1. INTRODUCTION

The ProTrack system is a Project Tracking and Management System designed specifically for the Computer Science Department. It streamlines the process of managing student projects by providing a structured platform for administrators, teachers, and students to collaborate efficiently. The system ensures proper tracking of project progress, submission deadlines, and review schedules, making project evaluation more organized and transparent.

The project consists of three modules: Admin (Head of the Department), Teacher, and Student. The Admin module, developed as a web application, enables department heads to manage students, assign teachers, and oversee project progress. The Teacher and Student modules, developed as mobile applications, allow teachers to review projects and provide feedback, while students can submit projects and track their review status. With features such as project type management, technology tracking, and scheduled reviews, ProTrack enhances efficiency in academic project management, ensuring a smooth and structured workflow for all users.

2. SYSTEM ANALYSIS

System analysis is the process of gathering and interpreting facts, diagnosing problems and using the facts to improve the system. Analysis is a detailed study of various operations performed by a system and their relationship within and outside of the system. This involves gathering information and using structured tools for analysis. System analysis is a step-by-step process used to identify and develop or acquire the software need to control the processing of specific application. System analysis is a continuing activity the stages of the systems development. System analysis is the process of gathering and interpreting facts, diagnosing problems and using the facts to improve the system. The outputs from the organization are traced through the various processing that the input phases through in the organization. This involves gathering information and using structured tools for analysis. A detailed study of this process must be made by various techniques like interviews; questionnaires etc. It is necessary to have such a good system analysis and then by a project development cycle so that the project can be completed in a strictly manner and able to finish with the desired time. The analyst must be so careful about his responsibilities. As the next step the current system analysis is done which identifies the real need of establishing our project in the environment, its opportunities and constraints etc. All of the steps discussed above are collectively known as the system analysis.

2.1 EXISTING SYSTEM

The current system for managing academic projects in educational institutions relies heavily on manual processes, spreadsheets, and basic digital tools such as emails or local file storage. Students submit their project proposals, reports, and review files through email or physical copies, which faculty members then review and provide feedback on. This method lacks centralized tracking, making it difficult for administrators and teachers to monitor project progress effectively. Communication between students and teachers often takes place through disconnected channels, leading to delays, miscommunication, and missed deadlines. Additionally, project records are often scattered across different platforms, making it challenging to maintain a structured workflow for submission, evaluation, and feedback.

Drawbacks of the Existing System:

1. Lack of Centralized Management: Data is scattered across multiple platforms, making it difficult to track project progress efficiently.

2. Time-Consuming Process: Manually handling project approvals, reviews, and feedback increases administrative workload.

3. Limited Collaboration: Communication gaps between students, teachers, and administrators lead to delays in project completion.

4. Difficulty in Tracking Changes: There is no structured version control for project updates, making it hard to track modifications and feedback.

5. Security Concerns: Sensitive project data may not be securely stored, increasing the risk of data loss or unauthorized access.

Due to these limitations, a more automated, centralized, and user-friendly system like ProTrack is needed to streamline project tracking, enhance communication, and ensure efficient management of academic projects.

2.2 PROPOSED SYSTEM

The ProTrack system is designed to provide an efficient and structured platform for managing academic projects in the Computer Science department. This system will automate project tracking, submission, review, and evaluation by integrating a centralized web and mobile application. It aims to enhance coordination between students, teachers, and administrators, ensuring seamless communication and efficient project management. The system will consist of three modules: the Admin (Head of Department) module as a web application, and the Teacher and Student modules as mobile applications.

The proposed system enables students to submit project files digitally, while teachers can review, provide feedback, and track progress in real-time. Automated notifications ensure deadline updates, and secure storage maintains data integrity. Analytics help administrators monitor progress and workload. ProTrack enhances efficiency, reduces errors, and improves academic collaboration.

ADVANTAGES

The ProTrack system introduces several key advantages over the existing manual or semi-digital project management methods.

1. Efficient Project Management – The system provides a structured and centralized platform for tracking project progress, making it easier for students, teachers, and administrators to stay updated on the status of each project.

2. Paperless Workflow – By digitizing project submissions, reviews, and approvals, the system eliminates the need for physical documentation, reducing paperwork and ensuring an eco-friendly approach.

3. Improved Communication – The system facilitates seamless communication between students and faculty members through real-time notifications, comments, and feedback mechanisms, reducing delays caused by miscommunication.

4. Automated Notifications and Reminders – Students and faculty receive automatic reminders for submission deadlines, review schedules, and progress updates, minimizing the chances of missing important tasks.

5. Secure and Organized Data Storage – All project-related documents and records are stored securely in a centralized database, making retrieval easy while preventing data loss or unauthorized access.

6. Enhanced Evaluation Process – Teachers can efficiently review submissions, provide feedback, and track students' progress, ensuring a transparent and systematic evaluation process.

7. Role-Based Access Control – The system ensures that only authorized users (admin, teachers, students) can access specific functionalities, enhancing security and maintaining confidentiality.

8. Time-Saving and User-Friendly Interface – The intuitive UI allows users to navigate effortlessly, making project management more efficient while saving time for both students and faculty.

2.3 SYSTEM REQUIREMENT SPECIFICATION

A software requirements specification (SRS) is a comprehensive description of the intended purpose and environment for software under development. The SRS fully describes what the software will do and how it will be expected to perform. An SRS minimizes the time and effort required by developers to achieve desired goals and also minimizes the development cost. A good SRS defines how an application will interact with system hardware, other programs and human users in a wide variety of real worked situations.

Problem to be Solved

This system solves the manual errors happening in a daycare.

Customer requirements

- The system should be fast.
- User friendly.
- Maintaining the security of customer's data.
- Efficiency in data retrieval and appointment management

What does the developer need to know?

Must know the existing system and its drawbacks.
Must know what will be needed in the proposed system.

Business Requirements

The system should be feasible both for the developer and the client. It should be effective and should be able to be completed in time. The developer should be responsible for developing the system, installing the software, updating the software whenever necessary, and conducting any user training that may be needed for using the system.

User Requirements

The user requirement(s) specification is a document usually that specifies the requirements the user expects from software to be constructed in a software project.

The administrator has overall control of the system.

- Provide customer details.
- Faster processing

Functional Requirements

Functional requirements define what a system is supposed to do. The system should perform the following functionalities.

- Login – Login of admin, teacher, and student.
- Admission– Teacher can register new student.
- Edit profile– Students can update their profile.
- View status – Students can view the status of their activities in the app.
- Approve – The teacher can approve the abstract.
- Logout – System users can log out from the app.

2.3.1 HARDWARE SPECIFICATIONS

Processor	: Intel core i5 or higher processor
Speed	: 2.50GHz or Higher
System bus	: 64 bits
Memory	: 16 GB RAM or Higher
Hard disk	: 40 GB or Higher
Monitor	: 15.6” FHD
Keyboard	: 104 Keys

2.3.2 SOFTWARE SPECIFICATIONS

Operating System: Windows 11
Front End: Flutter (Dart)
Back End: Supabase
Development Tools: Android Studio, Visual Studio Code
Platform: Android, Web
Browser Compatibility: Google Chrome,

2.3.3 FRONT END

FLUTTER

Flutter is an open-source UI software development kit created by Google. It is used to develop cross-platform applications from a single codebase, allowing developers to build high-performance, visually appealing apps for mobile, web, and desktop.

- Flutter uses the Dart programming language, which is optimized for fast app development.
- It provides a rich set of pre-designed widgets that help create responsive and attractive user interfaces.
- Flutter's hot reload feature allows developers to see real-time updates without restarting the application.
- It ensures a smooth user experience with a high frame rate and native-like performance.

Common Uses of Flutter

- Flutter is used for building mobile applications that work on both Android and iOS.
- It enables web and desktop development using the same codebase.
- It is ideal for applications requiring rich UI elements and animations.
- It integrates seamlessly with Supabase for backend services such as authentication, cloud storage, and real-time databases.

Characteristics of Flutter

Flutter's advantages include:

Cross-Platform Development – Develop once, deploy anywhere.
Fast Development – Hot reload accelerates the development process.
Beautiful UI – Uses Material Design and Cupertino widgets.

High Performance – Delivers near-native performance.
Strong Community Support – Backed by Google and a vast developer community

2.3.4 BACK END

SUPABASE

Supabase is an open-source backend-as-a-service (BaaS) platform that provides a scalable and powerful alternative to Firebase. It is built on PostgreSQL and offers real-time capabilities, authentication, and cloud storage.

- Supabase Authentication allows secure user login using email, Google, and other OAuth providers.
- Supabase Database is a managed PostgreSQL database with real-time capabilities.
- Supabase Storage enables file and media storage with easy access control.
- Supabase Functions provide serverless computing for custom backend logic.
- Supabase API allows seamless integration with Flutter applications

2.4 FEASIBILITY ANALYSIS

A feasibility study is an evaluation and analysis of the potential of the proposed project which is based on extensive investigation and research to give full comfort to the decision makers. Feasibility studies aim to objectively and rationally uncover the strengths and weaknesses of existing business of proposed venture, opportunities and threats as presented by the environment, the resources required to carry through, and ultimately the process for success. In its simplest terms, the two criteria to judge feasibility are cost required and value to attain. As such, a well-designed feasibility study should provide a historical background of the business or project, a description of the product or service, accounting statements, details of the operations and management, marketing research and policies, financial data, legal requirements, and tax obligations.

The four aspects of the feasibility study are:

- Technical feasibility
- Economic feasibility
- Operational feasibility
- Behavioural feasibility

Technical Feasibility

The technical feasibility centers on the existing system and to what extent it can support the proposed addition. The technical feasibility assessment is focused on gaining an understanding of the present technical resources of the organization and their applicability to the expected needs of the proposed system. The minimum requirements of the system are met by the average user. The developer system has á modest technical requirements as only minimal or null changes are required for implementing the system.

Normally associated with the technical feasibility includes:

- Development risk
- Resource availability
- Technology

The proposed system can work without any additional hardware or software support other than the computer system and networks. So, I analyzed that the proposed system is much more technically feasible than other systems when compared with the benefits of the new system.

Economic Feasibility

Economic feasibility analysis is also known as cost/benefit analysis. The purpose of the economic feasibility assessment is to determine the positive economic benefits to the organization that the proposed system will provide. The proposed system reduces the operating cost in terms of time by automating the process. This system is economically feasible.

Operational Feasibility

Operational feasibility is a measure of how well a proposed system solves problems and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.

Behavioral Feasibility

People are inherently resistant to changes and computer is known for facilitating the changes.

An estimate should be made to how strongly the users react toward the e development of the system. The proposed system consumes less time. Thus, the people are made to engage in some other important work.

2.5 DATA FLOW DIAGRAM (DFD)

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. It differs from the flowchart as it shows the data flow instead of the control flow of the program. A data flow diagram can also be used for the visualization of data processing (structured design).

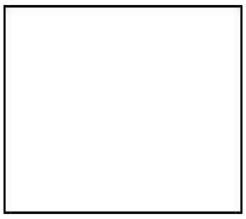
Data flow diagrams were invented by Larry Constantine, the original developer of structured design based on Martin and Estrin's "data flow graph" model of computation.

Data flow diagrams (DFDs) are one of the three essential perspectives of Structured System Analysis and Design Method SSADM. The sponsor of a project and the end users will need to be briefed and consulted throughout all stages of a system's evolution. With a data flow diagram, users can visualize how the system will operate, and what the system will accomplish. and how the system will be implemented. The old system's data flow diagrams can be drawn up and compared with the new system's data flow diagrams to draw comparisons to implement a more efficient system. Data flow diagrams can be used to provide the end user with a physical idea of where the data they input ultimately has an effect on the structure of the whole system from order to dispatch to report. How any system is developed can be determined through a data flow diagram.

Developing a data flow diagram helps in identifying the transaction data in the data model. There are different notations to draw data flow diagrams, defining different visual representations for processes, data stores, data flow, and external entities. The first step is to draw a data flow diagram (DFD). A DFD also known as a "bubble chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So, it is the starting point of the design phase that functionally decomposes the requirements specification down to the lowest level of details DFD consists of a series of bubbles joined by lines. The bubbles represent data transformation and the Iines represent data flow in the system.

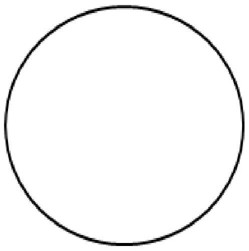
DFD Symbols: -

- Square- Defines the source or destination of the system.

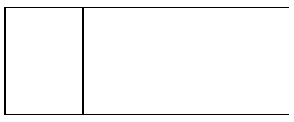


- Data flow - Identifies data flow Circle.

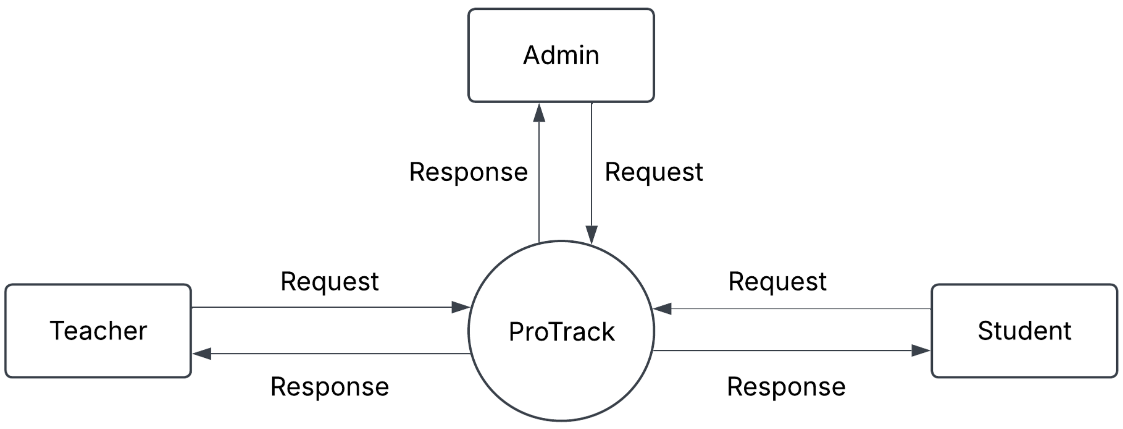
- Bubble - Represents a process that transforms incoming data to outgoing data.



