

## CONTENT

<b>S.NO</b>	<b>INDEX</b>	<b>Page No</b>
1	<b>INTRODUCTION</b>	1
2	<b>SYSTEM STUDY</b>	2
3	<b>SYSTEM ANALYSIS</b>	4
4	<b>SYSTEM DESIGN</b>	8
5	<b>SYSTEM DEVELOPMENT</b>	14
6	<b>TESTING</b>	19
7	<b>USER MANUAL</b>	24
8	<b>CONCLUSION</b>	27
	<b>BIBLIOGRAPHY</b>	29
	<b>APPENDIX</b>	30

# CHAPTER 1

## INTRODUCTION

### 1.1 About the Project

The **Lab Mark Entry Application** is a mobile application that helps staff to enter internal marks for software labs courses. Developed using flutter and firebase, it helps the staff to store the internal marks digitally and convert them into pdf format at any point of time.

For all staff members, this application has secure log in mechanism with forgot password functionality. Upon successful login user is taken to the homepage of the application where they can create courses by providing the required course name, starting D. No and ending D. No. After successfully creating a course, the particular course will be visible in the homepage of the staff who created the course.

After successfully setting up the course it will be available in the homepage. To enter marks for that course the staff need to click the desired course from the homepage. Upon clicking the course name, staff will be directed to mark entry screen which will show the following options.

- Exercise number
- Date of the lab
- Student D.no
- Preparation mark
- Viva-voce
- Next D. No
- Previous D. No

After entering the marks staff can download the stored marks into a structured pdf file by clicking the download button available in the homepage.

## **CHAPTER 2**

### **SYSTEM STUDY**

#### **2.1 Existing System**

In the past, staff members had to rely on physical paper to record student's marks during evaluations. This method was widely accepted and effective at the time, primarily because internet access was not consistently available. Without a reliable online system, carrying physical mark sheets ensured that staff could document student performance without any technological dependency.

However, times have changed, and technology has become an integral part of education and data management. In today's digital era, individuals prioritize digital records over physical documents due to their numerous advantages. Digital data is more secure, easily accessible, and can be stored for a lifetime without the risks associated with physical records, such as misplacement, damage, or deterioration over time. Additionally, digital storage allows for quick retrieval, editing, and sharing of information, making it a more efficient and convenient alternative to traditional paper-based systems.

#### **2.2 Disadvantages of Existing System**

If the staff member carrying the specific mark sheet becomes unreachable due to unforeseen circumstances, such as being out of contact, unavailable, or facing an emergency, there is no alternative way to retrieve the recorded student marks until they are accessible again. This can cause delays in processing and verifying student's academic records.

Furthermore, if the physical mark sheet is misplaced, stolen, or damaged due to external factors like water spills, fire, or wear and tear, the recorded data is permanently lost. Unlike digital records, which can be backed up and retrieved from a secure database, physical documents are vulnerable to loss and deterioration, making them unreliable for long-term data storage.

#### **2.3 Proposed System**

The proposed Lab Mark Entry application addresses the limitations of the traditional paper-based system by leveraging cloud storage to securely store student's marks. Instead of relying

on physical records that can be lost, damaged, or become inaccessible, this application ensures that all data is safely stored online, making it available anytime and from anywhere with internet access.

One of the key advantages of this system is its ability to enhance collaboration among staff members. For courses that require multiple evaluators, staff can work together seamlessly, accessing and updating student marks in real time. By utilizing Firebase cloud storage, the application enables instant synchronization, ensuring that any updates made by one staff member are immediately visible to others. This eliminates the need for manual data transfers and reduces the chances of data loss, ultimately streamlining the entire mark entry and evaluation process.

## 2.4 Advantages of Proposed System

- Cloud storage for storing and retrieving marks
- Ability to collaborate with other staffs
- Ability to download the marks into PDF format
- Data security by google

## 2.5 Problem Definition and Description

The Mark-Entry application is a mobile based application built using flutter and firebase database. This application allows staff to enter mark digitally and store them into cloud storage. This application allow multiple staff to collaborate with other staffs to enter mark efficiently. After completing all the exercises the staff can download the stored marks into a pdf file which will be structured.

The system consists of following modules:

- **Staff Login:** Provides secure access for staff members to enter the system.
- **Marks Entry:** Allows staff to easily enter marks into cloud storage.
- **Create Course:** Enables staff to create courses and update marks after storing them.
- **Collaboration Functionality:** Allows multiple staff members to work on the same course.

# CHAPTER 3

## SYSTEM ANALYSIS

### 3.1 Packages Selected

Front end : Dart language (Flutter)

Back end : Firebase (NoSQL)

### 3.2 Resources Required

#### 3.2.1 Hardware Resources

Processor: Intel Core i3 (8th Gen or later) / AMD Ryzen 3

RAM: 8GB

Storage: 10GB free SSD space

#### 3.2.2 Software Resources

IDE (Integrated Development Environment): Visual Studio Code


Database Management System: Firebase

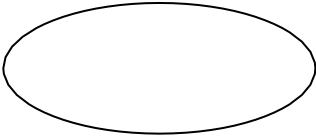
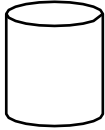


Operating System: Windows or linux

### 3.3 Data Flow Diagram

The data flow diagrams illustrate the flow of information within the Lab Mark Entry Application. They provide a visual representation of how data moves between different modules and processes, offering insight into the system's structure and functionality in a clear and concise manner.

Data Flow Diagram Symbols:

Symbol	Description
	An entity. A source of data or a destination for data.

	A process or task that is performed by the system
	A data store is a place where data is held between processes.
	A data flow
	Data flow in both direction

### Level 0

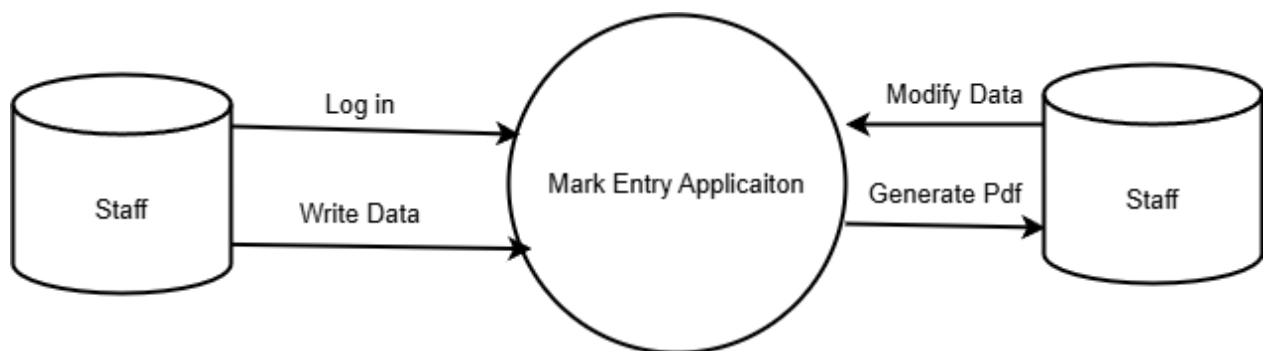


Fig: 3.3.1 Level 0

This diagram represents the data flow and interaction between staff members and the Mark Entry Application, which uses a NoSQL firebase real time database for data storage. The two cylindrical symbols on both sides represent Firebase databases storing staff-related information, course details, and student marks. The central circle represents the Mark Entry Application, which allows staff to input, update, and retrieve marks from the Firebase database in real time.

