safeguard attendance records and notes against loss or damage. This could involve creating duplicate copies or storing records in multiple physical locations.

8. Communication Channels: Use traditional communication channels such as verbal announcements, notice boards, or written memos to disseminate information related to attendance requirements and note-taking guidelines.

Circumstances leading to the current new system

- Now as we all know, almost everything can be done online.
- Like Money transfer, Shopping, Booking, Teaching, Data sharing, Admissions, Job search, etc. And so many other activities are done with the help of internet.
- So with the easy access and use of internet, we are going to take this existing Attendance and Notes Managing System Down
- We are going to develop an mobile application so that the same process could be done easily without the waste of time, afford, and energy.
- So firstly, faculty and students are required to login to our app using the email id and password provided to them
- Once registration is done, faculty can easily take attendance on the app for their respective subjects
- And similarly, Students will get real time updates for the attendance and students can also access notes for current as well as previous semester subjects
- So this system will save a lot of time, energy, and afford for both Students and Faculty

Work already carried out in the project domain

1. **Project Scope Definition:**

- Defined the scope of the project as developing a mobile application using React Native for both Android and iOS platforms.
- Identified the primary functionalities of the application as attendance management and storing notes in PDF format.
- Established the need for a backend database to store attendance records leading to the decision to use JSON Server.

2. **Technology Stack Selection:**

- Chose React Native as the primary framework for mobile app development due to its cross-platform compatibility and extensive community support.
- Selected JSON Server as the backend database solution for its simplicity and compatibility with React Native applications.
- Researched and evaluated alternative technologies for each component of the project, considering factors such as ease of implementation, scalability, and cost-effectiveness.

3. **Project Setup and Environment Configuration:**

- Set up development environments for both mobile app development using React Native and backend server development using JSON Server.
- Configured necessary dependencies and libraries for React Native development, including navigation, state management, and PDF handling.

- Initialized JSON Server and defined the data model for storing attendance records and notes in JSON format.

4. **User Interface Design:**
- Designed intuitive user interfaces for features such as attendance tracking, note-taking, and PDF storage, following best practices for mobile app design and usability.

5. **Implementation of Core Features:**
- Developed the core functionalities of the application, including:
- User authentication and authorization for secure access to attendance and notes.
- Attendance management system allowing users to mark attendance for classes.
- Implemented front-end components using React Native components and libraries, ensuring responsiveness and compatibility with various device screen sizes.

6. **Integration with JSON Server:**
- Established communication between the React Native application and JSON Server using RESTful API endpoints.
- Implemented CRUD operations for managing attendance records, ensuring data consistency and integrity.
- Handled authentication and authorization mechanisms to secure access to backend resources and user data.

Objective of the project

1. **Efficient Attendance Management:**

- Develop a feature-rich attendance management system that allows teachers or administrators to easily track and record student attendance for classes and events.
- Enable seamless marking of attendance using the mobile application, reducing the time and effort required for manual attendance tracking.

2. **Enhanced Accessibility and Mobility:**

- Design the application to be cross-platform compatible, ensuring accessibility for both Android and iOS users.
- Optimize the user interface for mobile devices, prioritizing responsiveness and usability on various screen sizes and resolutions.

3. **Secure User Authentication and Authorization:**

- Implement robust authentication mechanisms to ensure secure access to the application and its features.
- Incorporate role-based access control (RBAC) to define user permissions and restrict unauthorized access to sensitive data and functionalities.

4. **Real-time Data Synchronization:**

- Enable real-time synchronization of attendance records and notes between the mobile application and the backend server, ensuring data consistency across device

What is to be achieved and method of measuring the extent of that achievement

1. **Increased Efficiency in Attendance Tracking:**

- Objective: Reduce the time spent on manual attendance tracking by at least 50%.
- Measurement Method: Compare the time taken to mark attendance manually before using ClassKepper with the time taken to mark attendance using the application. Conduct surveys or interviews to gather feedback from teachers/administrators on time savings.

2. **Improved Attendance Accuracy:**

- Objective: Achieve an accuracy rate of 95% or higher in recording student attendance.
- Measurement Method: Compare attendance records generated by ClassKepper with manual records or existing attendance tracking systems. Calculate the percentage of discrepancies and identify areas for improvement.

3. **Enhanced Note-taking Experience:**

- Objective: Increase user satisfaction with note-accessing features by at least 30%.
- Measurement Method: Conduct user satisfaction surveys or interviews to gather feedback on the usability, functionality, and effectiveness of the note-accessing features in ClassKepper. Analyze feedback to identify areas for improvement.

ANALYSIS

System Requirement Analysis

1. **Functional Requirements:**

- **User Authentication:** Users should be able to create accounts, log in securely, and access the application's features.
- **Attendance Management:** Users should be able to mark attendance for classes, view attendance records, and generate attendance reports.
- **Note-Accessing:** Users should be able to Access, and organize notes in PDF format,.
- **Data Synchronization:** The application should synchronize attendance records and notes between the mobile app and the backend server in real-time.
- **User Management:** Administrators should have the ability to manage user accounts, roles, and permissions.
- **Notification System:** Users should receive notifications for important events such as upcoming classes, pending tasks, or system updates.
- **Offline Capability:** The application should support offline access to previously accessed data, with the ability to synchronize changes once an internet connection is available.