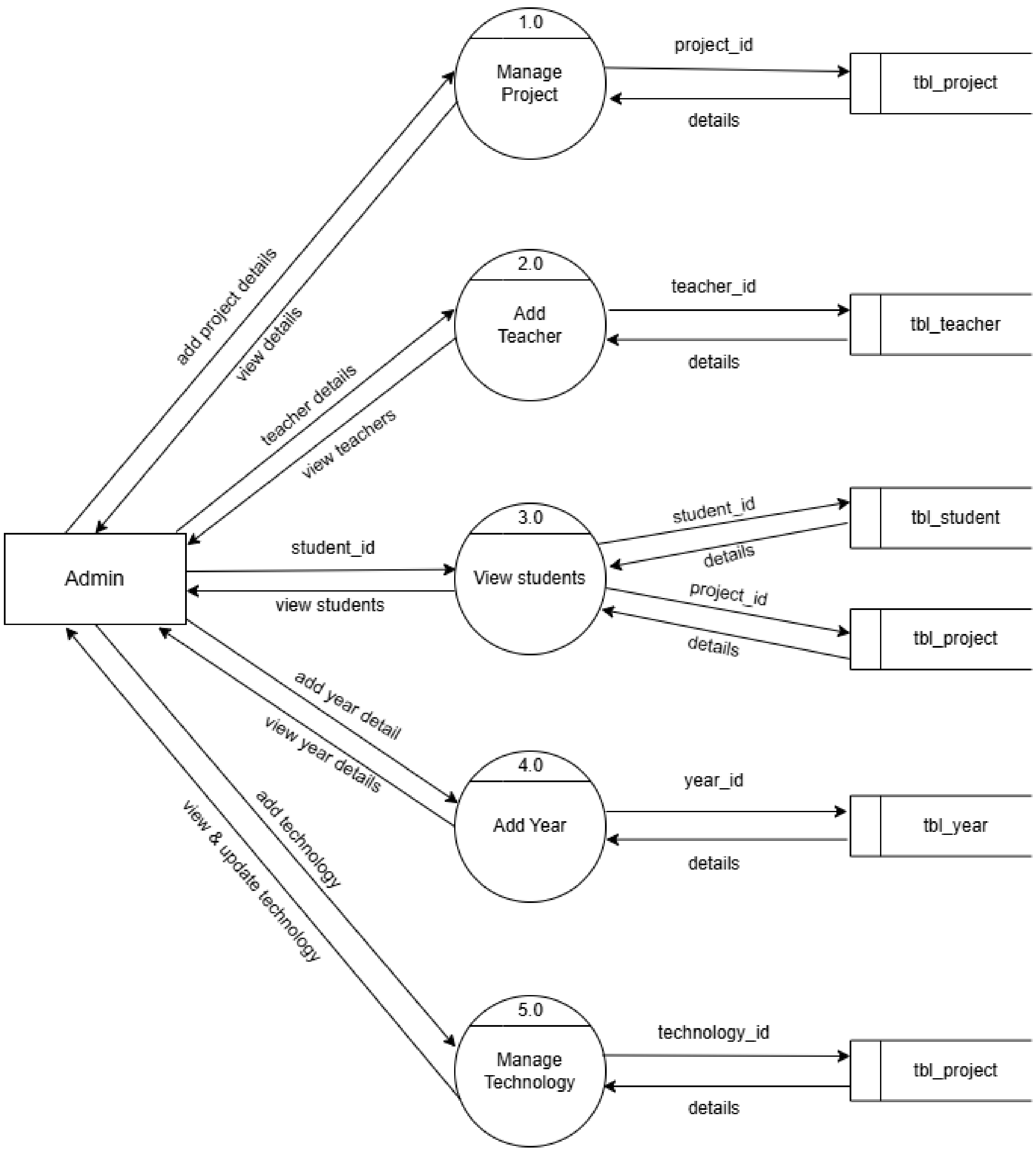
- Open rectangle- Data store

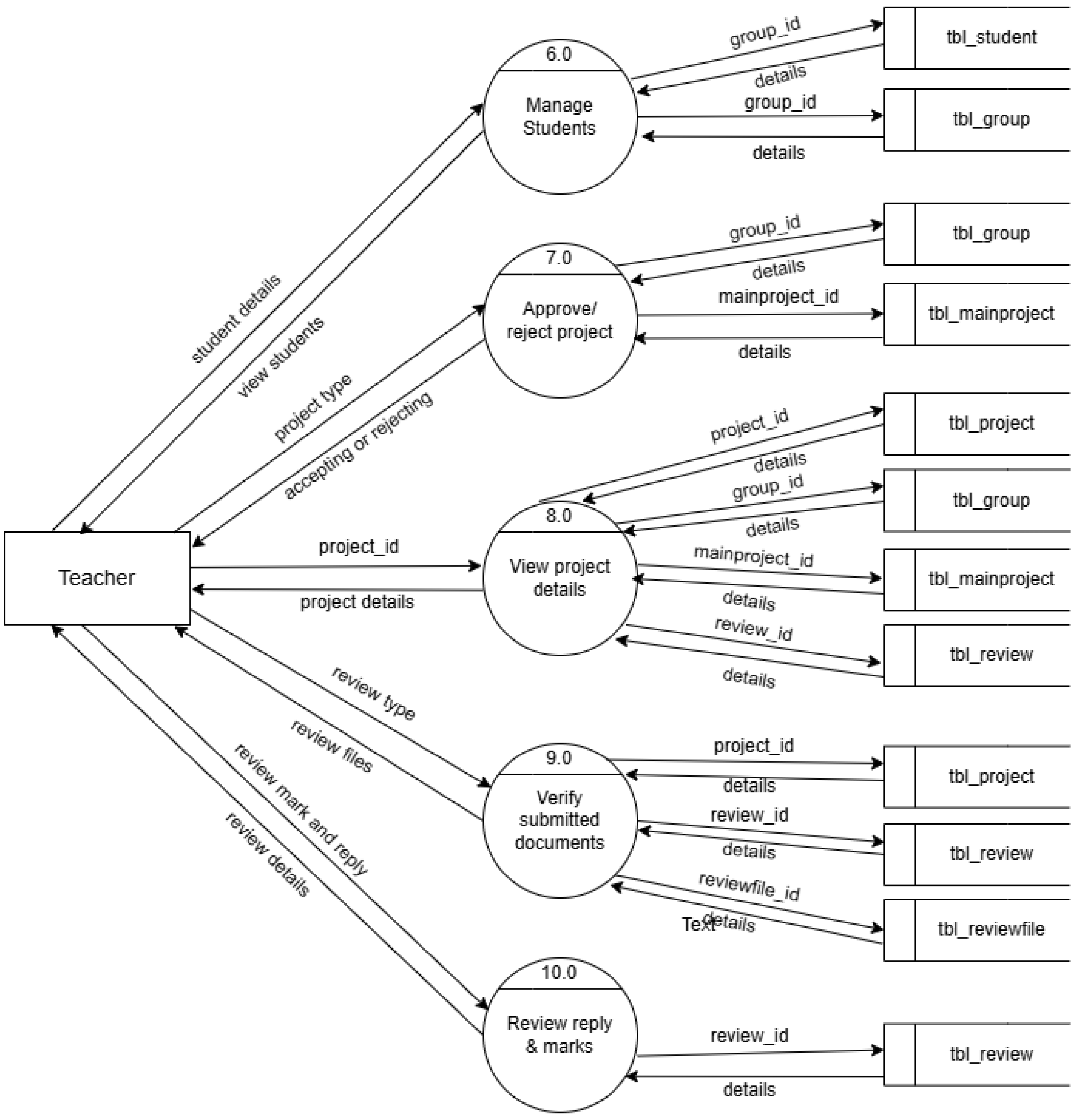


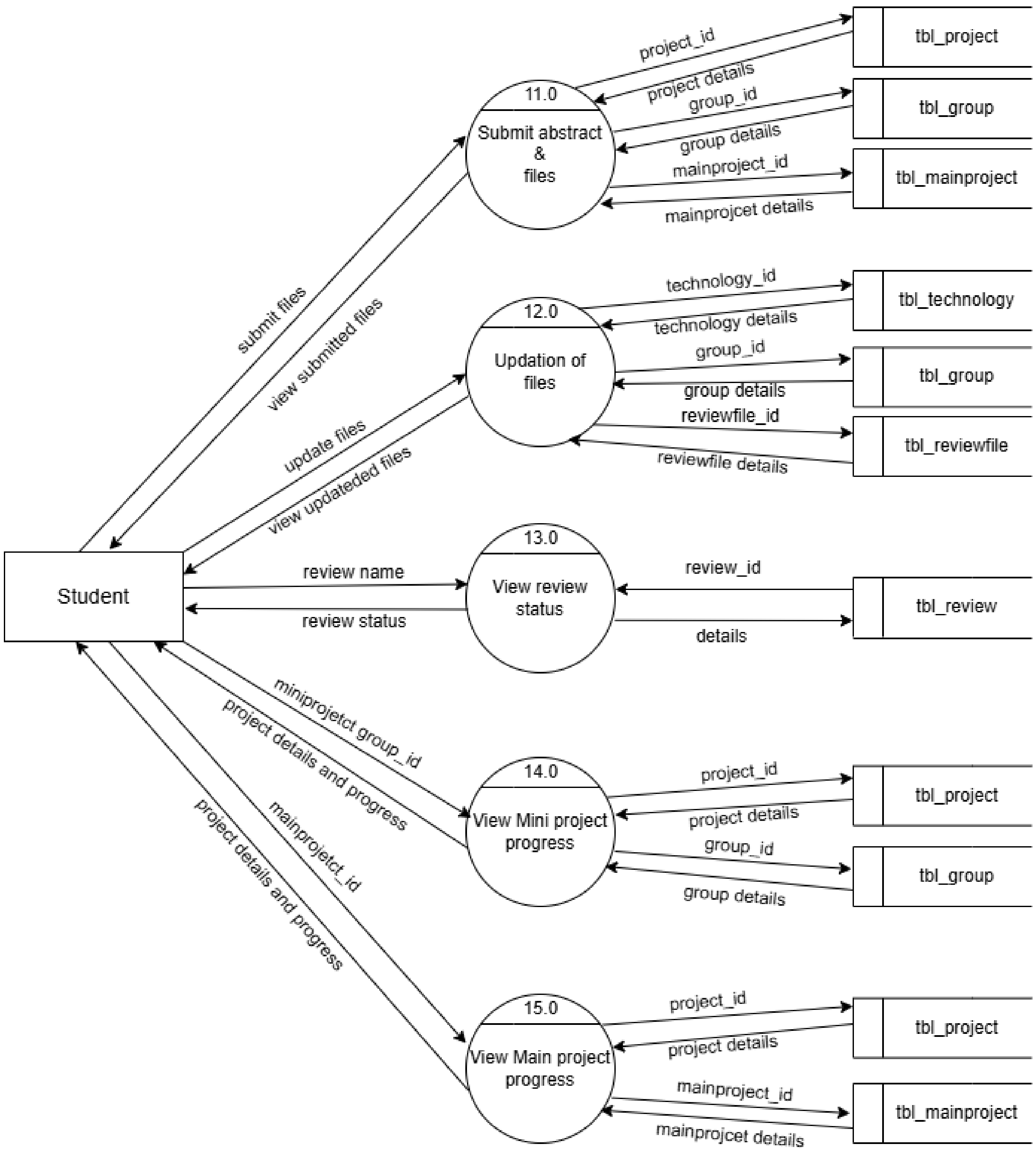
LEVEL 0



LEVEL 1


1.0
Manage Project
project_id
details
tbl_project
add project details
view details
2.0
Add Teacher
teacher_id
details
tbl_teacher
teacher details
view teachers
3.0
View students
student_id
details
tbl_student
project_id
details
tbl_project
Admin
student_id
view students
add year detail
view year details
4.0
Add Year
year_id
details
tbl_year
add technology
view & update technology
5.0
Manage Technology
technology_id
details
tbl_project


Teacher
6.0
Manage Students
7.0
Approve/ reject project
8.0
View project details
9.0
Verify submitted documents
10.0
Review reply & marks
student details
view students
project type
accepting or rejecting
project_id
project details
review type
review files
review mark and reply
review details
group_id
details
mainproject_id
project_id
review_id
reviewfile_id
Text
tbl_student
tbl_group
tbl_mainproject
tbl_project
tbl_review
tbl_reviewfile


Student
11.0
Submit abstract & files
12.0
Updation of files
13.0
View review status
14.0
View Mini project progress
15.0
View Main project progress
tbl_project
tbl_group
tbl_mainproject
tbl_technology
tbl_reviewfile
tbl_review
submit files
view submitted files
update files
view updateded files
review name
review status
miniprojetct group_id
project details and progress
mainprojetct_id
project details and progress
project_id
project details
group_id
group details
mainproject_id
mainprojcet details
technology_id
technology details
reviewfile_id
reviewfile details
review_id
details

3. SYSTEM DESIGN

3.1 INPUT DESIGN

The quality of the system input determines the quality of the system output. Input specification describes the way data enter the system for processing. Input design features can ensure the reliability of the system and produce results from accurate data, or they can result in the production or erroneous information. The input design also determines whether the user can interact efficiently with the system.