## Level 1

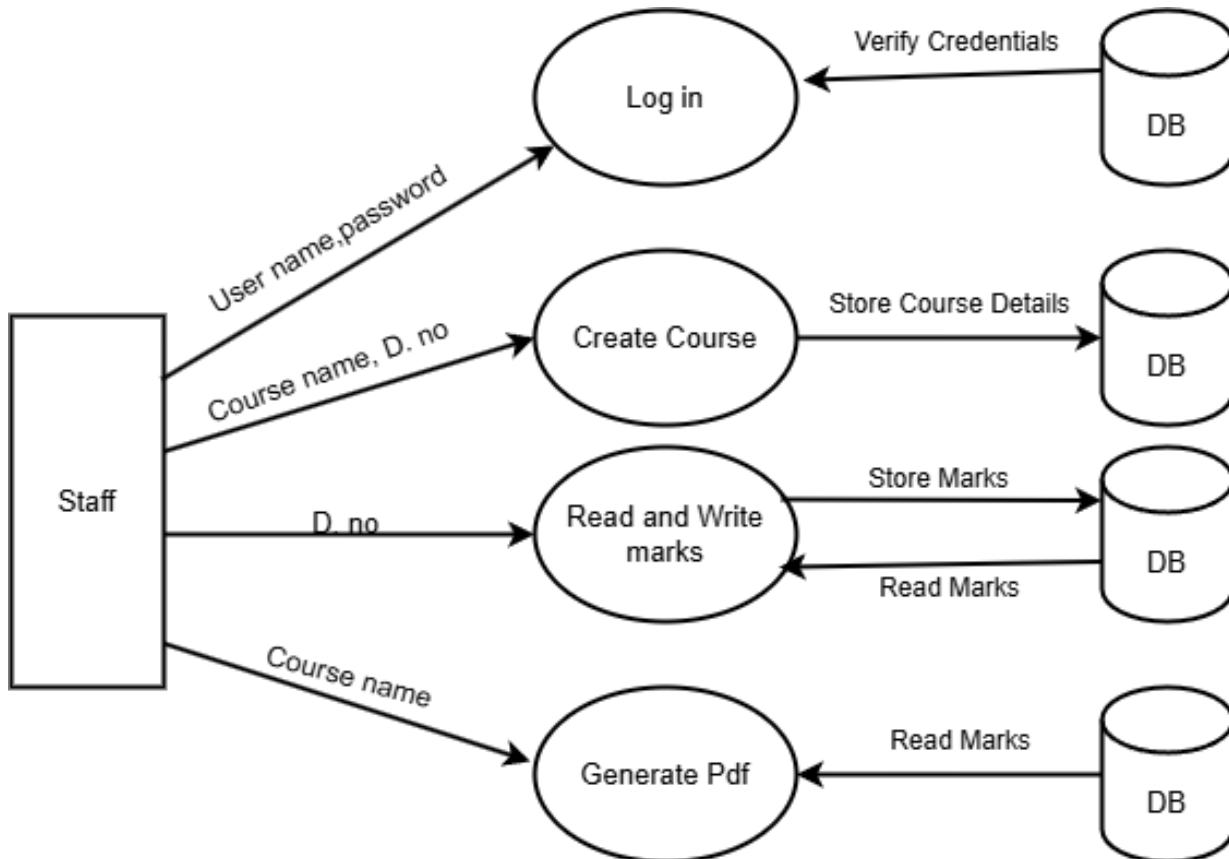


Fig: 3.3.2 Level 1

This diagram shows how staff members interact with the Firebase NoSQL database in the Lab Mark Entry Application. Staff can log in using firebase authentication, ensuring secure access. They can create courses by entering details like the course name and student range, which are stored in the database.

After setting up a course, staff can read and write marks, updating and retrieving student scores from firebase real time database. The system also allows staff to generate a PDF report to download and store student marks. The databases (DB) on the right represent Firebase, where each operation stores or fetches data. This setup ensures efficient data handling and real-time updates.

## Level 2

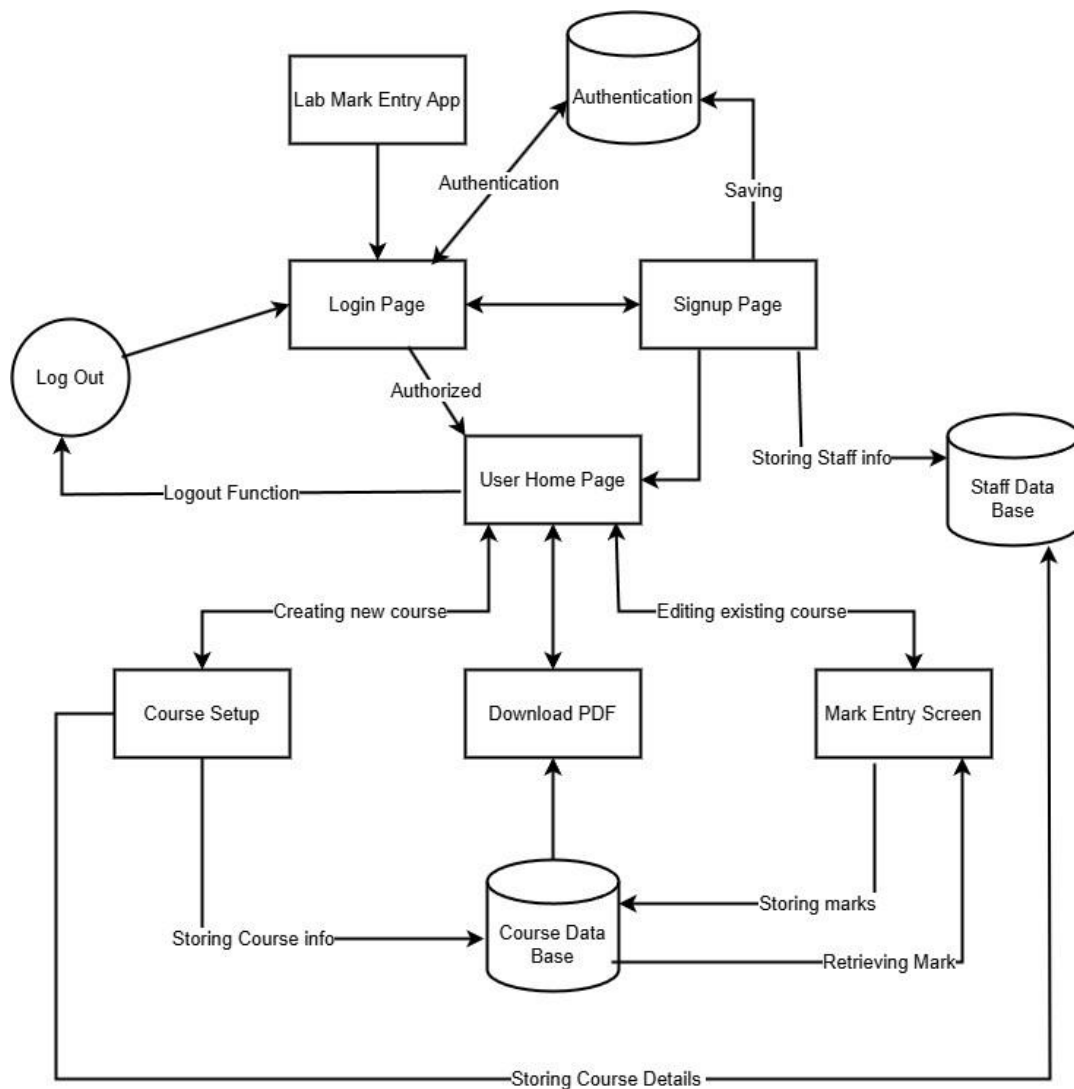


Fig: 3.3.3 Level 2

This diagram represents the workflow of the Lab Mark Entry Application in detail. The process starts with staff members either logging in or signing up, with authentication handled by a dedicated database. Once authenticated, new staff details are stored in the Staff Database. The User Home Page acts as the central navigation point, allowing staff to create new courses, edit existing courses, enter marks, and download reports in PDF format. The Course Database stores course details and marks entered by staff, while the Mark Entry Screen retrieves and updates student marks. The Logout Function ensures secure exit from the system. This structured approach enables a smooth and efficient mark management system for staff members.