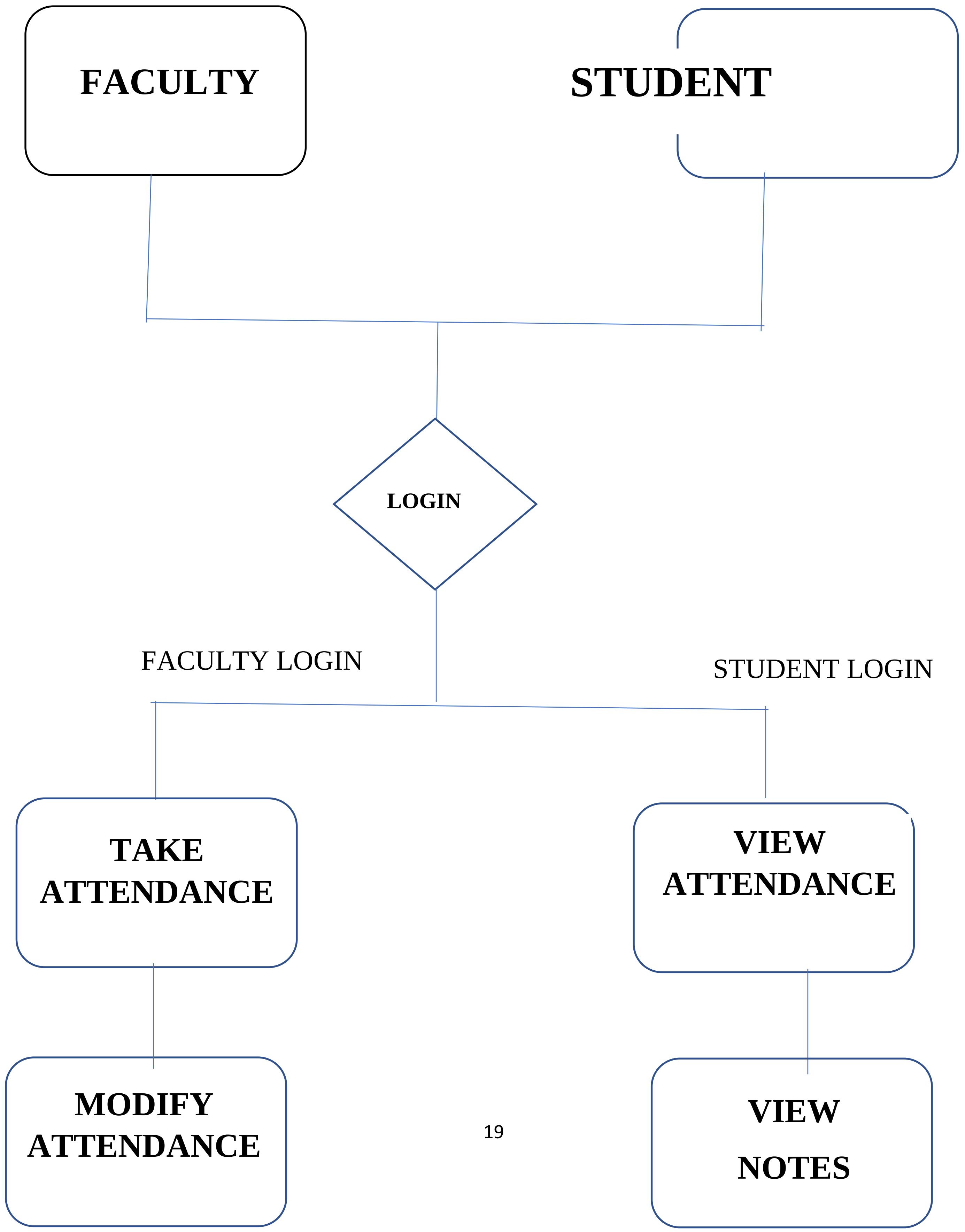
2. **Non-Functional Requirements:**

- **Performance:** The application should respond quickly to user interactions, with minimal loading times and efficient data retrieval and processing.

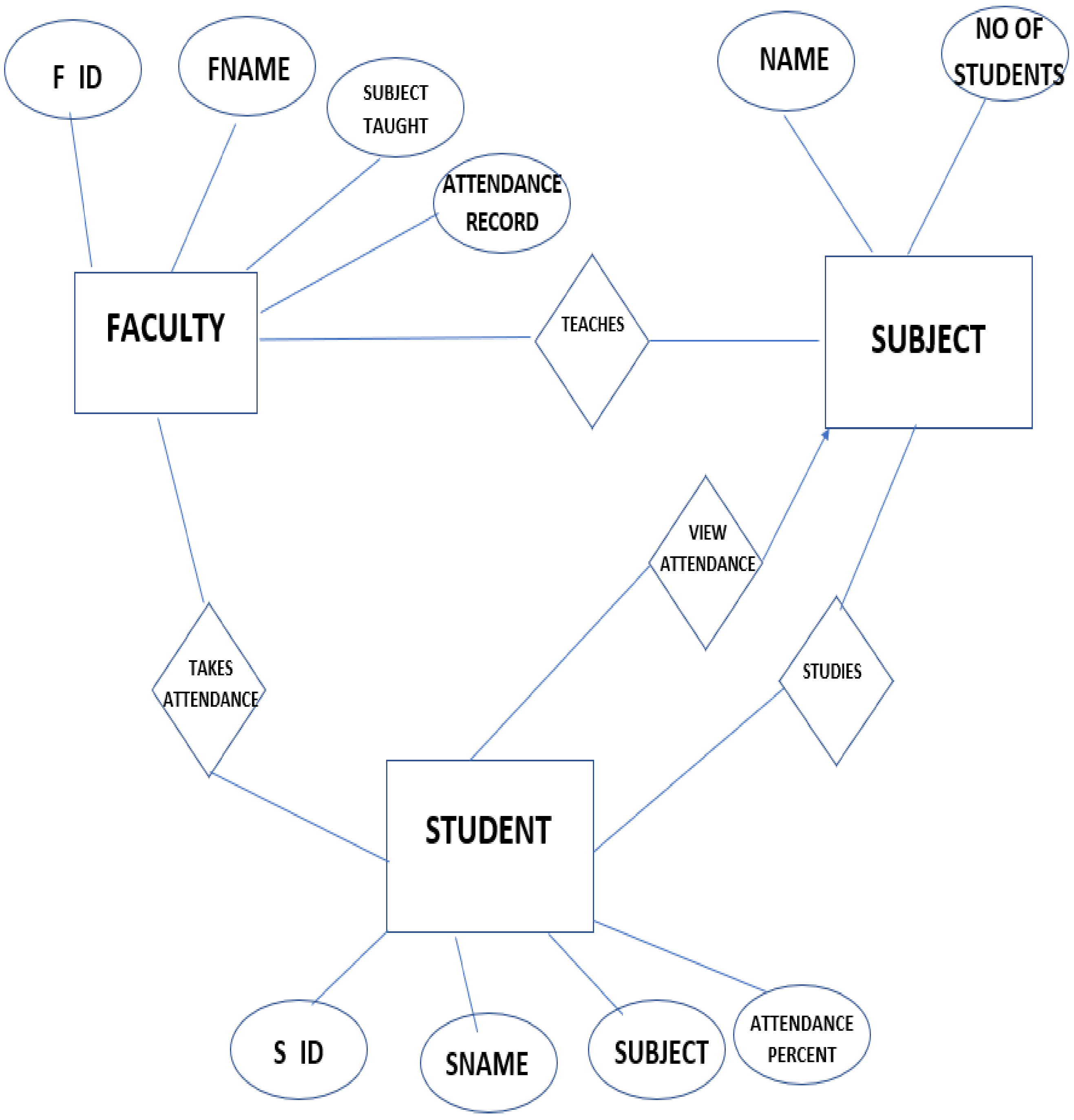
- **Scalability:** The application should be scalable to accommodate a growing user base and increasing data volume without sacrificing performance.
- **Reliability:** The application should be reliable and available for use at all times, with minimal downtime or service disruptions.
- **Security:** The application should implement robust security measures to protect user data, including encryption, secure authentication, and access controls.
- **Cross-platform Compatibility:** The application should be compatible with both Android and iOS platforms, with consistent functionality and user experience across devices.
- **Usability:** The application should be intuitive and easy to use, with a user-friendly interface and clear navigation paths.
- **Accessibility:** The application should be accessible to users with disabilities, with support for screen readers, keyboard navigation, and other accessibility features.