In our system almost all inputs are being taken from the databases. To provide adequate inputs we have to select necessary values from the databases and arrange it to the appropriate controls.

ADMIN

The Admin oversees project management, assigning teachers to student groups, setting review schedules, and monitoring progress. They ensure smooth coordination, manage academic data, and generate reports for efficient project tracking.

TEACHER

The Teacher supervises student projects, reviews submissions, provides feedback, and tracks progress. They guide students throughout the project and ensure timely completion.

STUDENT

The **Student** submits project proposals, reports, and reviews, receives feedback from teachers, and tracks project progress through the system.

3.2 OUTPUT DESIGN

In ProTrack, output plays a vital role in delivering meaningful information to users—Admins, Teachers, and Students. The success of the system heavily depends on the clarity and relevance of its output, as users often evaluate a system based on what it delivers. Poor or confusing output can lead to user dissatisfaction, making even a well-developed system feel ineffective.

To ensure value, ProTrack generates structured and user-friendly outputs such as review results, feedback, submission statuses, and progress reports. Students receive timely notifications and remarks, while teachers can track project details, view submissions, and provide feedback. Admins access analytics and summaries for efficient monitoring.

Each output is designed through continuous interaction with users, ensuring the information is relevant, clear, and timely. This enhances transparency, improves academic coordination, and encourages effective usage of the system.

3.3 DESIGN

The data design transforms the information domain model created during analysis into the data structures that will be required to implement the software. The data objects and relationships defined in the entity relationship diagram and the detailed data content depicted in the data dictionary provide the basis for the data design activity.

The overall objective in the development of database technology has been to treat data as an organizational resource and as an integrated whole. Database Management System allows data to be protected and organized separately from other resources. A database is an integrated collection of data. This is the difference between logical and physical data.

The organization of data in the database aims to achieve three major objectives:

- Data integration
- Data integrity
- Data independence

The databases are implemented using a DBMS package. Each DBMS has unique characteristics and general techniques for database design. There are 6 major steps in the design process. The first 5 steps are usually done on paper and finally, the design is implemented.

- Identify the table and relationships.
- Identify the data that is needed for each table and relationship.
- Resolve the relationship.
- Verify the design.
- Implement the design.

The database uses tables for storage. A table also contains records, which is a set of fields.
All records, in a table have the same set of fields with different information. Uses 11 tables.
Each table contains key fields that establish relationships in the database and how the records are stored. There are primary key fields that uniquely identify a record in a table. There are also fields that contain the primary key from another table called foreign keys.

Table Design

1. Table name: tbl_admin

Primary key: admin_id

Description: Stores the details of admin.

FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
admin_id	UUID	--	Unique id of admin
admin_name	TEXT	--	Name of admin
admin_email	TEXT	--	Email id of admin

2.Table name: tbl_teacher

Primary key: teacher_id

Description: Stores the details of teacher.

FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
teacher_id	UUID	--	Unique id of teacher
teacher_name	TEXT	--	Name of teacher
teacher_email	TEXT	--	Email id of teacher
teacher_contact	TEXT	--	Contact of teacher
teacher_photo	TEXT	--	Photo

3. Table name: tbl_student

Primary key: student_id

Foreign key: teacher_id, year_id

Description: Stores the details of student.

FIELD NAME	**DATA TYPE**	**SIZE**	**DESCRIPTION**
student_id	UUID	--	Unique identifier for each attendance record.
student_name	TEXT	--	Name of student
student_photo	TEXT	--	Photo
student_contact	TEXT	--	Contact number of student
student_email	TEXT	--	Email of student
teacher_id	INT	--	The unique id of students project guide teacher
year_id	INT	--	Students year id

4. Table name: tbl_group

Primary key: group_id

Foreign key: project_id

Description: Stores the details of group.

FIELD NAME	**DATA TYPE**	**SIZE**	**DESCRIPTION**
group_id	INT	--	Unique id of group
group_abstract	TEXT	--	Group abstract of project
group_status	INT	--	Status of group
group_center	TEXT	--	Center for group project
project_title	TEXT	--	Title of project
project_id	INT	--	Unique id of project

5. Table name: tbl_groupmember

Primary key: groupmember_id

Foreign key: group_id, student_id

Description: Stores the details of groupmember.

FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
groupmember_id	INT	--	Unique id of groupmember
group_id	INT	--	Unique id of group
student_id	INT	--	Unique id of student

6. Table name: tbl_mainproject

Primary key: event_id

Foreign key: student_id, technology_id, project_id

Description: Stores the details of the main project.

FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
mainproject_id	INT	--	Unique id of mainproject
mainproject_abstarct	TEXT	--	Abstract of mainproject
mainproject_status	INT	--	Status of mainproject
mainproject_title	TEXT	--	Title of mainproject
mainproject_certificate	TEXT	--	Certificate of mainproejct
mainproject_center	TEXT	--	Center of mainproject
project_id	INT	--	Unique id of project
technology_id	INT	--	Unique id of technology
student_id	INT	--	Unique id of student

7. Table name: tbl_project

Primary key: project _id

Description: Stores the details of the project.

FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
project _id	INT	--	Unique id of project
project _type	TEXT	--	Type of project
project _date	TEXT	--	Starting date of project
project _review1	TEXT	--	First review date
project _review2	INT	--	Second review date
project _review2	TEXT	--	Third review date
project _status	TEXT	--	Status of project

8. Table name: tbl_review

Primary key: review_id

Foreign key: group_id, mainproject_id

Description: Stores the details of the review.

FIELD NAME	DATA TYPE	SIZE	DESCRIPTION
review_id	INT	--	Unique id of review
review_date	TEXT	--	Date of review
review_type	TEXT	--	Type of review
review_mark	TEXT	--	Mark of review
review_reply	TEXT	--	Reply of review
group_id	INT	--	Unique id of group
mainproject_id	INT	--	Unique id of mainproject

9.Table Name: tbl_reviewfile

Primary Key: reviewfile_id

Foreign key: review_id

Description: Stores details of reviewfile.

FIELD NAME	**DATA TYPE**	**SIZE**	**DESCRIPTION**
reviewfile_id	INT	--	Unique id of reviewfile
reviewfile_file	TEXT	--	Files in review
file_type	TEXT	--	Type of file
review_id	INT	--	Unique id of review

10.Table Name: tbl_technology

Primary Key: technology_id

Description: stores details of technology.

FIELD NAME	**DATA TYPE**	**SIZE**	**DESCRIPTION**
technology_id	INT	--	Unique id of technology
technology_name	TEXT	--	Name of technology

11.Table Name: tbl_year

Primary Key: year_id

Description: store details of year

FIELD NAME	**DATA TYPE**	**SIZE**	**DESCRIPTION**
year_id	INT	--	Unique id of year
year_name	TEXT	--	Name of year

4. SYSTEM IMPLEMENTATION AND TESTING

4.1 SYSTEM TESTING

Testing is the process of examining the software to compare the actual behaviour with that of the excepted behaviour. The major goal of software testing is to demonstrate that faults are not present. In order to achieve this goal, the tester executes the program with the intent of finding errors. Though testing cannot show the absence of errors by not showing their presence it is considered that these are not present.

System testing is defined as the process by which one detects defects in the software. Any software development organization or team has to perform several processes. Software testing is one among them. It is the final opportunity of any programmer to detect and rectify any defects that may have appeared during the software development stage. Testing is the process of testing a program with the explicit intention of finding errors that make the program fail. In short system testing and quality assurance is a review of software products and related documentation for completion, correctness, reliability, and maintainability.

System testing is the first stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct the goal will be successfully achieved. A series of tests are performed for the proposed system before the proposed system is ready for user acceptance testing. The testing steps are,

- Unit testing
- Integration testing
- Acceptance Testing
- Validation
- Output testing

System Testing provides the file assurance that software once validated must combined with all other system elements. System testing verifies whether all elements have been combined properly and that overall system function and performance are achieved. FA the integration of modules, the validation test was carried out over the system. It was that all the modules worked well together and met the overall system function and performance.

1. Unit Testing

Unit testing is carried out screen-wise, with each screen being identified as an object. Attention is diverted to individual modules, independently to one another to locate errors. This has enabled the detection of errors in coding and logic.

Various test cases are prepared. For each module, these test cases are implemented, and it is checked whether the module is executed as per the requirements and outputs the desired result.

In this test each service input and output parameters are checked.

In unit testing

- Module interface was tested to ensure that information properly flows into and out of the program under test.
- Boundary condition was tested to ensure that the module operates properly at boundaries established to limit or restrict processing.
- All independent paths through the control structures were executed to ensure that all statements in the modules have been executed at least once.
- Error handling paths were also tested.

2. Integration Testing

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing.

Unit-tested modules were taken and a single program structure was built that has been dictated by the design. Incremental integration has been adopted here.

The modules are tested separately for accuracy and modules are integrated too.th tn. using bottom-up integration i.e., by integrating from moving from the bottom to the top of the system is checked and errors found during integration are rectified. In this testing individual modules were combined and the module-wise Shifting was verified to be right. The entire software was developed and tested in small segments, where errors were easy to locate and rectify. The program builds (group of modules) were constructed corresponding to the successful testing of user interaction, data manipulation analysis, display processing, and database management.

3. Validation Testing

Validation testing is done to ensure complete assembly of the error-free software. Validation can be termed successful only if it functions in a manner. Reasonably expected by the student under validation is alpha and beta testing. The student-side validation is done in this testing phase. It is checked whether the data passed to each student is valid or not. Entering incorrect values does the validation testing and it is checked whether the errors are being considered. Incorrect values are to be discarded. The errors are rectified.

In “ProTrack" verifications are done correctly. So, there is no chance for users to enter incorrect values. It will give error messages by using different validations. The validation testing is done very clearly and it is error-free.

4. Output Testing

After performing the validation testing the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in a specific format.

The output format on the screen was found to be correct as the format was designed in the system design phase according to the user's needs. For the hard copy also, the output comes out as specified requirement by the user. Hence output testing does not result in any Correction in the system.

Output This project is developed based on the user's choice. It is user-friendly. The output format is very clear to the user.

5. Acceptance testing

Acceptance involves running a suite of tests on the completed system. Each individual test, known as a Case, exercises a particular operating condition of the operating condition of the user's environment or feature of the system and will result in a pass-fail, or Boolean outcome.

4.2 SYSTEM IMPLEMENTATION

The implementation is the final stage, and it is an important phase. It involves invalid programming system testing. user training and the operational running of the developed proposed system that constitutes the application subsystems. A major task in preparing for implementation is the education of users. which should really have taken place much carrier in the project when they were belong involved in the investigation and design work. During the implementation phase system takes physical shape. In order to develop a system implemented planning is very essential.

The implementation phase of the software development is concerned with translating design specifications into source code. The user tests the developed system and changes are made according to their needs. Our system has been successfully implemented.

Before implementation several tests have been conducted to ensure that no errors are encountered during the operation. The implementation phase ends with an evaluation of the system after placing into operation for a period of time.

The process of putting the developed system into actual use is called system implementation. This includes all those activities that take place to convert from the old system to new system. The system can be implemented only after testing is done and is found to be working to specifications. The implementation stage is a systems project in its own right.

The implementation stage involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of method to achieve changeover.
- Evaluation of the changeover method.

In the case of this project all the screens are designed first. To make it to be executable, codes are written on each screen, and performs the implementation by creating the database and connecting to the server. After that the system is Checked, whether it performs all the transactions correctly. Then databases are cleared and made it to be usable to the technicians.

Implementation Plans

The following are the steps involved in the implementation plan of " TinyTots daycare system":

- Test system with sample data
- Detection and correction of errors
- Make the necessary changes in the system.
- Check the existing system.
- Installation of hardware and software utilities
- Training and involvement of user personnel

5. SECURITY TECHNOLOGIES AND POLICIES

The protection of computer-based resources that includes hardware, software, data procedures and people against unauthorized use or natural disaster is known as System Security.

System Security can be divided into four related issues:

- Security
- Integrity
- Privacy
- Confidentiality

SYSTEM SECURITY refers to the technical innovations and procedures applied to the hardware and operation systems to protect against deliberate or accidental damage from a defined threat.

DATA SECURITY is the protection of data from loss, disclosure, modification, and destruction.

SYSTEM INTEGRITY refers to the power functioning of hardware and programs, appropriate physical security and safety against external threats such as eavesdropping and wiretapping.

PRIVACY defines the rights of the user or organizations to determine what information they are willing to share with or accept from others and how the organization can be protected against unwelcome, unfair, or excessive dissemination of information about it.

CONFIDENTIALITY is a special status given to sensitive information in a database to minimize the possible invasion of privacy. it is an attribute of information that characterizes its need for protection.

SECURITY IN SOFTWARE System security refers to various validations of data in the form of checks and controls to prevent the system from failing. It is always important to ensure that only valid data is entered, and only valid operations are performed on the system.