# CHAPTER 4

## SYSTEM DESIGN

### 4.1 Architectural Design

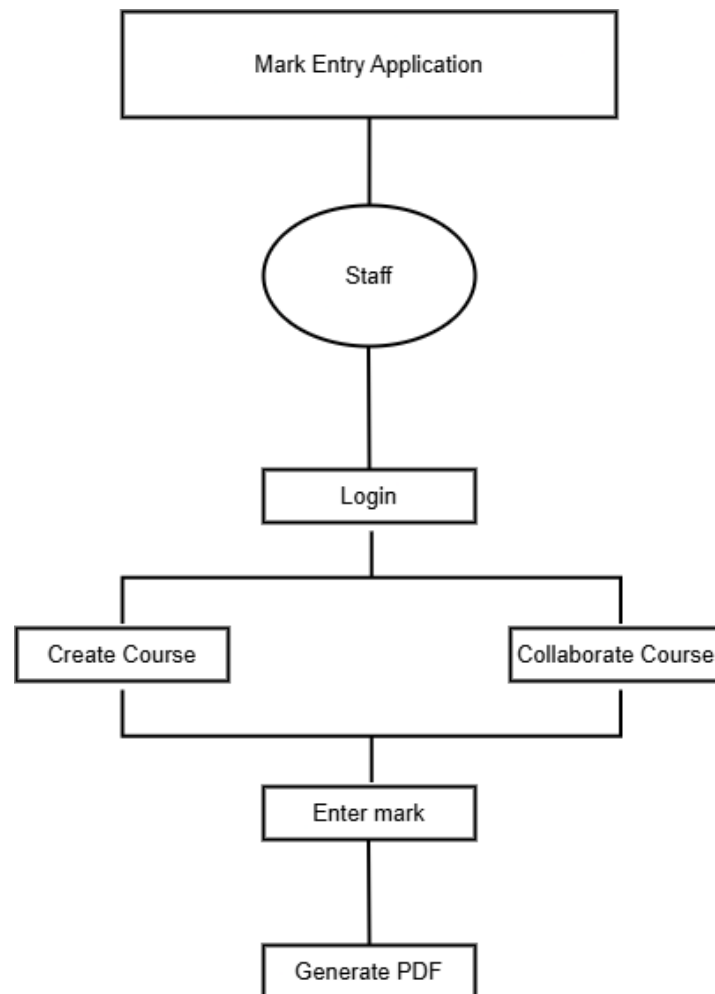


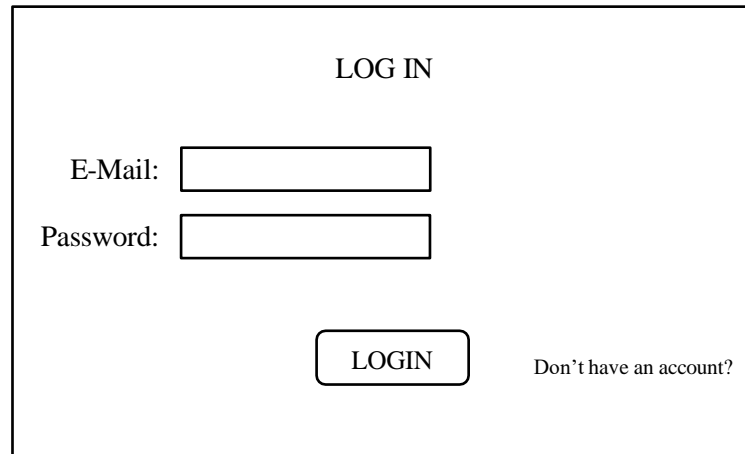
Fig: 4.1.1 Architectural Design

This architecture design represents the structure and workflow of the Mark Entry Application. The system is designed for staff members, who access the application to manage student marks efficiently.

The process starts with staff logging into the system, ensuring secure access. Once logged in, they have the option to create a course or collaborate on an existing course. After setting up a course, staff can enter marks for students. The system ensures that all mark entries are stored efficiently. Finally, staff can generate a PDF report, which allows them to save and download the entered marks for record-keeping.

## 4.2 I/O Forms

### 4.2.1 Login Form

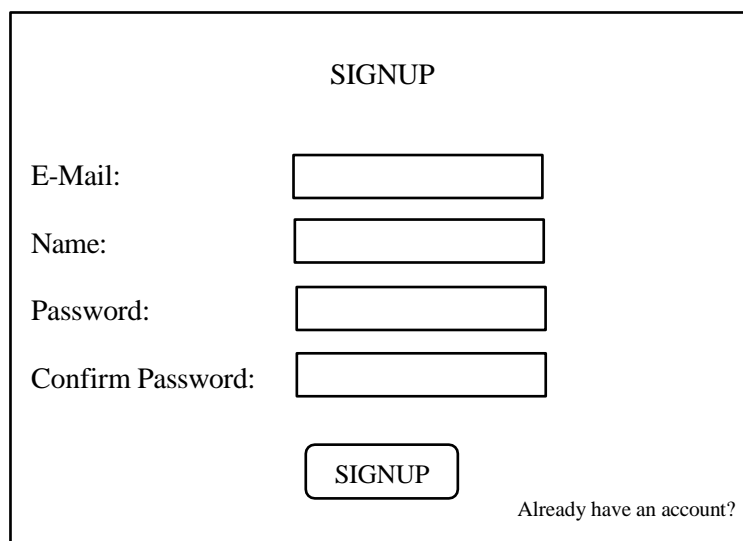


A rectangular form titled "LOG IN" at the top center. Below the title, there are two input fields: "E-Mail:" followed by a text box, and "Password:" followed by a text box. Below these fields, there is a rounded rectangular button labeled "LOGIN". To the right of the button, there is a link that says "Don't have an account?".

Fig: 4.2.1.1 Staff Login page

This page enables staff members to login using their credentials. This page requires staff to enter their registered email and the password created. Authorization takes place when login button is clicked, when the provided credentials are correct the application moves to home screen.