3. **System Constraints:**
    - **Technology Stack:** The application will be developed using React Native for mobile app development and JSON Server for the backend database.
    - **Hardware Requirements:** The application should run smoothly on a wide range of mobile devices, with varying screen sizes, resolutions, and hardware specifications.

Information flow representation



ER-DIAGRAM


F ID
FNAME
SUBJECT TAUGHT
ATTENDANCE RECORD
NAME
NO OF STUDENTS
FACULTY
TEACHES
SUBJECT
VIEW ATTENDANCE
TAKES ATTENDANCE
STUDIES
STUDENT
S ID
SNAME
SUBJECT
ATTENDANCE PERCENT

Method/Technology to be used

REACT NATIVE

React Native an a cutting-edge framework that has redefined the landscape of mobile app development. React Native, powered by Facebook, is a powerful and versatile open-source framework that enables developers to build high-performance mobile applications using familiar web development principles.

At its core, React Native leverages the popular React library, allowing developers to create rich, native-like experiences for both iOS and Android platforms, all while writing code in JavaScript and React. This innovative approach not only accelerates the development process but also facilitates code reuse across platforms, reducing time and effort in maintaining separate code bases.

API CALLS

- To make API Calls in react native we use Jason-server to make connections to a local API using following steps:-

- **Fetch API:** Utilize the built-in fetch API provided by React Native to make HTTP requests to your JSON server. This API is widely supported and provides a simple interface for making network requests.

- **URL Configuration:** Set up the correct URL configuration to point to your JSON server. Ensure that the URL is correctly formatted and includes the necessary route to access the desired data
- **Handling JSON Responses:** As JSON server responds with JSON data, ensure that you handle the response properly. Use the .json() method to parse the response and work with the JSON data in your React Native application
- Both approaches are valid and can be used depending on your preference and coding style

JSON SERVER

- Most developers have heard of the term JSON. It abbreviates for "JavaScript Object Notation". JSON is nothing but an open standard file format and data interchange format. It uses human-

- readable text to store and transfer data objects. It generally consists of two human-readable attributes i.e. value pairs and arrays. JSON is a very common data format having different applications. For example, a web application communicating with a server uses JSON
- In the similar sense, JSON Server is no different. JSON Server is a Node Module that you can use to create demo REST JSON services within a short span of minutes. All we need to do is have a JSON file as sample data.

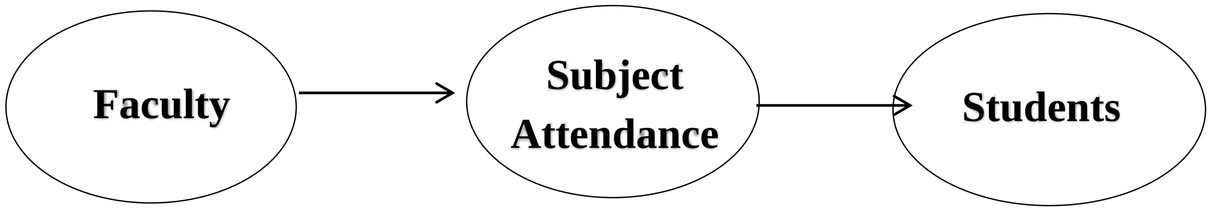
Testing Tools

- For testing purpose, we have used an open-source automation tool DETOX it is used for testing mobile applications across different platforms such as iOS and Android. In a React Native application for attendance management, DETOX is an Grey box End-to-End testing automation framework for mobile apps

- The working of Detox involves creating test scripts in a programming language like JavaScript. These scripts interact with the React Native application just like a real user would, performing actions such as tapping buttons, entering text, and navigating through the app's features. DETOX uses the WebDriver protocol to communicate with the mobile device or emulator, allowing it to control the app's behavior and collect data about its performance

DESIGN

System Architecture



1. **Faculty**

   Faculties are the people who will first sign up on Classkepper. And then at the time of Attendance, they will login and take Attendance of students according to their name and enrollment number via Classkepper.

   Following data from Faculty side will be provided to the system at the time of Login:

- Faculty email address of medicaps
- Password for the email address of medicaps

Faculty responsibilities:

- Login to system
- Take attendance of the students

2. **Subject Attendance**

   Subject Attendance refers to the subject that is automatically assigned to the faculty on the time of login and a list of all the students currently enrolled in that list will be shown one by one to the faculty with their attendance count and total number of lecture being held and also the percentage of that attendance will be shown of every particular student to the student as well as the faculty

   System responsibilities:

- Registration of both faculty and student
- Display of Students attendance and percentage along side with total number of lectures held count of lectures attended by student
- Display of profile info , password changing and modifying student attendance will be available on faculties profile

- Maintaining record for each student and faculty without making any duplicate record.