The system employees two types of checks and controls:

CLIENT-SIDE VALIDATION

Various client-side validations are used to ensure on the client side that only valid data is entered. Client-side validation saves server time and load to handle invalid data. Some checks imposed are:

- Forms cannot be submitted without filling up the mandatory data so that manual mistakes of submitting empty fields that are mandatory can be sorted out on the client side to save the server time and load.
- Tab indexes are set according to the need and take into account the ease of the user while working with the system.

SERVER-SIDE VALIDATION

Some checks cannot be applied on the client side. Server-side checks are necessary to save the system from failing and inform the user that some invalid operation has been performed or the performed operation is restricted. Some of the server-side checks imposed are:

- Server-side constraint has been imposed to check for the validity of the primary key and foreign key. A primary key value cannot be duplicated. Any attempt to duplicate the primary value results in a message intimating the user about those values through the forms using foreign keys can be updated only with the existing foreign key values.
- User is intimating through appropriate messages about the successful operations or exceptions occurring at the server side.
- Various Access Control Mechanisms have been built so that one user may not agitate upon another. Access permissions to various types of users are controlled according to the organizational structure. Only permitted users can log on to the system and can have access according to their category. User- names, passwords, and permissions are controlled on the server side.
- Using server-side validation, constraints on several restricted operations are imposed.

6. MAINTENANCE

Software maintenance is the modification of a software product and delivery to correct faults, and improve performance, or other attributes. Maintenance is the ease with which a program can be corrected if any error is encountered, adapted if its environment changes or enhanced if the customer desires a change in requirement. Maintenance follows conversation to extend that changes are necessary to maintain satisfactory operations relative to changes in the user's environment.

Maintenance often includes minor enhancements or corrections to problems that surface in the system's operation. Maintenance is also done based on fixing the problems reported, changing the interface with other software or hardware enhancing the software.

CATEGORIES OF MAINTENANCE

Corrective Maintenance

Corrective maintenance is the most used maintenance approach, but it is easy to see its limitations. When equipment fails, it often leads to downtime in production, and sometimes damages other parts. In most cases, this is expensive. Also, if the equipment needs to be replaced, the cost of replacing it alone can be substantial. The reliability of systems maintained by this type of maintenance is unknown and cannot be measured. Corrective maintenance is possible since the consequences of failure or wearing out are not significant and the cost of this maintenance is not great.

Perfective Maintenance

Modification of a software product alters delivery to improve performance or maintainability. This term is used to describe changes undertaken to expand the existing requirements of the system. A successful piece of software lends to be subjected to the Succession of changes resulting in an increase in our requirements. This is based on the premise that as the software becomes useful, the user experiments with new cases beyond the of

Scope for which it was initially developed. Vxpansi01 n requirements can take the form of enhancement of existing system functionality and improvement in computational efficiency.

Adaptive Maintenance

Modification of a software product performed after delivery to keep a product usable, changed or changing environment. Adaptive maintenance includes any work initiated because of moving the software to a different hardware or software platform. It is a change driven by the need to accommodate modifications in the environment of a software system. The environment in this context refers to the totality of all conditions and influences which act from outside upon the system. A change to the whole or part of this environment will Warrant a corresponding modification of the software.

Preventive Maintenance

Preventive maintenance is a schedule of planned maintenance actions aimed at the prevention of breakdowns and failures. The primary goal of preventive maintenance is to prevent the failure of equipment before it occurs. It is designed to preserve and enhance equipment reliability by replacing worn components before they fail. Preventive maintenance activities include equipment checks, and partial or complete overhauls at specified periods.

Long-term benefits of preventive maintenance include:

- Improved system reliability.
- Decreased cost of replacement.
- Decreased system downtime.

7. SCOPE FOR FUTURE ENHANCEMENT

In the future, the ProTrack system can be enhanced with advanced technologies to further streamline project management and improve academic collaboration. AI-powered features such as automated project topic recommendations, plagiarism detection, and intelligent feedback analysis can enhance evaluation processes. Machine learning algorithms could track student progress and suggest personalized improvements. Cloud-based integration will ensure seamless remote access, secure storage, and real-time collaboration among students and faculty. Blockchain technology can be implemented for secure and tamper-proof record-keeping of project submissions and evaluations. Additionally, a mobile application with push notifications will enable students and teachers to manage projects on the go. Augmented Reality (AR) can be incorporated for interactive project presentations, making evaluations more engaging. Integration with external research databases and industry mentorship platforms will provide students with valuable insights and networking opportunities. Furthermore, multilingual support and accessibility features will make the system more inclusive for a diverse academic community. By leveraging these technological advancements, ProTrack can evolve into a comprehensive, smart project management solution that enhances efficiency, transparency, and collaboration in educational institutions.

8. CONCLUSION

The ProTrack system revolutionizes project tracking and management within academic institutions by providing a structured, digital platform for students, teachers, and administrators. By eliminating manual paperwork and scattered communication, it ensures seamless collaboration, secure document storage, and efficient project evaluation. Features like real-time feedback, automated notifications, and analytics enhance productivity and transparency. The system not only simplifies project management but also fosters a more organized and engaging academic environment. With potential future enhancements such as AI-based recommendations, cloud integration, and blockchain security, ProTrack can continue evolving to meet the growing needs of modern education. Ultimately, it serves as a powerful tool to streamline project workflows, improve academic outcomes, and enhance overall efficiency.

9. BIBLIOGRAPHY

BOOKS:

- Marco L. Napoli, *Beginning Flutter: A Hands-On Guide to App Development*
- Eric Windmill, *Flutter in Action*
- Marco L. Napoli, *Mastering Dart: Advanced Techniques for Modern Applications*
- Slobodan Stojanović, *Serverless Applications with Node.js*
- Shyam Seshadri, *Building Scalable Apps with Firebase*
- Elias M. Awad, *System Analysis and Design*

Websites:

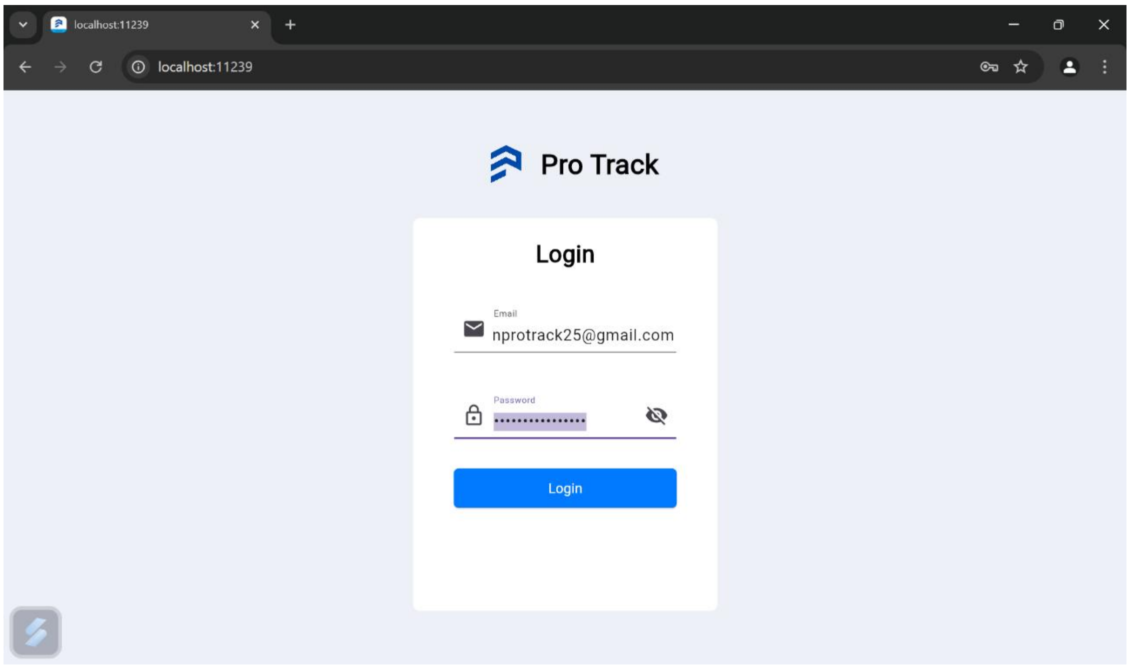
- https://flutter.dev
- https://dart.dev
- shttps://supabase.com
- https://flutterawesome.com
- https://developer.android.com
- https://medium.com
- https://stackoverflow.com
- https://pub.dev

10. APPENDIX

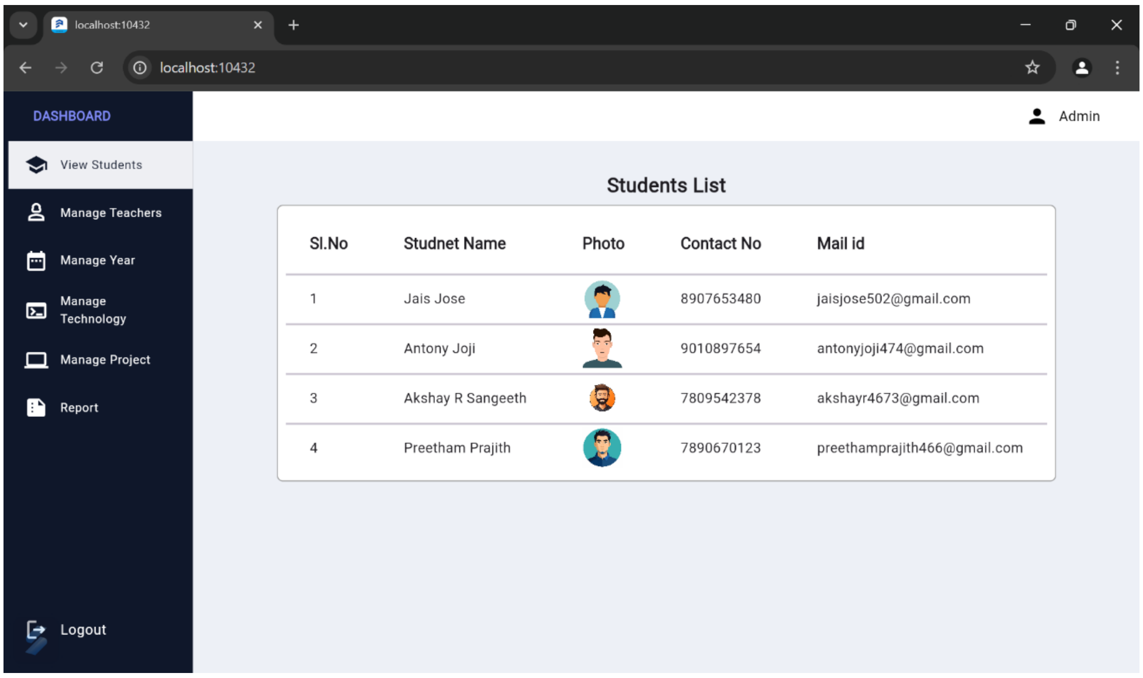
10.1 SCREENSHOTS

Admin

Login Page



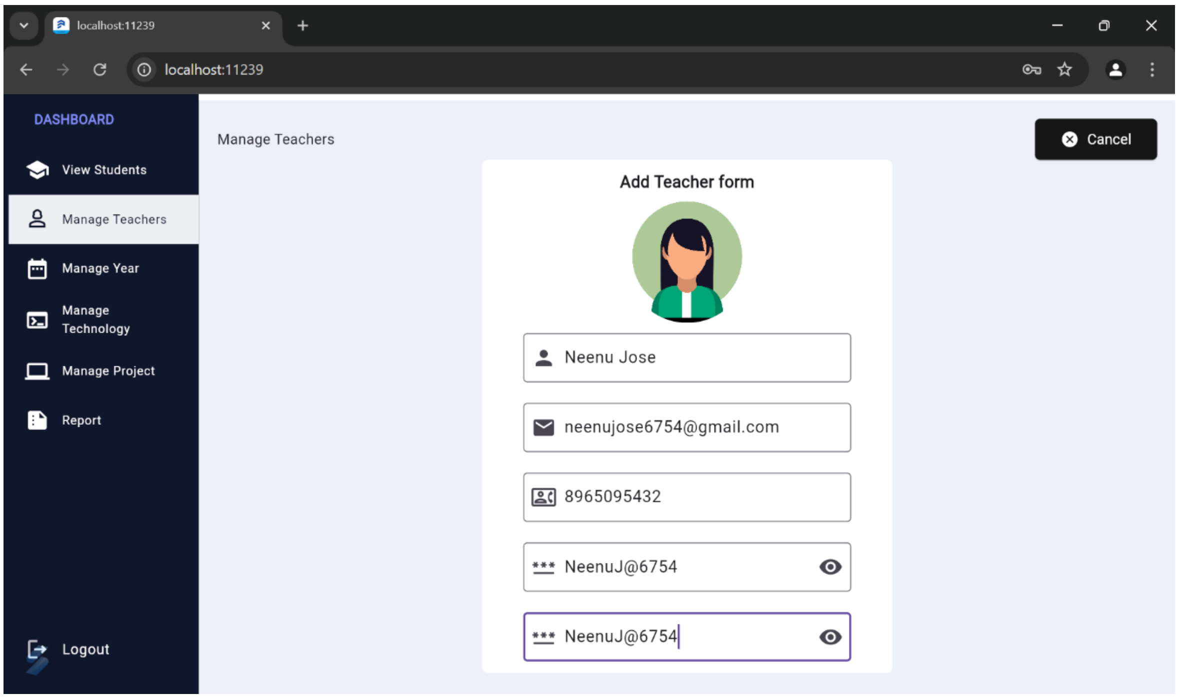
Home Page



Students List

Sl.No	Studnet Name	Photo	Contact No	Mail id
1	Jais Jose		8907653480	jaisjose502@gmail.com
2	Antony Joji		9010897654	antonyjoji474@gmail.com
3	Akshay R Sangeeth		7809542378	akshayr4673@gmail.com
4	Preetham Prajith		7890670123	preethamprajith466@gmail.com