### 4.2.2 Signup Form

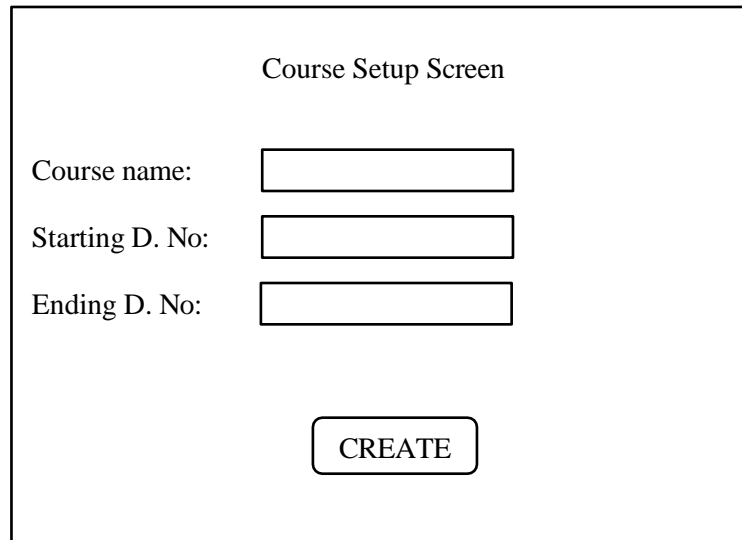


A rectangular form titled "SIGNUP" at the top center. Below the title, there are four input fields: "E-Mail:" followed by a text box, "Name:" followed by a text box, "Password:" followed by a text box, and "Confirm Password:" followed by a text box. Below these fields, there is a rounded rectangular button labeled "SIGNUP". To the right of the button, there is a link that says "Already have an account?".

Fig: 4.2.2.1 Staff Signup page

This page enables staff members to Sign up by providing their details to access the application. When signup button is clicked, User data are stored into the database.

#### 4.2.3 Course Setup Screen:

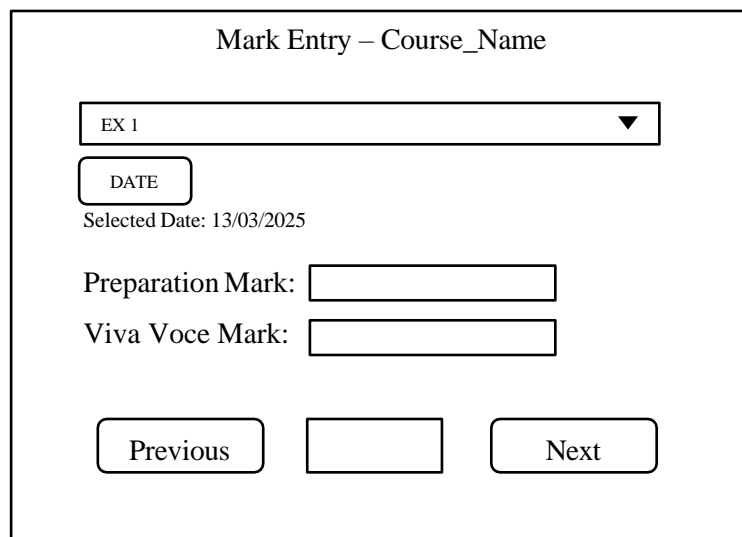


The Course Setup Screen is a web form with a title "Course Setup Screen" at the top center. Below the title, there are three input fields: "Course name:", "Starting D. No:", and "Ending D. No:". Each field is represented by a text label followed by a rectangular input box. At the bottom center of the form is a rounded rectangular button labeled "CREATE".

Fig: 4.2.3.1 Course Setup Screen

The system allows staff to define a course by entering the course name along with the starting and ending D.no. Upon entering the course name, D. No and clicking the submit button, the course is created and stored into the database

#### 4.2.4 Mark Entry Screen:

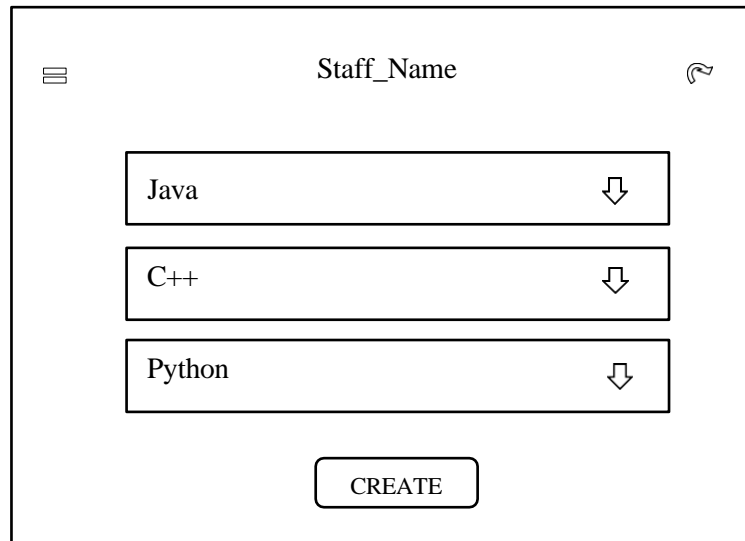


The Mark Entry – Course\_Name screen is a web form. At the top center is the title "Mark Entry – Course\_Name". Below the title is a dropdown menu showing "EX 1" with a downward arrow. Under the dropdown is a rounded rectangular button labeled "DATE". Below the "DATE" button, the text "Selected Date: 13/03/2025" is displayed. Further down are two input fields: "Preparation Mark:" and "Viva Voce Mark:", each followed by a rectangular input box. At the bottom of the form are three rounded rectangular buttons: "Previous", a blank button, and "Next".

Fig: 4.2.4.1 Mark Entry Screen

The system allows staff to choose the exercise number, date, and D.no for entering marks into the database. When user clicks next or previous button the marks are updated to the database.

#### 4.2.5 Staff Homepage

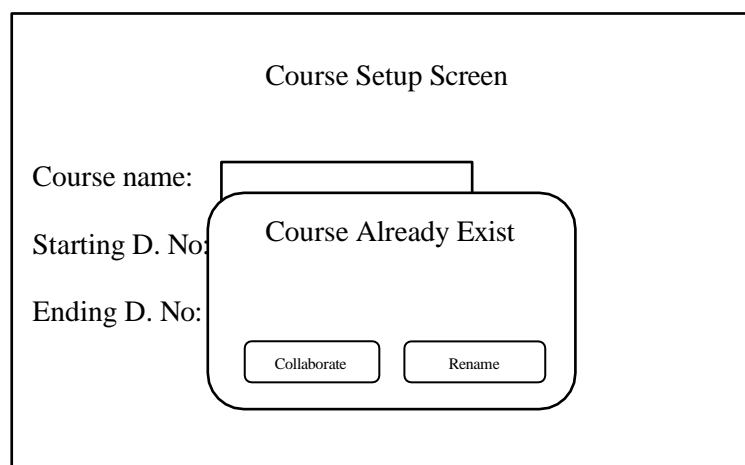


The Staff Homepage UI mockup features a header with a hamburger menu icon on the left, the text "Staff\_Name" in the center, and a refresh icon on the right. Below the header, there are three vertically stacked input fields. The first field contains the text "Java" and a download icon (a downward arrow inside a square). The second field contains "C++" and a similar download icon. The third field contains "Python" and a similar download icon. At the bottom center of the page is a rounded rectangular button labeled "CREATE".

Fig: 4.2.5.1 Homepage

Staff can view the courses created in the homepage and can download the stored marks into a pdf file by clicking the download button. By clicking create button staff can create a new course.

#### 4.2.6 Collaborate Functionality



The Course Setup Screen UI mockup has a title "Course Setup Screen" at the top. Below the title, there are three labels: "Course name:", "Starting D. No:", and "Ending D. No:". To the right of these labels are input fields. A modal dialog box is overlaid on the input fields. The dialog has a title "Course Already Exist" and two buttons at the bottom: "Collaborate" and "Rename".