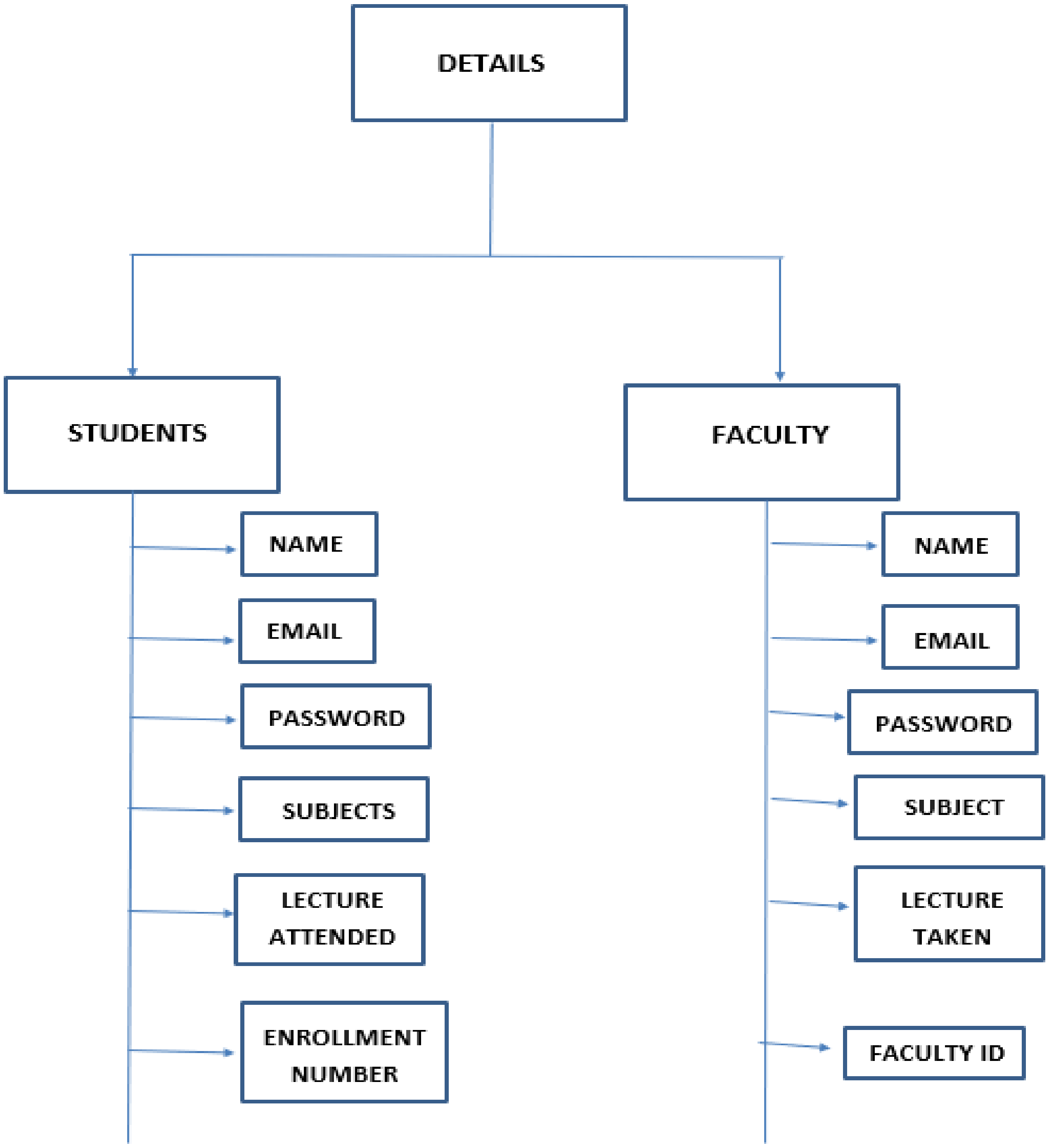
3. **Students**

   Students are those who are enrolled to the subjects and have given some attendance on the basis of how many lectures they have attended with respect to the total lectures taken by the faculty

   Following data from Students side will be provided to the system

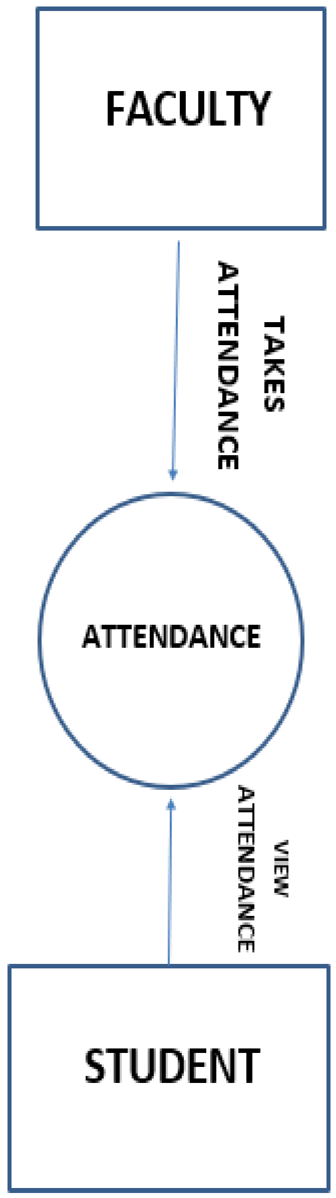
- Name
- Email id of medicaps
- Password for medicaps email id
- Count of total lecture attended of a particular subject