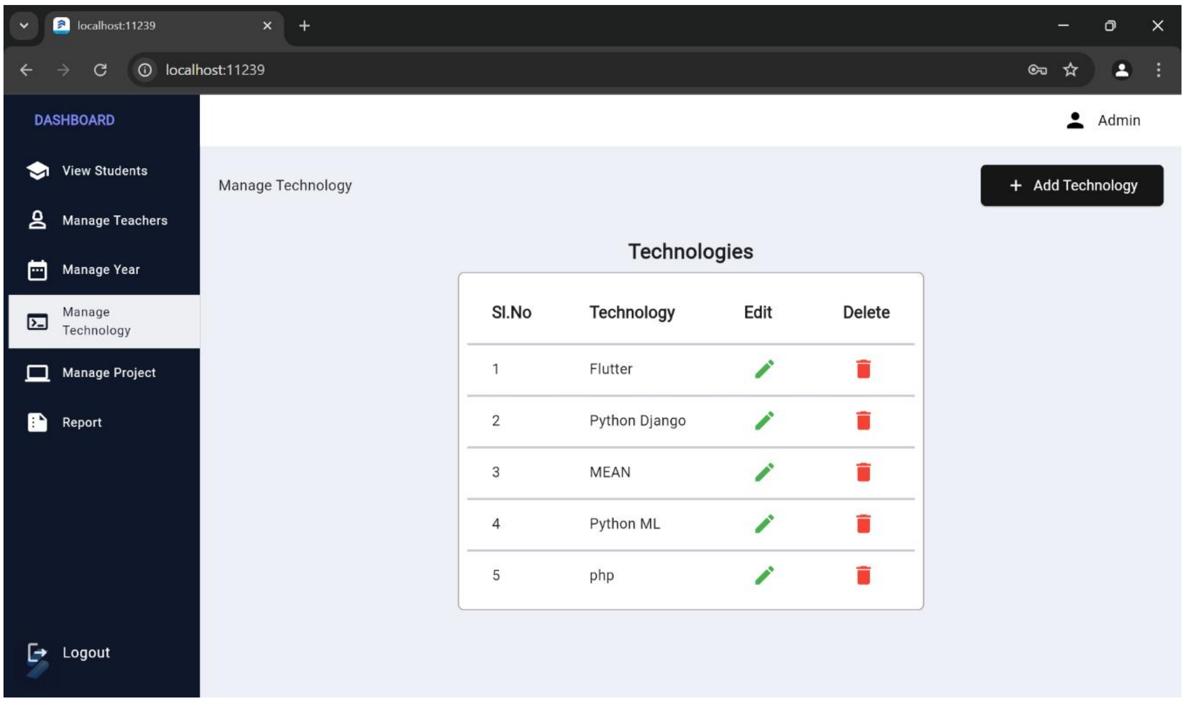
Add Teacher



Add Teacher form

Name	Neenu Jose
Email	neenujose6754@gmail.com
Phone	8965095432
Password	NeenuJ@6754
Confirm password	NeenuJ@6754

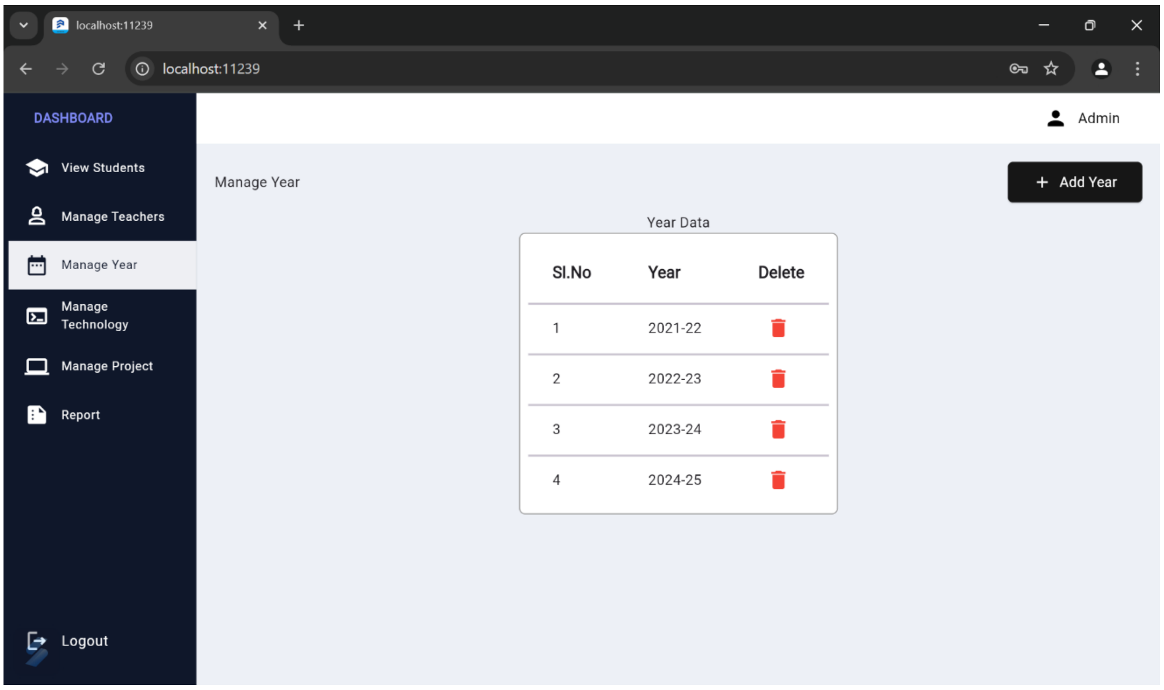
Manage Technology



Technologies

Sl.No	Technology	Edit	Delete
1	Flutter	✎	🗑
2	Python Django	✎	🗑
3	MEAN	✎	🗑
4	Python ML	✎	🗑
5	php	✎	🗑

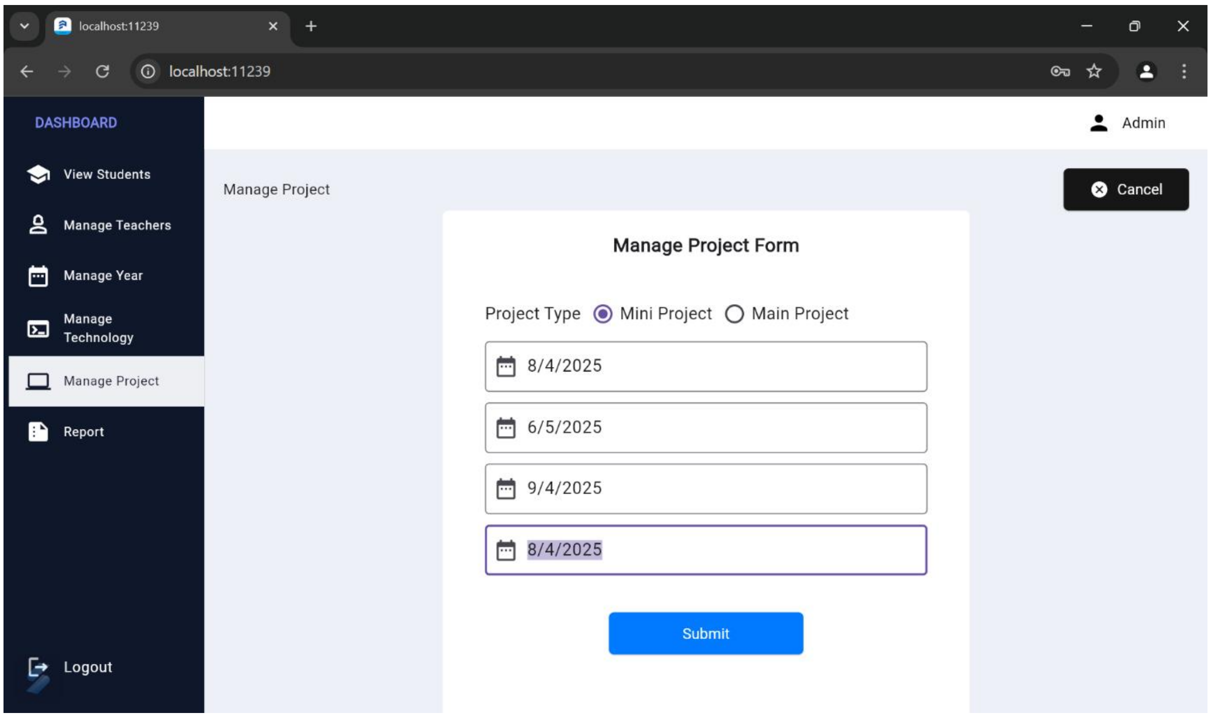
Manage Year



Year Data

Sl.No	Year	Delete
1	2021-22	
2	2022-23	
3	2023-24	
4	2024-25	

Manage Project



Manage Project Form

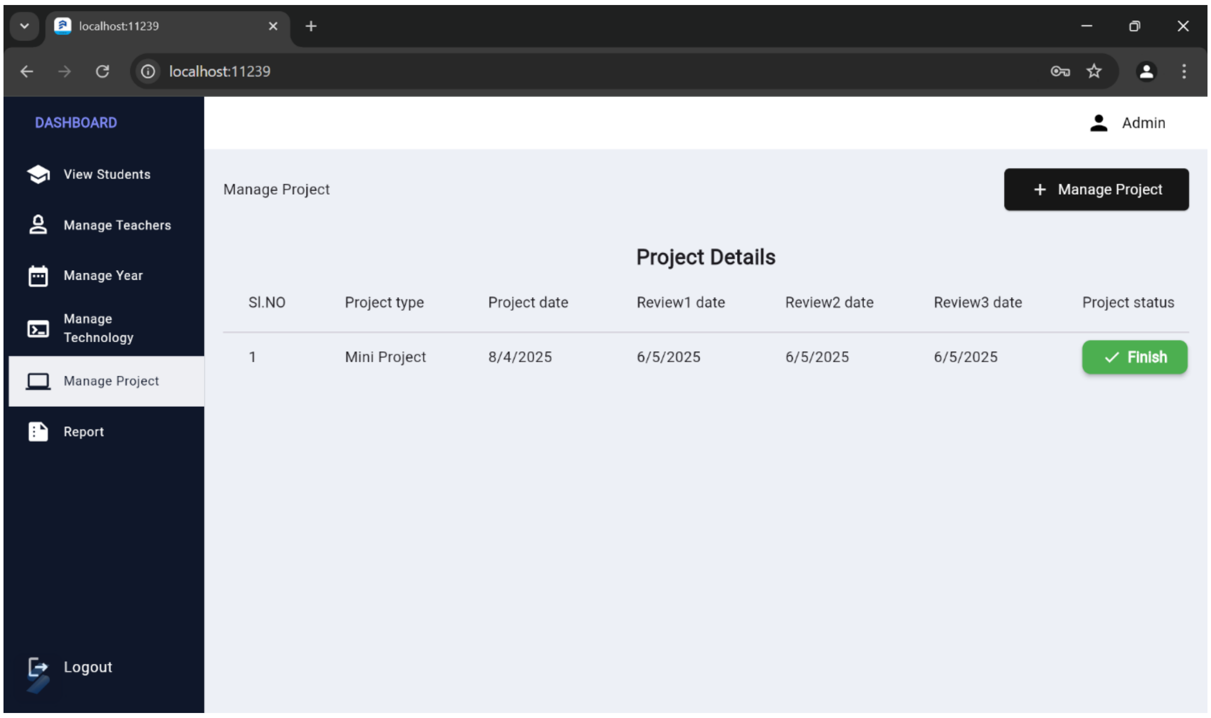
Project Type [✓ Mini Project] Main Project

8/4/2025

6/5/2025

9/4/2025

8/4/2025

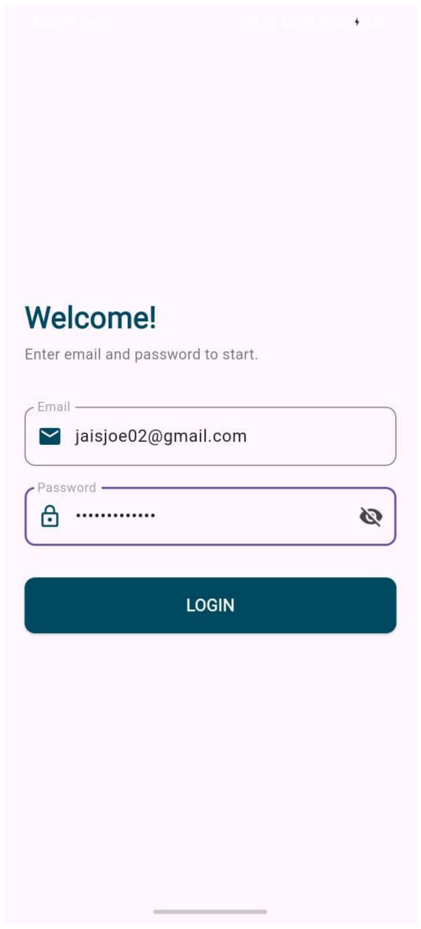


Project Details

Sl.NO	Project type	Project date	Review1 date	Review2 date	Review3 date	Project status
1	Mini Project	8/4/2025	6/5/2025	6/5/2025	6/5/2025	✓ Finish

Teacher

Login

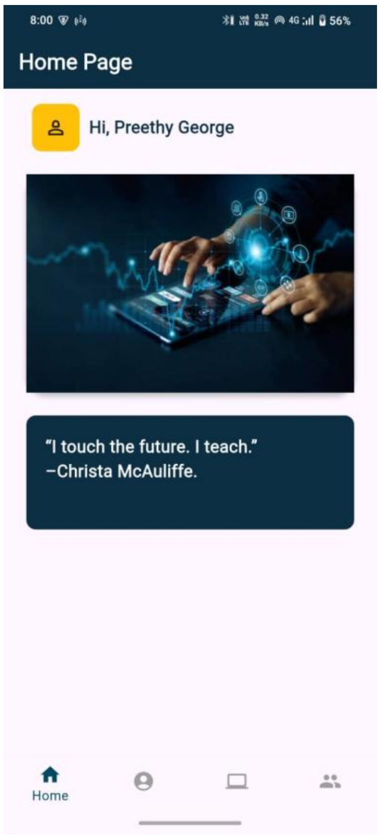

Welcome!