Fig: 4.2.6.1 Course Setup Page

The system allows a course to be shared among multiple staff members for effective collaboration and real-time mark entry. When staff clicks collaborate button, the course will be added to the staff's homepage.

### 4.3 Table Design

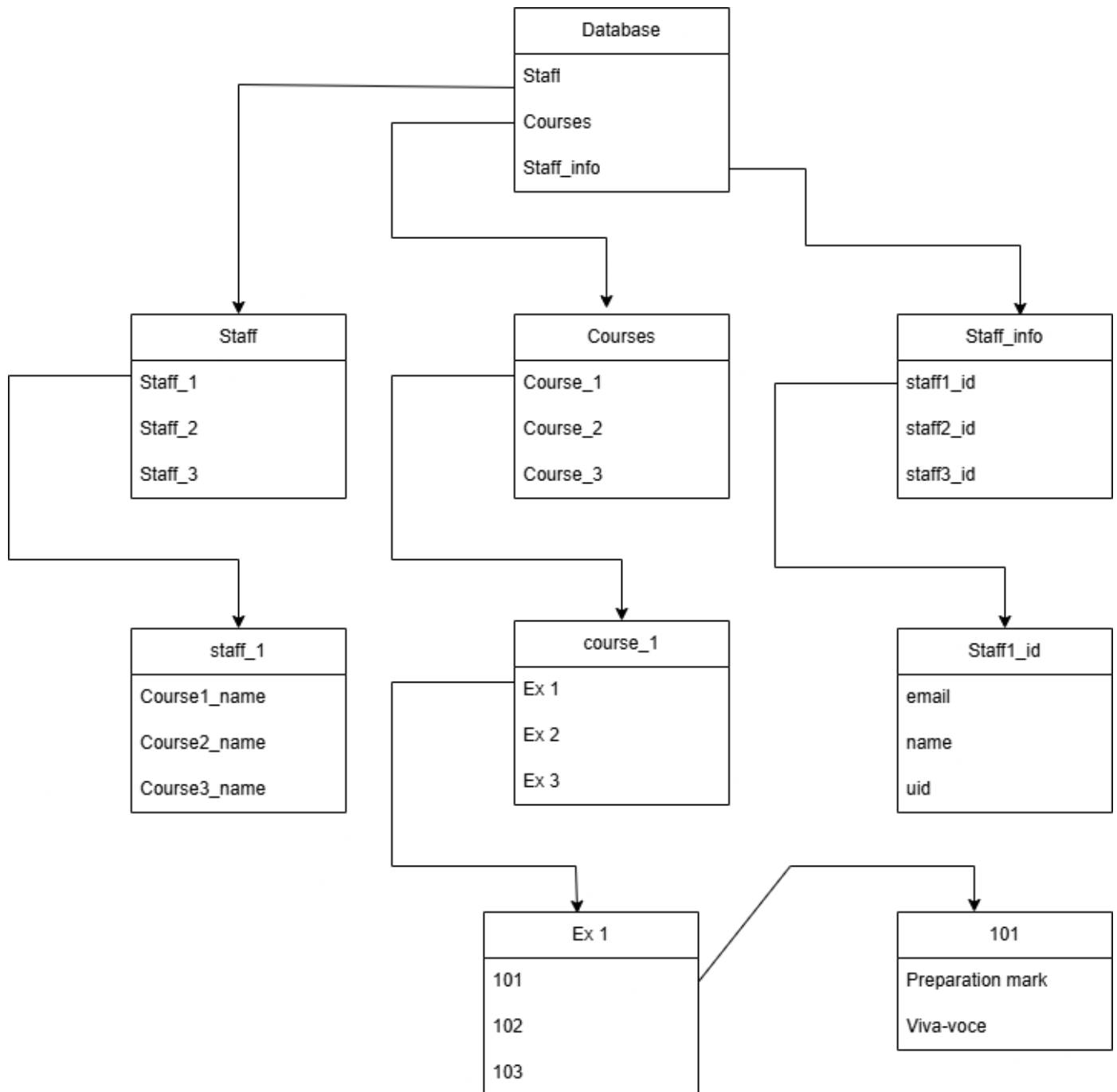

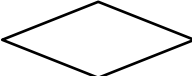



Fig: 4.3.1 Table Design

This table design represents the NoSQL firebase database structure for the Mark Entry Application. It organizes data into staff, courses, and staff information, linking them through unique IDs. Each course contains exercises, and each exercise stores student marks for preparation and viva-voce.

## 4.4 ER Diagram

ER Diagram Symbols:

Symbols	Description
	External Entity
	Relationship
	Attribute

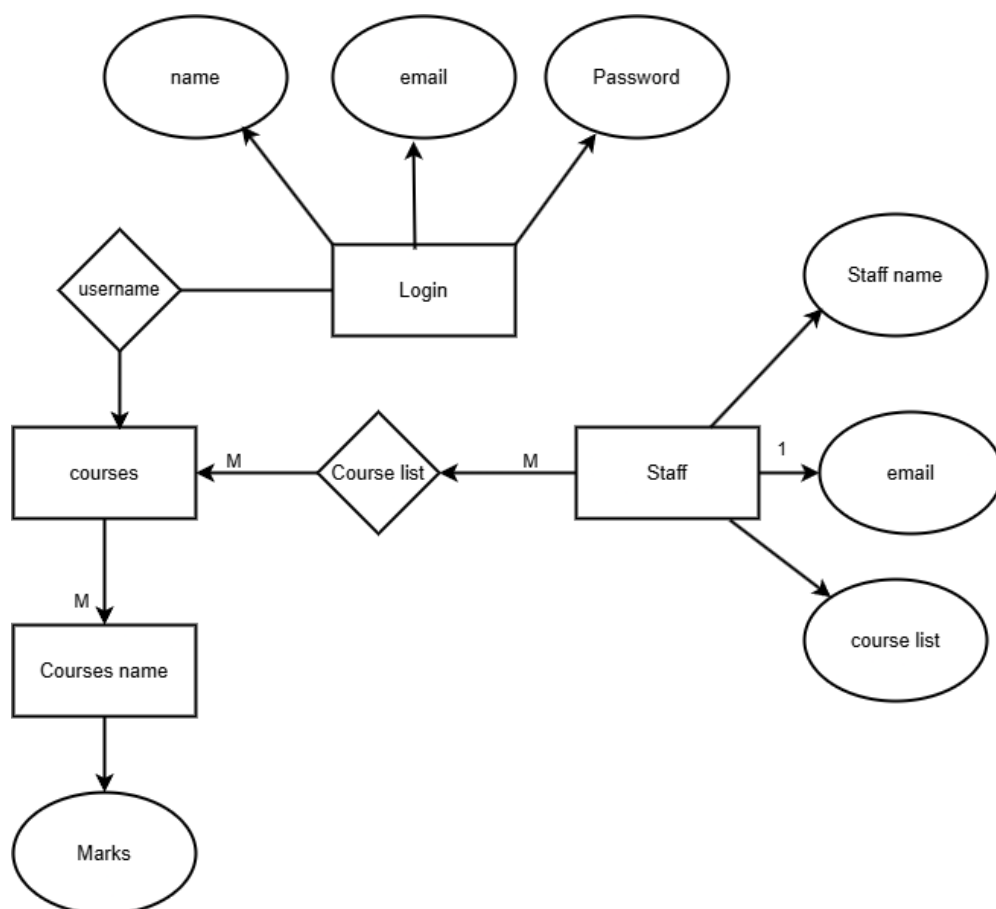


Fig: 4.4.1 E R Diagram

This ER diagram represents the Mark Entry Application's database structure, focusing on staff, login, and courses. It includes entities like Login, Staff, and Course List, with attributes such as username, email, password, and marks. The diagram shows relationships between users, their courses, and stored marks for students.