DATABASE STRUCTURE



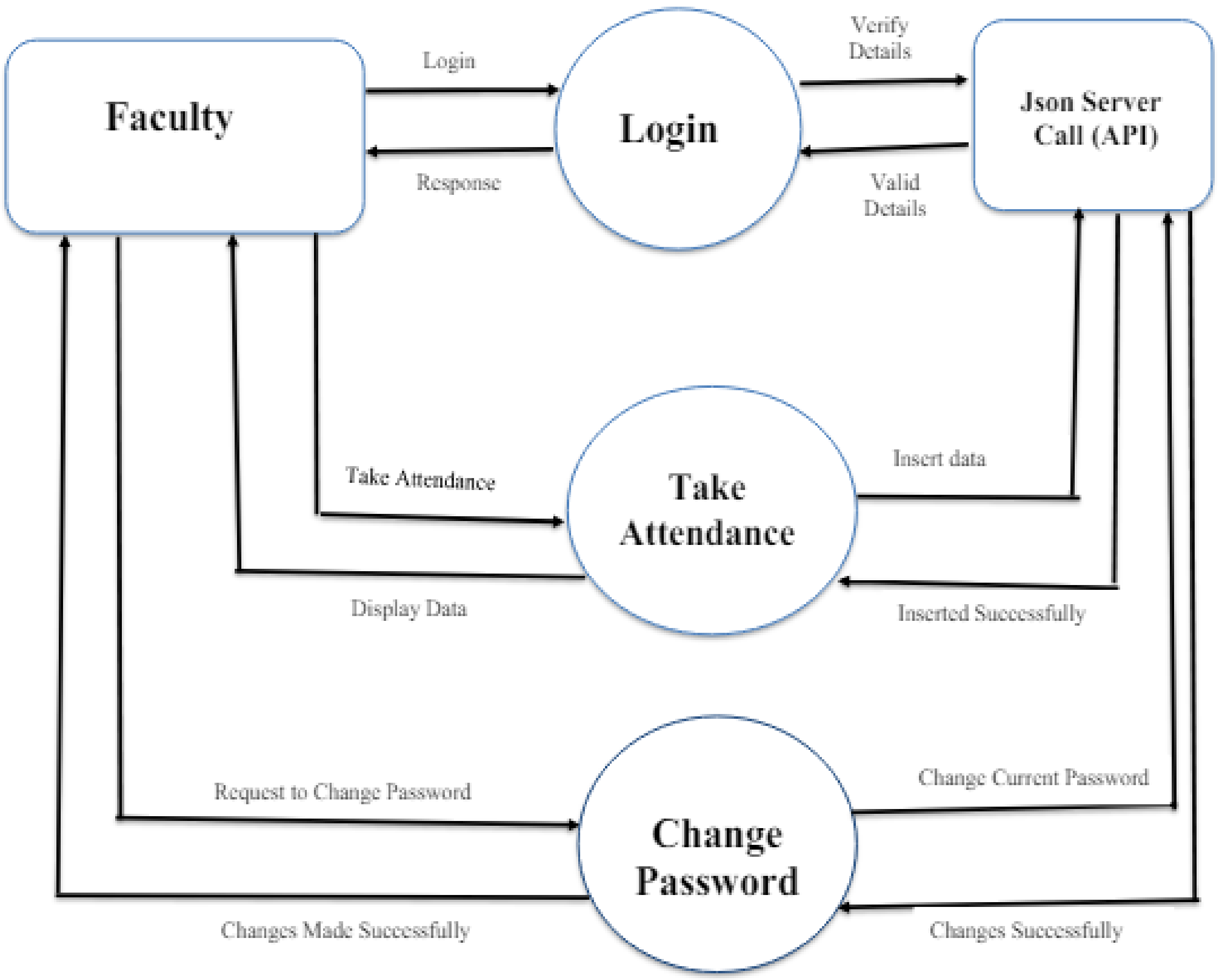
Data Design

DFD (DATA FLOW DIAGRAM)



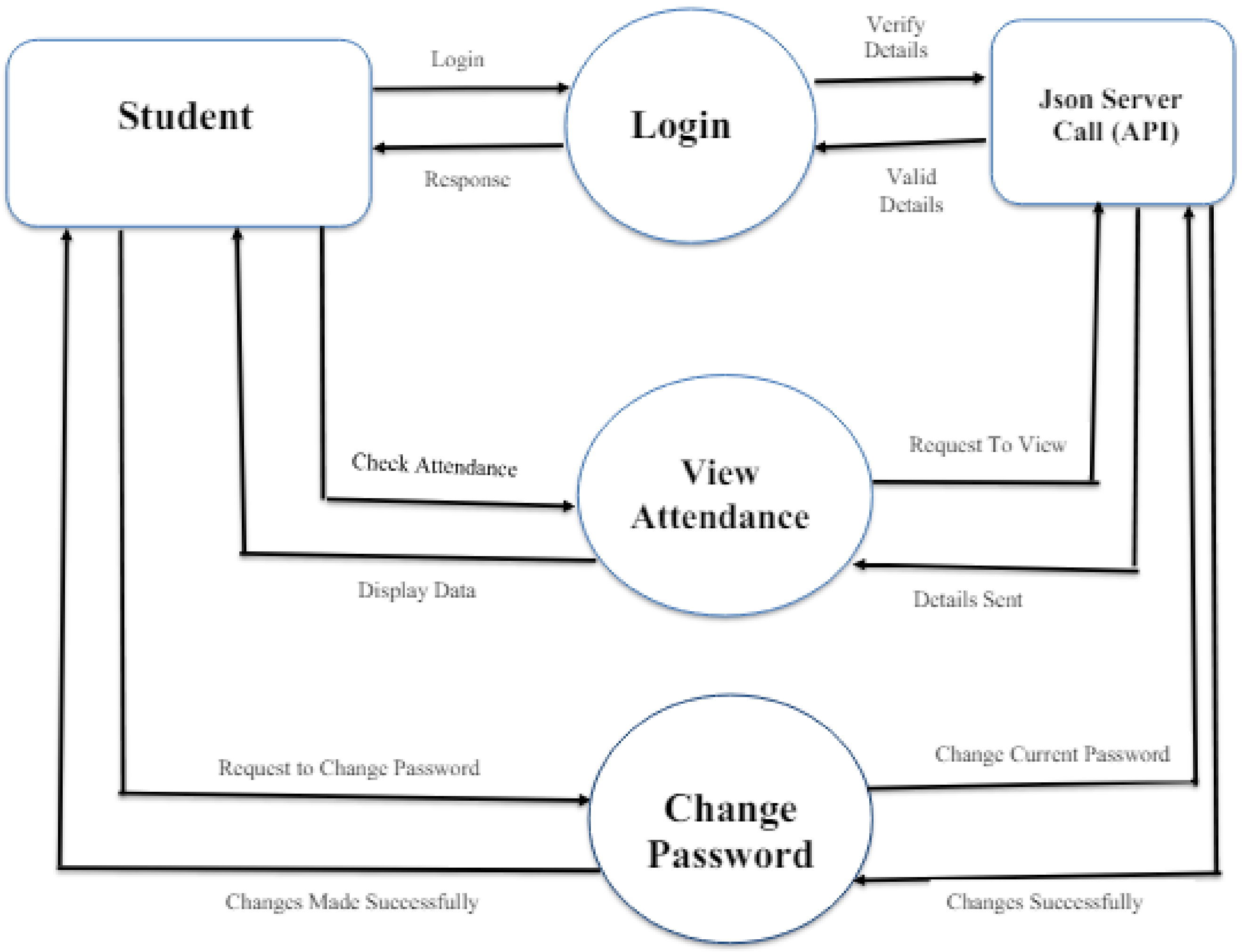
Data Flow Diagram

(Faculty)