Enter email and password to start.

Email: jaisjoe02@gmail.com

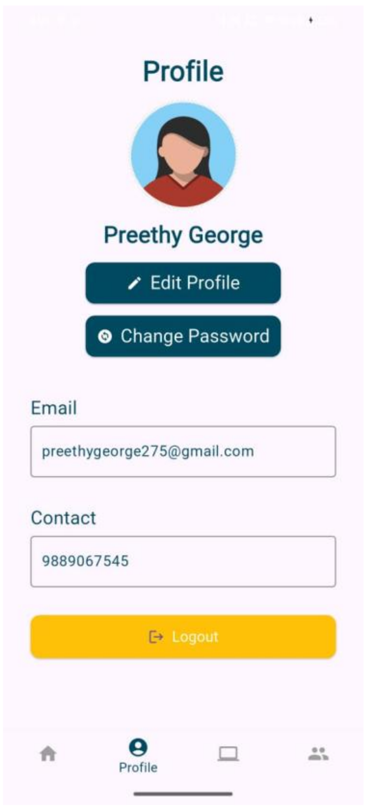
Password: •••••••••••••

HomePage



"I touch the future. I teach."
–Christa McAuliffe.

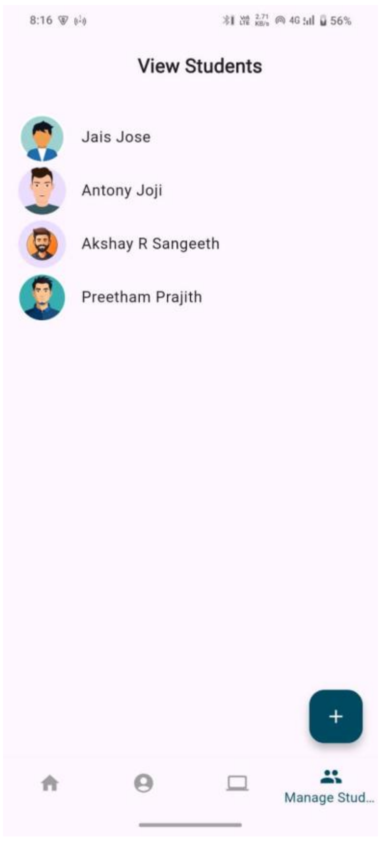
Profile



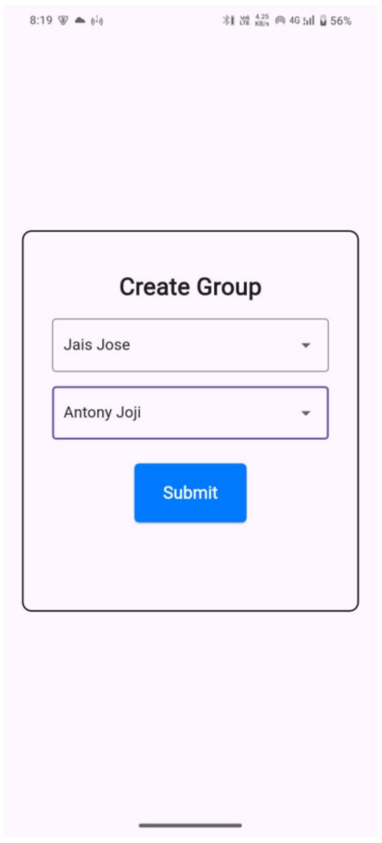
Email: preethygeorge275@gmail.com

Contact: 9889067545

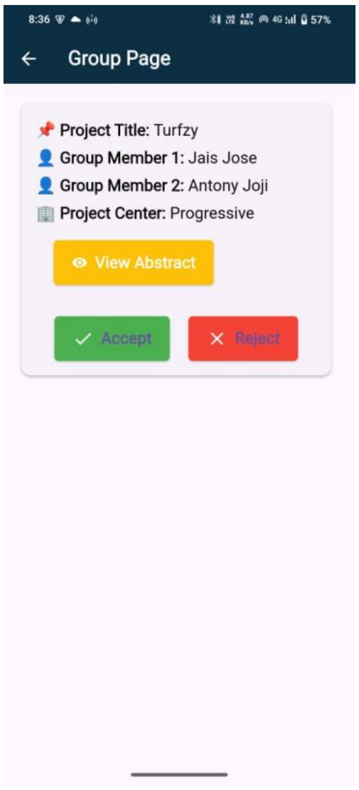
View Students



Create Group

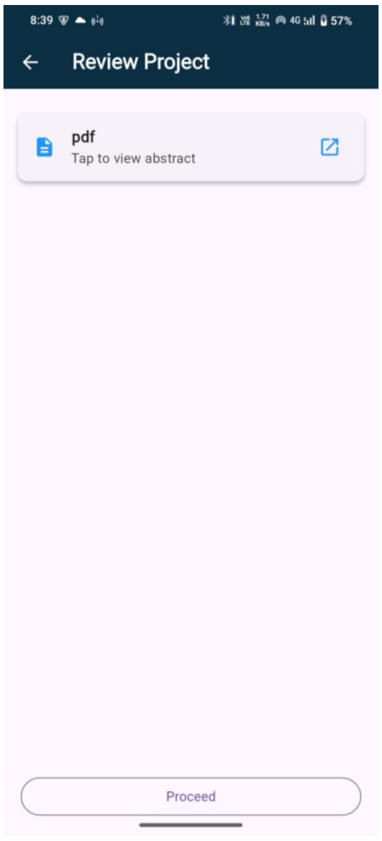


Group Page

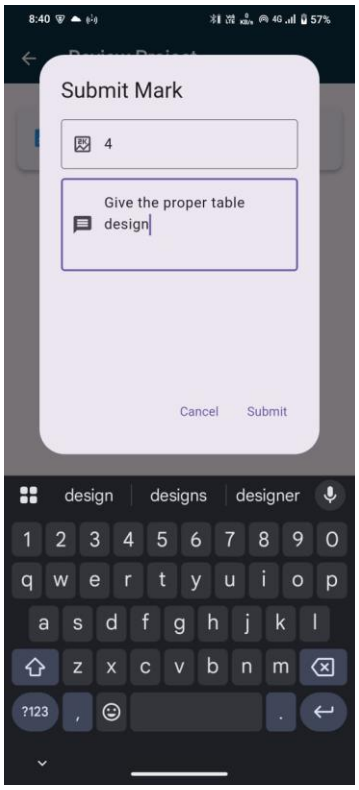


📌 **Project Title:** Turfzy
👤 **Group Member 1:** Jais Jose
👤 **Group Member 2:** Antony Joji
🏢 **Project Center:** Progressive

Review Page



Review Mark



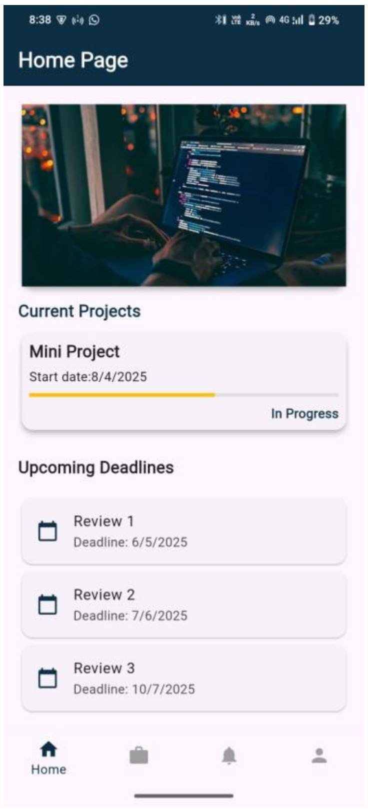
Submit Mark

4

Give the proper table design

STUDENT

Home Page



Home Page

Current Projects

Mini Project
Start date:8/4/2025
In Progress

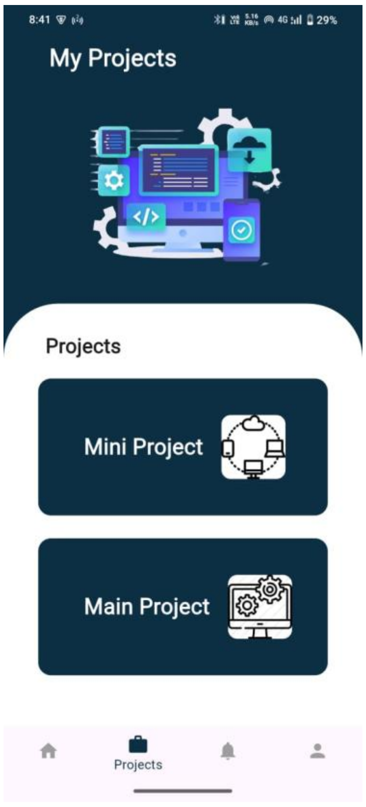
Upcoming Deadlines

Review 1
Deadline: 6/5/2025

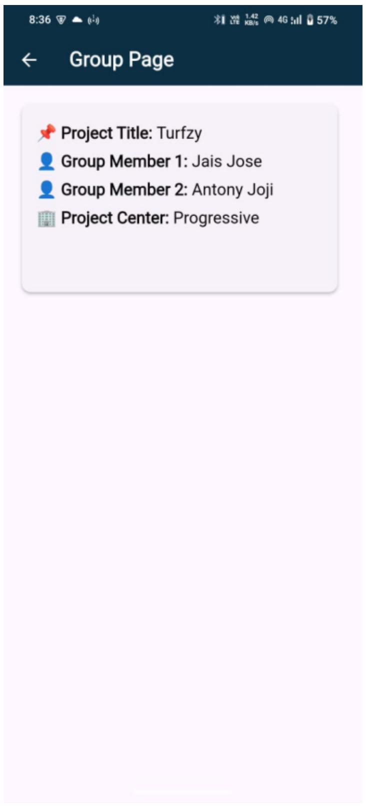
Review 2
Deadline: 7/6/2025

Review 3
Deadline: 10/7/2025

Projects

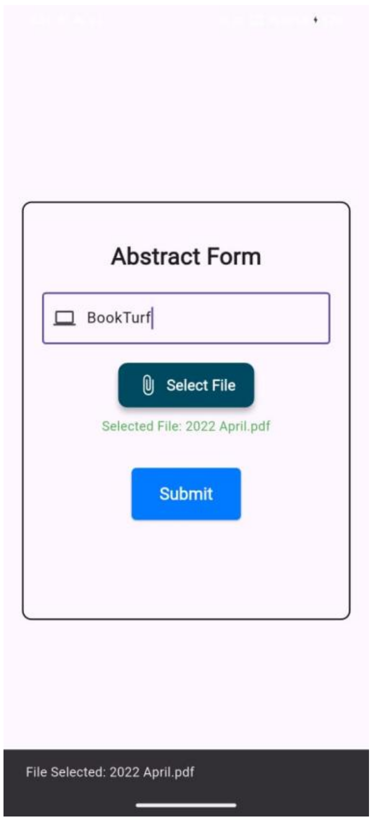


GroupPage

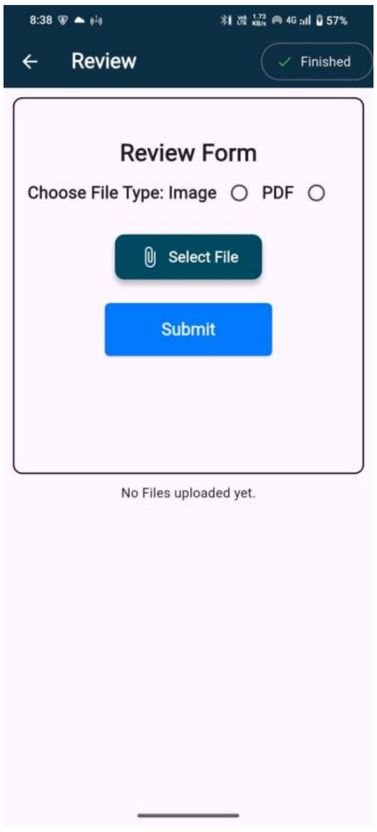


📌 **Project Title:** Turfzy
👤 **Group Member 1:** Jais Jose
👤 **Group Member 2:** Antony Joji
🏢 **Project Center:** Progressive