# **CHAPTER 5**

## **SYSTEM DEVELOPMENT**

### **5.1 Functional Documentation**

#### **5.1.1 Staff Login Page**

Purpose: The Staff Login Page allows authorized staff members to securely access their homepage.

Functionality:

- Authentication: Staff members must enter their username and password to log in successfully.
- Error Handling: If the login credentials are incorrect, an error message will be displayed asking them to enter the correct details.
- Forgot Password: If a staff member forgets their password, they can reset it using a password recovery option. A link will be provided to set a new password.

#### **5.1.2 Mark Entry Page**

Purpose: The Mark Entry Page allows staff to enter marks for a course that is available on their homepage if they have created one.

Functionality:

- Dropdown Selection: Staff can select the exercise number for which they want to enter marks.
- Selective D.no: Staff can pick a specific D.no for which they need to enter marks.
- Validation: The system will not accept text values in the marks field. If a staff member enters a non-numeric value, an error message will be shown.
- Confirmation: After successfully saving the marks, a confirmation message (popup) will be displayed at the bottom of the screen.

#### **5.1.3 Course Creation Page**

Purpose: This page allows staff members to create a new course by entering the course name, starting D.no, and ending D.no.

Functionality:

- Create Course: When staff enter all the required details, a new course will be created under their name.
- Collaboration: If a staff member tries to create a course with a name that already exists in the database, the system will suggest an option to collaborate with that course instead of creating a duplicate.

#### **5.1.4 User Home page**

Purpose: The User Home Page serves as the main dashboard for staff members. It lists all the courses assigned to them and allows easy navigation for mark entry and course management.

Functionality:

- Course Card: Each course is displayed as a card, making it easy to access individual courses.
- Download: Staff members can download the stored marks in a structured PDF file for offline use or record-keeping.
- Delete: If a staff member no longer needs a course, they can delete it by long-pressing the course card and selecting the delete option. A confirmation message will appear before deletion to prevent accidental removal.

## **5.2 Special Features of Language/Utility**

In building the Department Repository Management System, the programming language's unique features and utility tools are used to make sure everything functions properly and effectively.

### **5.2.1 Framework – Flutter:**

Flutter is a UI framework developed by Google for building cross-platform applications with a single codebase.

- Cross-Platform Development – Write one codebase and deploy it on Android, iOS, web, and desktop.
- Hot Reload & Hot Restart – Instantly see UI changes without losing the app state.
- Widget-Based UI – Everything in Flutter is a widget, making UI design flexible and reusable.
- Performance Optimization – Uses Skia for smooth, hardware-accelerated graphics and



animations.

- Customizable & Beautiful UI – Provides built-in Material and Cupertino (iOS) widgets for native-like experiences.
- Open-Source & Backed by Google – Regular updates and strong community support.

### **5.2.2 Programming Language- Dart:**

Dart is the programming language behind Flutter, designed for front-end development with fast execution and scalability.

- Null Safety – Prevents null reference errors, improving code reliability.
- Asynchronous Programming – Uses `async/await` and `Future` for smooth handling of network calls and UI interactions.
- Just-In-Time (JIT) & Ahead-Of-Time (AOT) Compilation – JIT speeds up development, while AOT ensures optimized performance in production.
- Strongly Typed with Type Inference – Combines static typing with flexibility to infer types automatically.
- Garbage Collection – Efficient memory management for better app performance.

### **5.2.3 Database – Firebase:**

Firebase is a cloud-based backend-as-a-service (BaaS) that provides real-time data storage, authentication, and cloud functions.

- Firebase Authentication – Supports multiple sign-in methods, including email/password, Google, Facebook, and OTP.
- Firestore & Realtime Database – NoSQL databases for real-time syncing and offline capabilities.
- Cloud Functions – Allows serverless execution of backend logic triggered by events.
- Cloud Storage – Secure file storage for images, videos, and documents.
- Firebase Hosting – Provides fast and secure hosting for web applications.
- Firebase Messaging (FCM) – Enables push notifications across multiple platforms.
- Firebase Analytics – Tracks user behavior and app performance for insights.

## **5.3 Pseudo Code**

### **5.3.1 User Login**

Step 1: Display Login Page

Step 2: If User Enters Valid Credentials

Step 2.1: Move to User Home Page

Step 3: If user enters invalid credentials

Step 3.1: Display error message

### **5.3.2 Create Course**

Step 1: Display Course Setup Page

Step 2: Enter Course Name

Step 3: Enter Starting and Ending D.no

Step 4: If D.no is in correct format

Step 4.1: Successfully Created Course

Step 5: If D.no is not in correct format

Step 5.1: Display error message

### **5.3.3 Mark Entry Screen**

Step 1: Display Mark Entry Page

Step 2: Choose Exercise number and Data of the exercise

Step 3: Enter Mark for desired Department number

Step 4: Click next D.no or Previous Dno to save the Mark

Step 5: If D.no is not in correct format

#### **5.3.4 Generate PDF file**

Step 1: Display Home page

Step 2: Click Download button of the Course

Step 3: The PDF will start to generate with loading indicator

Step 4: Generated PDF can be saved in user's local storage

Step 5: If D.no is not in correct format

#### **5.3.5. Collaborate Function**

Step 1: Display Course Setup Page

Step 2: Enter Course name

Step 2.1: If Course already exist show collaborate option

Step 3: If course setup failed

Step 3.1: Show error message

## CHAPTER 6

### TESTING

#### 6.1 Types of Testing Done

##### 6.1.1 Unit testing

The unit testing table outlines various test cases for validating user inputs and authentication in the application. It includes function names, test descriptions, test data, and expected results to ensure proper error handling and validation. These test cases help maintain data integrity and improve the user experience by preventing invalid inputs.

Test Case Number	Function name	Test case description	Test Data	Expected Result
1	Validatefields()	Check if username is empty	Username: “ ”	Error message should be displayed: “Enter a Username”
2	loginuser ()	Check incorrect login details	Username: “Balaji” Password: “12345”	Error message should be displayed: “Invalid Username or Password”
3	ValidateDno()	Check if staff enter valid D. No	D. No: 20us111	Error message should be displayed: “Enter a valid D.no”
4	ValidateMark()	Check if staff enter valid mark	Mark: “dfs”	Error message should be displayed: “Enter a valid mark”
5	Validatefields()	Validate email format	Email: absc	Error message should be displayed: “Email is badly formatted”

### 6.1.2 Integration Testing

This integration testing table outlines various test scenarios to verify the interaction between different modules of the application, including Firebase authentication, database operations, and PDF generation. Each test case specifies a module, test scenario, test data, and expected outcome to ensure seamless data flow and correct functionality. The objective is to validate how well these modules work together in an integrated environment.