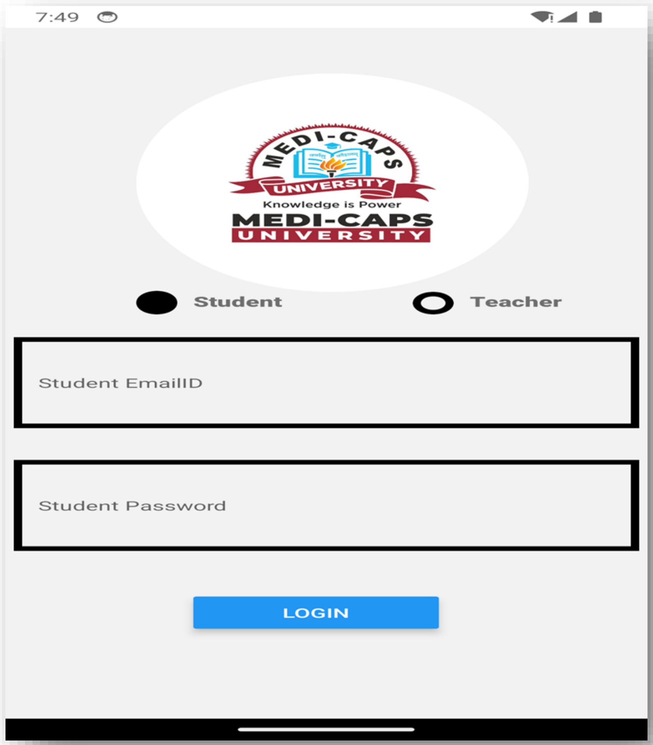


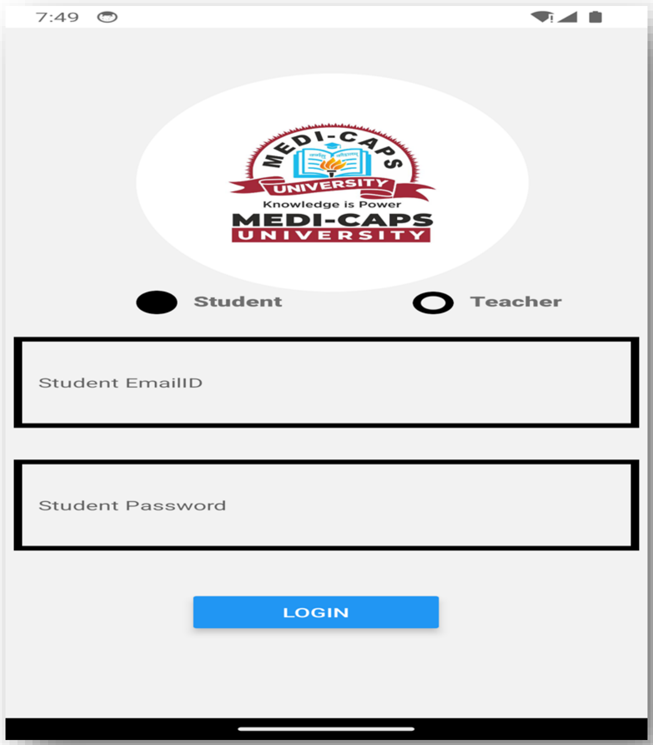
UML, And UI Interface

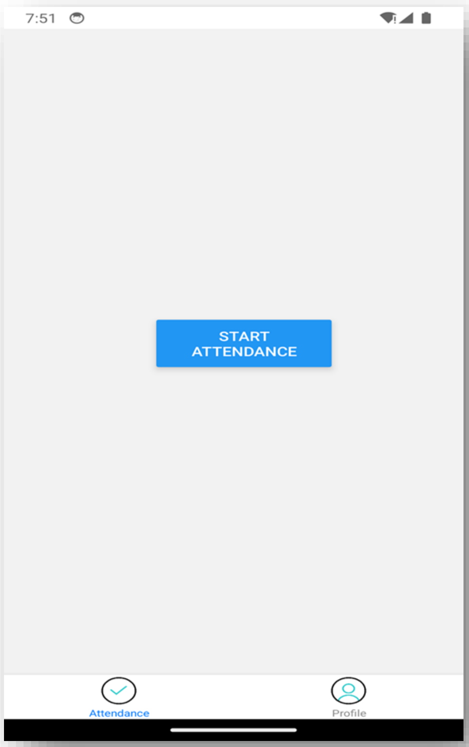
Data Flow Diagram (Student)