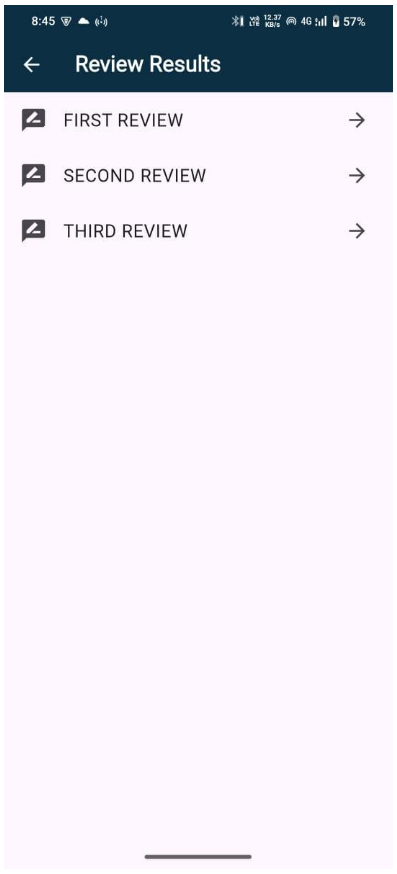
Abstract Form

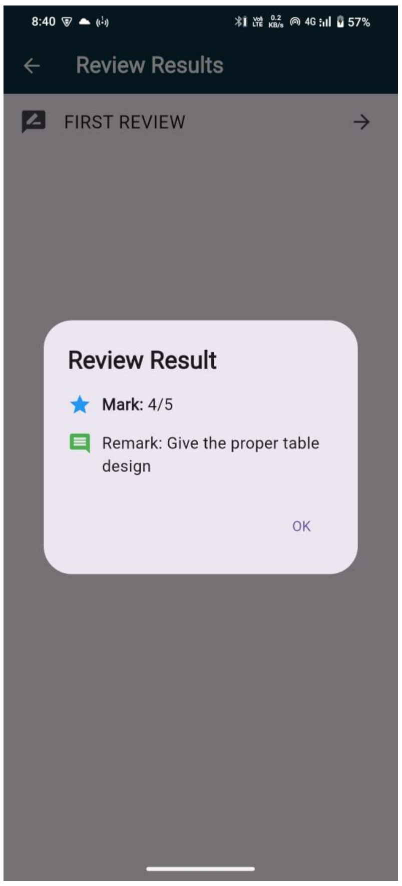


Review Form



Review Results





Review Result

Mark: 4/5

Remark: Give the proper table design

10.2 CODE

Main.dart

```
import 'package:flutter/material.dart';

import 'package:protrack_student/screens/newlogin.dart';

import 'package:supabase_flutter/supabase_flutter.dart';

Future<void> main() async {

  await Supabase.initialize(

    url: 'https://jjgpiuqajneevdvwhmku.supabase.co',

    anonKey:


'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6ImpqZ3Bpd
XFham5lZXZkdndobWt1Iiwicm9sZSI6ImFub24iLCJpYXQiOjE3MzQzNDY3ODQsImV4cCI6
MjA0OTkyMjc4NH0.D6huZEvSmiouOe_Tzqky0FcJV_bOrAOA1PrtWjnzETM',

  );

  runApp(const MainApp());

}

final supabase = Supabase.instance.client;



class MainApp extends StatelessWidget {

  const MainApp({super.key});

  @override

  Widget build(BuildContext context) {

    return const MaterialApp(

        debugShowCheckedModeBanner: false, home: NewLoginPage());

  }
}
```

Loign.dart

```
import 'package:flutter/material.dart';

import 'package:protrack_student/main.dart';

import 'package:protrack_student/screens/dashboard.dart';

import 'package:cherry_toast/resources/arrays.dart';

import 'package:cherry_toast/cherry_toast.dart';


class LoginScreen extends StatefulWidget {

  const LoginScreen({super.key});


  @override

  State<LoginScreen> createState() => _LoginScreenState();

}


class _LoginScreenState extends State<LoginScreen> {

  bool passkey = true;

  final TextEditingController _passwordController = TextEditingController();

  final TextEditingController _emailController = TextEditingController();


  Future<void> signIn() async {

    try {

      await supabase.auth.signInWithPassword(

          password: _passwordController.text, email: _emailController.text);
    Navigator.pushReplacement(
      context,
      MaterialPageRoute(
        builder: (context) => Dashboard(),
      ));
   } catch (e) {
    print("Error occur in login:$e");
    CherryToast.error(
          description: Text("Invalid user name or password",
              style: TextStyle(color: Colors.black)),
          animationType: AnimationType.fromRight,
          animationDuration: Duration(milliseconds: 1000),
          autoDismiss: true)
        .show(context);
    print('No user found for that email.');
   }
  }

  @override
  Widget build(BuildContext context) {
   return Scaffold(
    backgroundColor: Color(0xFFEDF0F6),
    body: SingleChildScrollView(
     child: Form(
        child: Center(
      child: Column(
        children: [
          SizedBox(
            height: 170,
          ),
          Row(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              SizedBox(height: 100),
              Padding(
                padding: const EdgeInsets.all(15),
                child: Image.asset(
                  'assets/Logo1.png',
                  width: 40,
                  height: 40,
                ),
              ),
              Text(
                'Pro Track',
                style: TextStyle(
                  fontSize: 28,
                  fontWeight: FontWeight.bold,
                  color: Colors.black,
                ),
              ),
            ],
          ),
          SizedBox(
            height: 5,
          ),
          Container(
            width: 310,
            height: 400,
            decoration: BoxDecoration(
              borderRadius: BorderRadius.circular(7),
              color: Color(0xFFFFFFFF),
            ),
            child: Column(
              children: [
                SizedBox(
                  height: 20,
                ),
                Text(
                  'Login',
                  style: TextStyle(
                    fontSize: 24,
                    fontWeight: FontWeight.bold,

                    color: Colors.black,

                  ),

                ),

                SizedBox(

                  height: 5,

                ),

                Padding(

                  padding: const EdgeInsets.all(20.0),

                  child: Padding(

                    padding: const EdgeInsets.only(left: 22, right: 22),

                    child: TextFormField(

                      controller: _emailController,

                      decoration: InputDecoration(

                          labelText: "Email",

                          labelStyle: TextStyle(fontSize: 12),

                          hintStyle: TextStyle(fontSize: 11),

                          hintText: 'Enter your email',

                          border: UnderlineInputBorder(),

                          prefixIcon: Icon(Icons.email)),

                    ),

                  ),

                ),

                Padding(
                  padding: const EdgeInsets.all(20.0),
                  child: Padding(
                    padding: const EdgeInsets.only(left: 22, right: 22),
                    child: TextFormField(
                      controller: _passwordController,
                      decoration: InputDecoration(
                          labelText: 'Password',
                          labelStyle: TextStyle(fontSize: 12),
                          hintStyle: TextStyle(fontSize: 11),
                          hintText: 'Enter your password',
                          border: UnderlineInputBorder(),
                          prefixIcon: Icon(Icons.password),
                          suffixIcon: IconButton(
                            icon: Icon(passkey
                                ? Icons.visibility_off
                                : Icons.visibility),
                            onPressed: () {
                              setState(() {
                                passkey = !passkey;
                              });
                            },
                          )),
                      obscureText: passkey,
                    ),
                  ),
                ),
                Row(
                  mainAxisAlignment: MainAxisAlignment.end,
                  children: [
                    Padding(
                      padding: const EdgeInsets.only(
                          right: 39, top: 10, bottom: 10),
                      child: Text(
                        'Forgot password?',
                        style: TextStyle(
                            color: Color(0xFF017AFF), fontSize: 12),
                      ),
                    ),
                  ],
                ),
                Padding(
                  padding: const EdgeInsets.all(10.0),
                  child: ElevatedButton(
                      style: ElevatedButton.styleFrom(
                          backgroundColor: Color(0xFF017AFF),
                          padding: EdgeInsets.symmetric(
                              horizontal: 96, vertical: 18),
                          shape: RoundedRectangleBorder(
                              borderRadius: BorderRadius.circular(5))),
                      onPressed: () {
                        signIn();
                      },
                      child: Text(
                        'Login',
                        style: TextStyle(color: Colors.white),
                      )),
                )
              ],
            ),
          )
        ],
      ),
     )),
    ),
    );
    }
    }
```

HomePage.dart

```
import 'package:flutter/material.dart';

import 'package:protrack_student/main.dart';
```

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


  @override
  State<Homepage> createState() => _HomepageState();
}


class _HomepageState extends State<Homepage> {
  String projectType = "";
  String startDate = "";
  String review1Date = "";
  String review2Date = "";
  String review3Date = "";


  Future<void> FetchProject() async {
    try {
      final response = await supabase
          .from('tbl_project')
          .select()
          .eq('project_status', 1)
          .single()
          .limit(1);


      setState(() {
        projectType = response['project_type'];
        startDate = response['project_date'];
        review1Date = response['project_review1'];
        review2Date = response['project_review2'];
        review3Date = response['project_review3'];
      });
    } catch (e) {
      print("Error fetching project data:$e");
    }
  }

  @override
  void initState() {
    // TODO: implement initState
    super.initState();
    FetchProject();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        iconTheme: IconThemeData(color: Colors.white),
        backgroundColor: const Color.fromARGB(255, 12, 47, 68),
        title: Text(
          "Home Page",
          style: TextStyle(
              fontSize: 23, fontWeight: FontWeight.bold, color: Colors.white),
        ),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: SingleChildScrollView(
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Card(
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(12),
                ),
                elevation: 4,
                child: Image.asset(
                  'assets/tech.avif',
                  fit: BoxFit.cover,
                ),
              ),
              SizedBox(
                height: 10,
              ),
              Text(
                'Current Projects',
                style: TextStyle(
                    fontSize: 18,
                    fontWeight: FontWeight.bold,
                    color: const Color.fromARGB(255, 12, 47, 68)),
              ),
              SizedBox(height: 6),
              Card(
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(12),
                ),
                elevation: 4,
                child: Padding(
                  padding: const EdgeInsets.all(
                      8.0), // Add padding for better spacing
                  child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [
                      Text(
                        projectType,
                        style: TextStyle(
                            fontSize: 18, fontWeight: FontWeight.bold),
                      ),
                      SizedBox(height: 4), // Space between title and subtitle
                      Text("Start date:$startDate"),
                      SizedBox(
                          height: 8), // Space between subtitle and progress bar
                      LinearProgressIndicator(
                        value: 0.6, // Change this value dynamically (0.0 - 1.0)
                        backgroundColor: Colors.grey[300],
                        valueColor: AlwaysStoppedAnimation<Color>(Colors.amber),
                      ),
                      SizedBox(
                          height: 8), // Space between progress bar and status
                      Align(
                        alignment: Alignment.centerRight,
                        child: Text(
                          'In Progress',
                          style: TextStyle(
                              color: const Color.fromARGB(255, 12, 47, 68),
                              fontWeight: FontWeight.bold),
                        ),
                      ),
                    ],
                  ),
                ),
              ),
              SizedBox(height: 24),
              Text(
                'Upcoming Deadlines',
                style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
              ),
              SizedBox(height: 16),
              Card(
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(12),
                ),
                child: ListTile(
                  leading: Icon(Icons.calendar_today,
                      color: const Color.fromARGB(255, 12, 47, 68)),
                  title: Text('Review 1'),
                  subtitle: Text('Deadline: $review1Date'),
                ),
              ),
              Card(
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(12),
                ),
                child: ListTile(
                  leading: Icon(Icons.calendar_today,
                      color: const Color.fromARGB(255, 12, 47, 68)),
                  title: Text('Review 2'),
                  subtitle: Text('Deadline: $review2Date'),
                ),
              ),
              Card(
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(12),
                ),
                child: ListTile(
                  leading: Icon(Icons.calendar_today,
                      color: const Color.fromARGB(255, 12, 47, 68)),
                  title: Text('Review 3'),
                  subtitle: Text('Deadline: $review3Date'),
                ),
              ),
              SizedBox(
                height: 5,
              ),
            ],
          ),
        ),
      ),
    );
    }
}
```

Dashboard.dart

```
import 'package:flutter/material.dart';
import 'package:protrack_student/screens/homepage.dart';
import 'package:protrack_student/screens/myaccount.dart';
import 'package:protrack_student/screens/myprojects.dart';
import 'package:protrack_student/screens/notification.dart';

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

  @override
  State<Dashboard> createState() => _DashboardState();
}

class _DashboardState extends State<Dashboard> {
  int _selectedIndex = 0;
  final List<Widget> items = [
    Homepage(),
    ProjectScreen(),
    NotificationPage(),
    Account(),
  ];
```

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    body: items[_selectedIndex],
    // body: DashBoard(),
    bottomNavigationBar: BottomNavigationBar(
        showUnselectedLabels: false,
        showSelectedLabels: true,
        type: BottomNavigationBarType.fixed,
        selectedItemColor: const Color.fromARGB(255, 12, 47, 68),
        unselectedItemColor: Colors.grey,
        currentIndex: _selectedIndex,
        onTap: (index) {
          setState(() {
            _selectedIndex = index;
          });
          items[_selectedIndex];
        },
        items: [
          BottomNavigationBarItem(
            icon: Icon(Icons.home),
            label: 'Home',
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.work),
            label: 'Projects',
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.notifications),
            label: 'Notifications',
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.person),
            label: 'Profile',
          ),
        ]),
    );
  }
}
```