Test Case Number	Modules	Test Scenario	Test Data	Expected Result
1	Login to Firebase Auth	Staff enters valid data	Email: <a href="mailto:staff@mail.com">staff@mail.com</a> , Password: Correctpass	Login successful redirect to homepage
2	Signup to Firebase Auth	Staff enter valid credentials	Email: <a href="mailto:staff@mail.com">staff@mail.com</a> , Name: Staff, Password: Correctpass	Signup successful redirect to homepage
3	Mark entry to Database	Staff enter valid mark and click save	D. No: 22ucs111, Preparation mark: 4, Viva voce: 5	Data stored in firebase, snackbar “Mark saved for D. No 111”
4	Course creation to Database	Staff create new course	Course Name: Java, Starting D. No: 22ucs101, Ending D. No: 22ucs169	Course stored in firebase, snackbar “Course created successfully”
5	Stored marks to PDF download	Staff download marks	Course: java	Marks converted into PDF file, snackbar “PDF for java generated successfully”

### 6.1.3 System Testing

This system testing table ensures that the application functions correctly as a whole by testing key features such as staff login, mark entry, course creation, course deletion, and password reset. Each test case outlines a feature, a specific test scenario, input data, and the expected outcome to validate the system's behavior. The goal is to confirm that all components work seamlessly together in a real-world environment.

Test Case Number	Feature	Test Scenario	Test Data	Expected Result
1	Staff login	Staff Login with valid credentials	Email: <a href="mailto:staff@mail.com">staff@mail.com</a> , Password: Correctpass	Redirect to homepage
2	Mark entry	Staff select a course and enters valid mark	D. No: 22ucs111, Preparation Mark: 5, Viva voce: 3	Mark saved in firebase, Snackbar: "Mark saved for D.No 111"
3	Course Creation	Staff tries to create an existing course	Course: Java	Option to collaborate is shown
4	Delete Course	Staff deletes an existing course	Course Name: Java	Course removed from firebase
5	Forgot password	Staff requires a password reset	Email: staff@example.com	Password reset mail sent

## 6.2 Test Data & Output

**Form Name:** Staff Login

**Test case Description:** To test staff fill all the field

**Test Data:** Username: “ ”

Password: “ ”

**Expected Result:** Error message should be displayed: “Enter all the fields”



The screenshot shows a mobile application interface for staff login. At the top, the text "Log In" is centered. Below it is the circular logo of St. Joseph's College Tiruchirappalli, which includes the college's name and the motto "Remember - Rejoice - Reach out". The login form consists of two input fields: "Enter Your Mail" with an envelope icon and "Enter password" with an eye icon. Below the password field is a link labeled "Forgot Password?". A purple "Log in" button is positioned below the links. At the bottom of the form, there is a link "No account? Signup". A dark grey bar at the very bottom of the screen contains the error message "Enter all the fields" in white text.

Fig: 6.2.1 Staff Login

**Form Name:** Course setup

**Test case Description:** To test staff fill Dno in the proper format

**Test Data:** Coursename: “c++”, D.no: “21cs101”

**Expected Result:** Error message should be displayed: “D.no must be exactly 8 characters”

11:09

Setup Course

Course Name c++

Starting D.no 21cs101

Ending D.no 21cs152

Create Course

D.no must be exactly 8 characters

Fig: 6.2.2 Course Setup



# **CHAPTER 7**

## **USER MANUAL**

### **7.1 Hardware Requirements**

Processor: Minimum Quad-Core 1.3 GHz (Recommended: Octa-Core 1.8 GHz)

RAM: 1GB (Minimum), 2GB+ (Recommended)

Storage: 100MB available disk space

Network: Wi-Fi or Mobile Data connectivity

### **7.2 Software Requirements**

Operating System: Android 7.0 (Nougat) or later

Development Language: Java 17

Database: Firebase Realtime Database / Firestore

IDE: Android Studio (Latest Version)

Build Tool: Gradle

Additional Dependencies: Firebase SDK, Jetpack Libraries

### **7.3 Installation Procedure**

- **System Requirements Check:**
  - Operating System: Android 7.0 (Nougat) or later
  - Internet Connection: Stable Wi-Fi required for real-time mark entry
  - Storage Space: At least 100MB free for app installation
- **Accessing the Department Repository Management System:**
  - Open the installed app from the home screen or app drawer.
  - Log in using your provided credentials.
  - Enter student marks and save them to Firebase.
  - Navigate through different courses and exercises using the UI controls.

## 7.4 Sample I/O

### 7.4.1 Login Input

Email: [user@example.com](mailto:user@example.com)

Password: Password123

Expected output:

If credentials are correct move to homescreen

If credentials are incorrect show “Invalid email or password”

### 7.4.2 Course Creation Input

Course name: Java

Starting D. no: 22ucs101

Ending D. no: 22ucs169

Expected output:

If there exist a course with same name then allow user to either collaborate with the course or rename the new course

If there is no existing course then show “Course setup successful”

### 7.4.3 Signup Input

Email: [user@example.com](mailto:user@example.com)

Name: Ajay

Password: Password123

Expected output:

If the email is not already registered move to homescreen

If email is already registered show “Already signed up”

## **7.5 Error Messages**

- 1) Error Message: "Invalid username or password."

Form Name: Staff Login

Description: This message appears when the user enters incorrect login credentials.

- 2) Error Message: "Fill all the fields"

Form Name: Staff Login

Description: This message appears when staff click login without fill the fields

- 3) Error Message: "D no Must be 8 characters"

Form Name: Staff Profile

Description: This message appears when the staff click create button without providing proper format

## **CHAPTER 8**

### **CONCLUSION**

#### **8.1 Summary of the Project**

The Lab Mark Entry Application is a mobile app designed to replace the traditional pen-and-paper method of recording student marks with a faster and more efficient digital system. This app makes it easy for staff members to enter and manage student marks in a user-friendly and organized way.

The application is developed to work smoothly on both low-end and high-end devices, ensuring that all staff members can use it without any performance issues. The simple and intuitive design allows users to navigate the app easily, making the mark entry process quick and hassle-free.

Key Features:

- **Secure Login:** Staff members can log in using their email and password, ensuring secure access to the system.
- **Course Management:** Staff can create a course by entering the course name, starting D.no, and ending D.no.
- **Mark Entry:** After setting up a course, staff can enter marks for students directly into the system.
- **PDF Export:** Once marks are entered, staff can download the marks as a PDF file and save them to their local storage for future reference.

#### **8.2 Future Enhancements**

In future updates, the Lab Mark Entry Application can be improved with additional features to enhance functionality and user experience. One major enhancement is integrating the application with the college server, allowing staff members to be automatically assigned to their respective courses without manual intervention. This will streamline the workflow and reduce administrative effort.

Another planned feature is a staff list option, which will display all the staff members who have collaborated on a particular course. This will improve transparency and make it easier for staff to track contributions within shared courses.

To ensure data security and prevent loss of important records, a backup server will be implemented. This will allow the system to store all entered marks securely and retrieve them in case of accidental deletion or technical issues. With these enhancements, the application will become more efficient, reliable, and user-friendly for staff members.