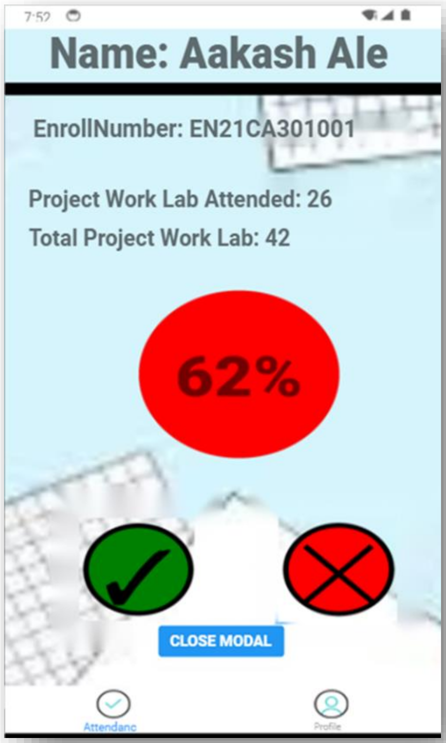


Interface Design







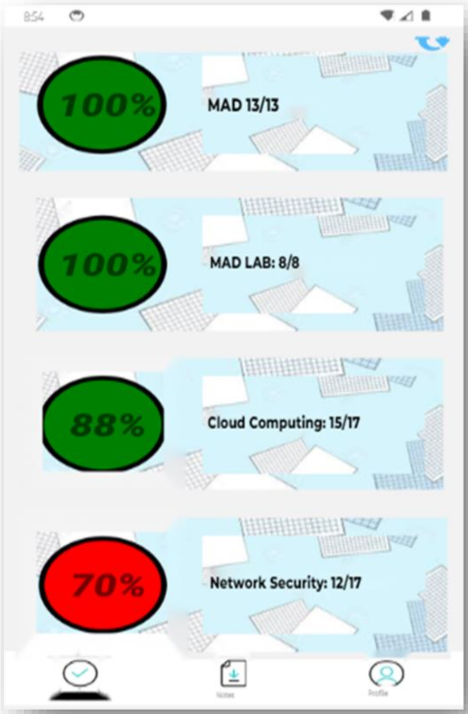


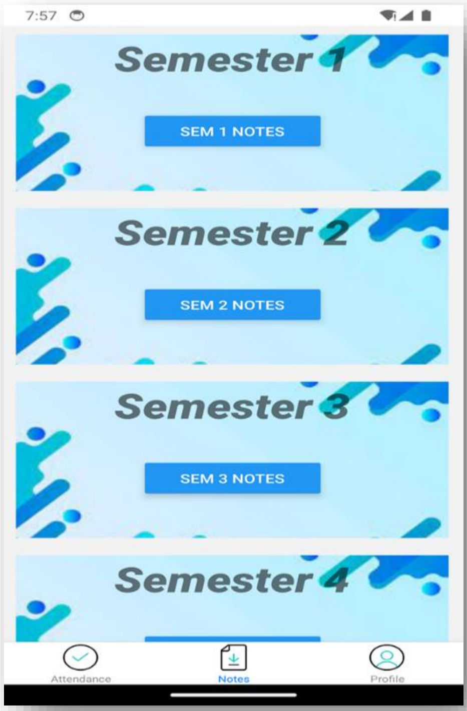
Name: Aakash Ale

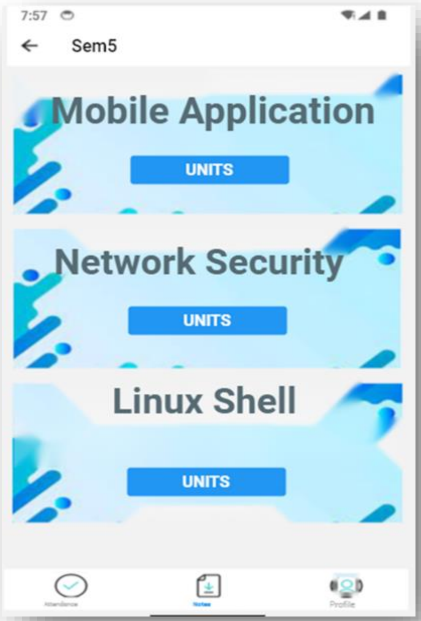
EnrollNumber: EN21CA301001

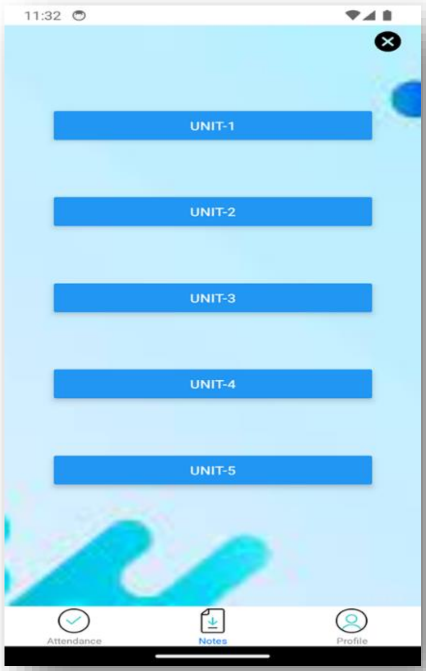
Project Work Lab Attended: 26

Total Project Work Lab: 42


8:54
100%
MAD 13/13
100%
MAD LAB: 8/8
88%
Cloud Computing: 15/17
70%
Network Security: 12/17


7:57
Semester 1
SEM 1 NOTES
Semester 2
SEM 2 NOTES
Semester 3
SEM 3 NOTES
Semester 4
Attendance
Notes
Profile


7:57
Sem5
Mobile Application
UNITS
Network Security
UNITS
Linux Shell
UNITS
Attendance
Notes
Profile


11:32
UNIT-1
UNIT-2
UNIT-3
UNIT-4
UNIT-5
Attendance
Notes
Profile

Testing

Scope of Testing/Test plan

1. **Unit Testing**:
   - **React Native Components**: Verify individual components (screens, buttons, forms) to ensure they function correctly.
   - **Redux Actions and Reducers**: Test state management logic using unit tests.
   - **JSON Server API Calls**: Validate that API requests and responses work as expected.

2. **Integration Testing**:
   - **Component Interaction**: Test how different components interact with each other.
   - **Redux Integration**: Ensure Redux actions and reducers work seamlessly together.
   - **API Integration**: Validate communication between the app and the JSON Server.

3. **End-to-End (E2E) Testing**:
   - **User Flows**: Test complete user flows, such as logging in, marking attendance, and adding notes.
   - **Navigation**: Verify navigation between screens.

4. **Functional Testing**:
   - **Attendance Management**:
     - Verify accurate attendance recording.
     - Test handling of edge cases (e.g., duplicate entries, invalid data).
   - **PDF Notes**:

- Ensure proper storage and retrieval of PDF notes.
- Validate rendering of bullet points within notes.

5. **Performance Testing**:
   - **Load Testing**: Assess how the app handles concurrent users during attendance submission.
   - **Response Time**: Measure API response time for attendance and note-related requests.

6. **Security Testing**:
   - **Authentication and Authorization**:
     - Verify secure login and session management.
     - Test role-based access control (e.g., admin vs. student).
   - **Data Privacy**:
     - Ensure sensitive data (e.g., attendance records) is properly secured.
     - Validate PDF note storage security.