# **BIBLIOGRAPHY**

## Book References

- Gilad Bracha, “Dart Programming Language”, Addison-Wesley Professional, 7 December 2015
- John Horton, “Android Programming for Beginners”, Packt Publishing, 31 October 2018
- Diego Rodrigues, “LEARN FIREBASE: Integrate Real-Time Backend for Web and Mobile Applications ”, StudioD21 Smart Tech Content, 14 February 2024

## Web References

- W3school: Dart Tutorial (<https://www.geeksforgeeks.org/dart-tutorial/>)
- Geeksforgeeks: Firebase Tutorial (<https://www.geeksforgeeks.org/firebase-tutorial/>)
- Tutorialpoint: Flutter Tutorial (<https://www.tutorialspoint.com/flutter/index.htm>)

## APPENDICES

### Sample Code

Login Page:

```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:markit/authentication.dart';
import 'package:markit/screens/signup_page.dart';
import 'package:markit/screens/userhome_page.dart';
import 'package:shared_preferences/shared_preferences.dart';

class loginPage extends StatefulWidget {
  const loginPage({super.key});

  @override
  State<loginPage> createState() => _loginPageState();
}

class _loginPageState extends State<loginPage> {
  bool isVisible = false;
  TextEditingController emailtxt = TextEditingController();
  TextEditingController passtxt = TextEditingController();

  Future<void> saveUserData(String name) async {
    final prefs = await SharedPreferences.getInstance();
    await prefs.setString('userName', name);
  }

  void loginUser() async {
    // Show the loading dialog
    showDialog(
      context: context,
      barrierDismissible: false, // Prevents the user from closing the dialog
      builder: (BuildContext context) {
        return const Center(
          child: CircularProgressIndicator(
            color: Colors.deepPurple,
          ),
        );
      },
    );

    // Login the user
    String res = await Authentication()
```

```

        .loginUser(email: emailtxt.text, password: passtxt.text);

if (res == "Success") {
    try {
        // Get the current user's UID
        String uid = FirebaseAuth.instance.currentUser!.uid;

        // Fetch user details from Firestore using UID
        DocumentSnapshot userDoc = await FirebaseFirestore.instance
            .collection("staffs")
            .doc(uid)
            .get();

        if (userDoc.exists) {
            // Extract the user's name
            Map<String, dynamic> userData =
                userDoc.data() as Map<String, dynamic>;
            String userName = userData["name"];

            // Close the loading dialog
            Navigator.of(context).pop();

            // Show success message
            ScaffoldMessenger.of(context).showSnackBar(
                const SnackBar(content: Text('Successfully logged In')),
            );
            await saveUserData(userName);
            // Navigate to the UserhomePage with the name
            Navigator.push(
                context,
                MaterialPageRoute(
                    builder: (context) => UserhomePage(
                        name: userName, // Pass the user's name to the next screen
                    ),
                ),
            );
        } else {
            throw Exception("User document not found");
        }
    } catch (e) {
        Navigator.of(context).pop(); // Close the loading dialog
        ScaffoldMessenger.of(context).showSnackBar(
            SnackBar(
                content: Text("Failed to fetch user details: ${e.toString()}"),
            ),
        );
    }
} else {
    Navigator.of(context).pop(); // Close the loading dialog

```



```

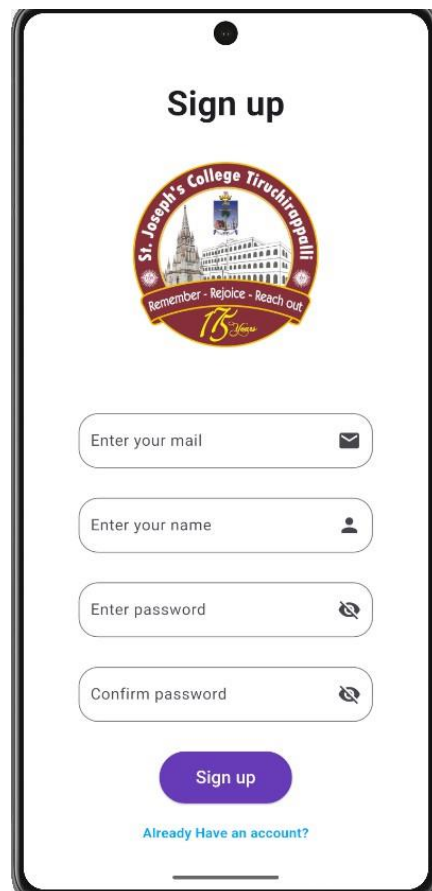
ScaffoldMessenger.of(context).showSnackBar(
  SnackBar(content: Text(res)),
);
}
}

```

## Screenshot



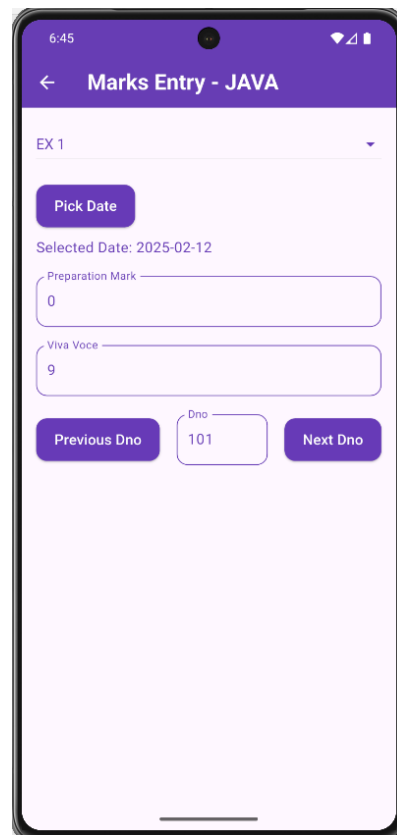
Login Screen



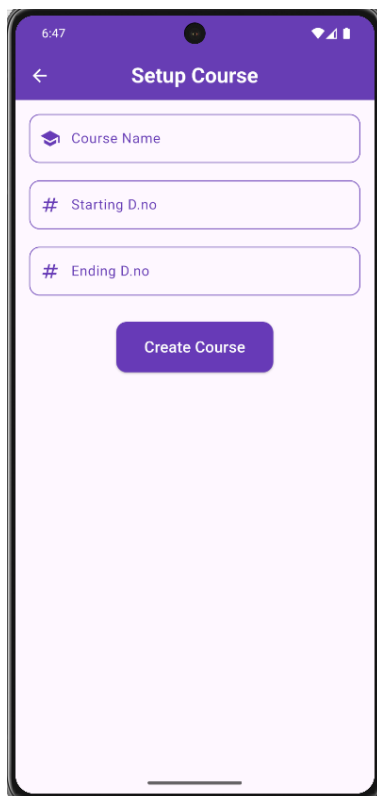
Signup Screen



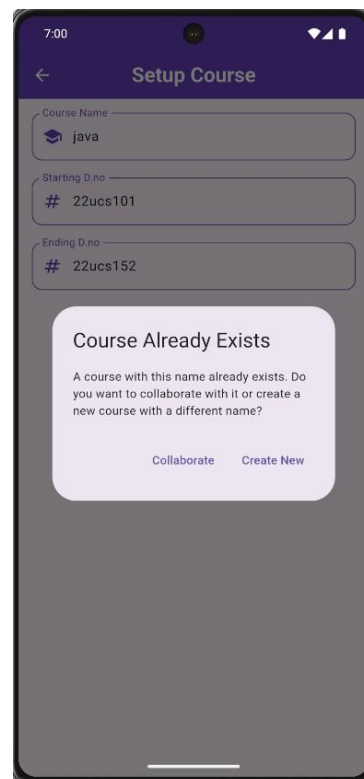
Home Screen



Mark Entry Screen



Course Setup Screen



Collaborate Option

## Marks Report - java

[illegible]

Generated PDF