7. **Usability Testing**:
   - **User Experience (UX)**:
     - Evaluate app usability, navigation, and responsiveness.
     - Gather feedback from potential users.
   - **Accessibility**:
     - Test screen readers, font sizes, and color contrasts.
     - Ensure compliance with accessibility standards.

Sample test data and results

1. **Positive Scenario: Marking Attendance Successfully**
   - **Test Steps**:
     1. Open the **ClassKepper** app.
     2. Navigate to the session with Subject
     3. Click the “Mark Attendance” button.
   - **Expected Result**: The attendance for the session is successfully marked, and the count of total lectures and percentage of attendance changes

2. **Negative Scenario: Marking Duplicate Attendance**
   - **Test Steps**:
     1. Open the **ClassKepper** app.
     2. Navigate to the session with Subject
     3. Click the “Mark Attendance” button twice.
   - **Expected Result**: The app prevents duplicate attendance marking and displays the name and enrollment number of next student

3. **Negative Scenario: Modifying Attendance of a Student and increased lecture attended greater then total lecture taken**
   - **Test Steps**:
     1. Open the **ClassKepper** app.
     2. Attempt to modify attendance for the session of a particular Student
   - **Expected Result**: The app displays an error message indicating that the Count is invalid

LIMITATIONS

- **Limited Platform Support**: As of now, the app is developed for iOS and Android platforms. It does not support other operating systems.

- **Device Compatibility**: The app may not work optimally on older devices with limited resources (e.g., memory, processing power).

- **Localization Challenges**: The app only supports English language

- **Security Risks**: Although the app uses JSON Server as a mock backend, real-world deployment would require robust security measures (e.g., encryption, secure APIs).

- **Scalability**: The app’s performance under heavy load (many users simultaneously) needs further testing and optimization.

- **PDF Rendering**: Displaying PDF notes within the app may face challenges related to rendering, especially for large files.

- **Usability on Small Screens**: The app’s usability on smaller screens (e.g., smartphones) should be thoroughly tested.

- **Data Synchronization**: Ensuring consistent data synchronization between the app and the server can be challenging.

SUMMARY AND CONCLUSIONS

Summary

ClassKepper is a versatile mobile application designed to streamline attendance management and note storage. Developed using **React Native**, it caters to both students and instructors, enhancing classroom efficiency.

1. **Attendance Management**:
   - Students can mark attendance for specific class sessions.
   - The app prevents duplicate entries and ensures real-time updates.
   - Accurate attendance records are maintained.

2. **PDF Note Storage**:
   - Users can upload and store PDF notes within the app.
   - Retrieval and display of notes are seamless.
   - Bullet points enhance readability.

3. **Integration and Testing**:
   - Components interact seamlessly due to Redux integration.
   - Rigorous testing covers functionality, performance, and security.
   - Regression testing prevents unintended regressions.

4. **Usability and Accessibility**:
   - The app prioritizes user experience (UX) with responsive design.
   - Accessibility features cater to diverse users, including screen readers.

Conclusion

The implementation of the Attendance and Notes Managing app is a substantial project that aims to enhance educational processes through streamlined attendance management and effective note-taking capabilities. With careful planning, dedicated development efforts, and a strong focus on security and usability, the app has the potential to significantly benefit educational institutions, teachers, students, and parents alike. It's important to adapt the time frame estimates based on the specific needs and constraints of the project, ensuring that quality and functionality are not compromised in the pursuit of efficiency.

Future Scope

FACULTY ACCESS

- Faculty would be given permission for uploading notes of their subjects along with the assignments
- Faculty would also be able to take live quizzes on the App itself and after the quiz is completed student score will be shown to the student and the response will be shown to faculty profile
- Faculty will be able to download all the submitted assignments form the app and also provide a due date after which the assignment submission will be disabled

STUDENT ACCESS

- Student can submit their assignments on the app in PDF format by scanning the document or directly uploading it from their google drive
- Students can take live quiz and the reminder of quiz and assignment will be shown to the student via App notification

Appendix

1. **Development Resources**:
   - **React Native Documentation**:
     - Explore the official React Native documentation for comprehensive guides, components, and APIs.
     - Learn about navigation using **React Navigation**.
   - **JSON Server**:
     - Refer to the JSON Server GitHub repository for setting up a mock backend.
     - Utilize mock API data for testing and development.

2. **Testing Strategies**:
   - **Unit Testing**:
     - Use **Jest** to test React Native components and Redux actions.
     - Mock API calls to JSON Server for isolated testing.
   - **Integration Testing**:
     - Validate component interactions and Redux integration.
     - Test API endpoints (CRUD operations).
   - **End-to-End (E2E) Testing**:
     - Consider tools like **Detox** or **Appium** for testing complete user flows.

3. **Security Considerations**:
   - **Authentication**:
     - Implement secure login mechanisms (e.g., JWT tokens, OAuth).
     - Protect against unauthorized access.
   - **Data Encryption**:

- Secure sensitive data (e.g., attendance records) using encryption.
- Ensure secure communication between the app and the server.

4. **Performance Optimization**:
   - **Network Efficiency**:
     - Optimize API requests (batching, caching) to handle slow or intermittent connections.
   - **Code Splitting**:
     - Split large bundles for faster initial app loading.
     - Use dynamic imports where possible.

5. **Future Enhancements**:
   - **Push Notifications**:
     - Implement notifications for upcoming classes or important announcements.
     - Integrate with services like Firebase Cloud Messaging (FCM).
   - **Offline Mode**:
     - Allow users to access cached data when offline.
     - Sync data once the network is available.
   - **Analytics and Monitoring**:
     - Integrate tools like **Google Analytics** or **Sentry** for monitoring app usage and detecting issues.

Bibliography

1. React Native Documentation:

- Author: Facebook, Inc.
- URL: https://reactnative.dev/docs/getting-started
- Description: Official documentation for React Native, providing essential information, tutorials, and examples for app development.

2.JavaScript: The Good Parts (Book):

- Author: Douglas Crockford
- Publisher: O'Reilly Media
- Year: 2008
- Description: A foundational book on JavaScript, which is the programming language underlying React Native.

3.JSON.org:

- Author: Douglas Crockford
- URL: https://www.json.org/
- Description: The official website for JSON, providing information on the JSON data
- format, which is commonly used in App Dev