MiniProject.dart

```
import 'package:flutter/material.dart';
import 'package:protrack_student/main.dart';
import 'package:protrack_student/screens/abstractform.dart';
import 'package:protrack_student/screens/resubmitabstract.dart';
import 'package:protrack_student/screens/reviewpage.dart';
import 'package:protrack_student/screens/reviewresults.dart';
import 'package:protrack_student/screens/viewabstract.dart';
```

```
class Miniproject extends StatefulWidget {
  final int id;
  const Miniproject({super.key, required this.id});


  @override
  State<Miniproject> createState() => _MiniprojectState();
}


class _MiniprojectState extends State<Miniproject> {
  String projectTitle = "";
  String projectCenter = "";
  String groupmember = "";
  String guide = "";
  int status = 0;
  int gid = 0;
  bool isLoading = false;
  bool isProjectCompleted = true;
  Future<void> fetchProject() async {
    try {
      final response = await supabase
          .from('tbl_groupmember')
          .select('tbl_group(*)')
          .eq('student_id', supabase.auth.currentUser!.id)
          .eq('tbl_group.project_id', widget.id)
          .order('created_at', ascending: false) // Get the latest
          .limit(1) // Restrict to one row
          .maybeSingle();
      if (response != null) {
        final groupId = response['tbl_group']['group_id'] as int;
        print("Group ID: $groupId");
        final response3 = await supabase
            .from('tbl_groupmember')
            .select(' *, tbl_student(*)')
            .eq('group_id', groupId)
            .neq('student_id', supabase.auth.currentUser!.id)
            .single();
        String member = response3['tbl_student']['student_name'];
        final response2 = await supabase
            .from('tbl_teacher')
            .select()
            .eq('teacher_id', response3['tbl_student']['teacher_id'])
            .single();
        String teacher = response2['teacher_name'];
        setState(() {
          projectTitle =
              response['tbl_group']['project_title'] ?? "NOT ASSIGNED";
          projectCenter =
            response['tbl_group']['group_center'] ?? "NOT ASSIGNED";
          guide = teacher;
          groupmember = member;
          status = response['tbl_group']['group_status'];
          gid = groupId;
          isLoading = false;
          if (status == 9) {
            isProjectCompleted = true;
          }
        });
      } else {
        print("No matching group found");
        setState(() {
          isLoading = false;
        });
      }

      // print("Status:$status");
      // print("Gid:$gid");
    } catch (e) {
      print("Error fetching project:$e");
    }
  }
```

```
 Future<void> addReview1() async {
   try {
     DateTime date = DateTime.now();
     String formattedDate = "${date.year}-${date.month}-${date.day}";

     final response = await supabase
         .from('tbl_review')
         .insert({
           'review_date': formattedDate,
           'review_type': "FIRST",
           'group_id': gid,
         })
         .select()
         .single();
     int id = response['review_id'];
     Navigator.push(
         context,
         MaterialPageRoute(
             builder: (context) => ReviewPage(
                   reviewId: id,
                   type: "FIRST",
                 )));
     print("Review details added");
   } catch (e) {
    print("Error inserting review details:$e");

   }

  }


  Future<void> addReview2() async {

   try {

    DateTime date = DateTime.now();

    String formattedDate = "${date.year}-${date.month}-${date.day}";


    final response = await supabase

        .from('tbl_review')

        .insert({

         'review_date': formattedDate,

         'review_type': "SECOND",

         'group_id': gid,

        })

        .select()

        .single();

    int id = response['review_id'];

    Navigator.push(

        context,

        MaterialPageRoute(

            builder: (context) => ReviewPage(

                  reviewId: id,
                  type: "SECOND",
                )));
        print("Review details added");
      } catch (e) {
        print("Error inserting review details:$e");
      }
    }

    Future<void> addReview3() async {
      try {
        DateTime date = DateTime.now();
        String formattedDate = "${date.year}-${date.month}-${date.day}";

        final response = await supabase
            .from('tbl_review')
            .insert({
              'review_date': formattedDate,
              'review_type': "THIRD",
              'group_id': gid,
            })
            .select()
            .single();
        int id = response['review_id'];
        Navigator.push(
            context,
            MaterialPageRoute(
                builder: (context) => ReviewPage(
                  reviewId: id,
                  type: "THIRD",
                )));
        print("Review details added");
      } catch (e) {
        print("Error inserting review details:$e");
      }
    }

    @override
    void initState() {
      // TODO: implement initState
      super.initState();
      fetchProject();
    }

    @override
    Widget build(BuildContext context) {
      return isLoading
          ? Scaffold(
              body: Center(
                child: CircularProgressIndicator(),
              ),
            )
          : Scaffold(
              appBar: AppBar(
                foregroundColor: Colors.white,
                title: Text(
                  "Mini Project",
                  style: TextStyle(
                  color: Colors.white,
                  fontSize: 22,
                  fontWeight: FontWeight.bold),
                ),
                backgroundColor: const Color.fromARGB(255, 12, 47, 68),
                actions: [
                  status >= 5
                  ? Padding(
                      padding: const EdgeInsets.only(right: 8),
                      child: TextButton.icon(
                        icon:
                            Icon(Icons.history, color: Colors.white), // Icon
                        label: Text(
                          'Review Results',
                          style: TextStyle(color: Colors.white), // Text style
                        ),
                        onPressed: () {
                          Navigator.push(
                              context,
                              MaterialPageRoute(
                                  builder: (context) => ReviewResults(
                                        gid: gid,
                                      )));
                          print('Review Results button pressed');
                        },
                        style: TextButton.styleFrom(
                          side: BorderSide(
                              color: Colors.white,
                              width: 2), // Border color and thickness
                          shape: RoundedRectangleBorder(
                              borderRadius:
                                  BorderRadius.circular(8)), // Rounded border
                          padding: EdgeInsets.symmetric(
                              horizontal: 8, vertical: 8), // Button padding
                        ),
                      ),
                    )
                  : SizedBox()
                ],
          ),
          body: gid == 0
              ? Center(
                  child: Column(
                    mainAxisAlignment: MainAxisAlignment.center,
                    crossAxisAlignment: CrossAxisAlignment.center,
                    children: [
                      Icon(
                        Icons.group_off_sharp,
                        size: 35,
                      ),
                      Text(
                        "Group not created",
                        style: TextStyle(fontSize: 16),
                      ),
                    ],
                  ),
                )
              : Padding(
                  padding: const EdgeInsets.only(left: 16, right: 16),
                  child: Column(
                    children: [
                      SizedBox(
                        height: 10,
                      ),
                      Align(
                        alignment: Alignment.bottomLeft,
                        child: Text(
                          "Project title: ${projectTitle}",
                          style: TextStyle(
                              fontSize: 20, fontWeight: FontWeight.bold),
                        ),
                      ),
                      SizedBox(
                        height: 10,
                      ),
                      Align(
                        alignment: Alignment.bottomLeft,
                        child: Text(
                          "Group member: ${groupmember}",
                          style: TextStyle(
                              fontSize: 20, fontWeight: FontWeight.bold),
                        ),
                      ),
                      SizedBox(
                        height: 10,
                      ),
                      Align(
                        alignment: Alignment.bottomLeft,
                        child: Text(
                          "Project Guide: ${guide}",
                          style: TextStyle(
                              fontSize: 20, fontWeight: FontWeight.bold),
                        ),
                      ),
                      SizedBox(
                        height: 10,
                      ),
                      Align(
                        alignment: Alignment.bottomLeft,
                        child: Text(
                          "Project Center: ${projectCenter}",
                          style: TextStyle(
                              fontSize: 20, fontWeight: FontWeight.bold),
                        ),
                      ),
                      SizedBox(
                        height: 15,
                      ),
                      Row(
                        children: [
                          status == 0
```

```
? Padding(
  padding: const EdgeInsets.all(4),
  child: ElevatedButton(
    style: ElevatedButton.styleFrom(
      backgroundColor: Color(0xFF017AFF),
      padding: EdgeInsets.symmetric(
        horizontal: 20, vertical: 15),
      shape: RoundedRectangleBorder(
        borderRadius:
          BorderRadius.circular(5))),
    onPressed: () {
     Navigator.push(
       context,
       MaterialPageRoute(
         builder: (context) =>
           AbstractForm()));
    },
    child: Row(
     children: [
      Icon(
       Icons.add,
       size: 24,
       color: Colors.white,
      ),
                   Text(
                    'Add abstract',
                    style: TextStyle(
                      color: Colors.white,
                      fontWeight: FontWeight.bold,
                      fontSize: 18),
                   ),
                  ],
                 )),
             )
            : SizedBox(),
          status == 3
            ? Padding(
              padding: const EdgeInsets.all(4),
              child: ElevatedButton(
                style: ElevatedButton.styleFrom(
                  backgroundColor: Color(0xFF017AFF),
                  padding: EdgeInsets.symmetric(
                    horizontal: 20, vertical: 15),
                  shape: RoundedRectangleBorder(
                    borderRadius:
                      BorderRadius.circular(5))),
                onPressed: () async {
                 final result = await Navigator.push(
                    context,
                    MaterialPageRoute(
                      builder: (context) =>
                        ResubmittAbstract()));

                if (result == true) {
                  fetchProject();
                }
              },
              child: Row(
                children: [
                  Icon(
                    Icons.add,
                    size: 24,
                    color: Colors.white,
                  ),
                  Text(
                    'Re-submit abstract',
                    style: TextStyle(
                      color: Colors.white,
                      fontWeight: FontWeight.bold,
                      fontSize: 18),
                  ),
                ],
          )),
        )
      : SizedBox(),
    status == 1
      ? Padding(
          padding: const EdgeInsets.all(4),
          child: ElevatedButton(
            style: ElevatedButton.styleFrom(
              backgroundColor: Colors.green,
              padding: EdgeInsets.symmetric(
                horizontal: 20, vertical: 15),
              shape: RoundedRectangleBorder(
                borderRadius:
                  BorderRadius.circular(5))),
            onPressed: () {
             Navigator.push(
               context,
               MaterialPageRoute(
                 builder: (context) =>
                   ViewAbstractPage()));
            },
            child: Row(
             children: [
              Icon(
                          Icons.remove_red_eye,
                          size: 24,
                          color: Colors.white,
                        ),
                        Text(
                          'View abstract',
                          style: TextStyle(
                              color: Colors.white,
                              fontWeight: FontWeight.bold,
                              fontSize: 18),
                        ),
                      ],
                    )),
               )
             : SizedBox()
        ],
      ),
      SizedBox(
        height: 30,
      ),
      Container(
        height: 100,
        width: 360,
        decoration: BoxDecoration(
    border: Border.all(
      color: getStatusBGColor(status), // Border color
      width: 2, // Border width
    ),
    borderRadius: BorderRadius.circular(8),
  ),
  child: Center(
    child: Text(
      getStatus(status),
      style: TextStyle(
        fontSize: 20,
        fontWeight: FontWeight.bold,
        color: getStatusColor(status),
      ),
    ),
  ),
),
SizedBox(
  height: 30,
),
status == 2
    ? ElevatedButton(
        onPressed: () async {
          addReview1();
```

```
                  print("Click triggered");
                },
                style: ElevatedButton.styleFrom(
                  backgroundColor: Colors.blue, // Button color
                  padding: EdgeInsets.symmetric(
                      horizontal: 20, vertical: 12),
                  shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(8),
                  ),
                ),
                child: Text(
                  "Proceed to Review 1 ",
                  style: TextStyle(
                      fontSize: 16,
                      fontWeight: FontWeight.bold,
                      color: Colors.white),
                ),
              )
            : SizedBox(),
        status == 5
            ? ElevatedButton(
                onPressed: () async {
                  addReview2();
                  print("Click triggered");
                },
                style: ElevatedButton.styleFrom(
                  backgroundColor: Colors.blue, // Button color
                  padding: EdgeInsets.symmetric(
                      horizontal: 20, vertical: 12),
                  shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(8),
                  ),
                ),
                child: Text(
                  "Proceed to Review 2 ",
                  style: TextStyle(
                      fontSize: 16,
                      fontWeight: FontWeight.bold,
                      color: Colors.white),
                ),
              )
            : SizedBox(),
        status == 7
            ? ElevatedButton(
                onPressed: () async {
                  addReview3();
                  print("Click triggered");
                },
                style: ElevatedButton.styleFrom(
                  backgroundColor: Colors.blue, // Button color
                  padding: EdgeInsets.symmetric(
                      horizontal: 20, vertical: 12),
                  shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(8),
                  ),
                ),
                child: Text(
                  "Proceed to Review 3 ",
                  style: TextStyle(
                      fontSize: 16,
                      fontWeight: FontWeight.bold,
                      color: Colors.white),
                ),
              )
            : SizedBox(),
        SizedBox(
          height: 20,
        ),
        status == 9
            ? Column(
                children: [
                  if (isProjectCompleted)
```

```
                    Text(
                      "Project completed successfully!",
                      style: TextStyle(
                        fontSize: 20,
                        fontWeight: FontWeight.bold,
                        color: Colors.green,
                        letterSpacing:
                            1.2, // Slight spacing for readability
                      ),
                    ),
                  // Other widgets...
                ],
              )
            : SizedBox()
          ],
        ),
      ),
    );
  }

  String getStatus(int st) {
    switch (st) {
      case 0:
        return "Submit your abstract";
      case 1:

        return "Abstract verification pending";

      case 2:

        return "Abstract approved";

      case 3:

        return "Abstarct rejected";

      case 4:

        return "1st review verification pending";

      case 5:

        return "1st review finished";

      case 6:

        return "2st review verification pending";

      case 7:

        return "2st review finished";

      case 8:

        return "3rd review verification pending";

      case 9:

        return "3rd review finished";

      case 10:

        return "Re-submit after updation";

      default:

        return "Something went wrong";

    }

  }
```

```
  Color getStatusColor(int status) {
    switch (status) {
      case 0:
        return Colors.orange;
      case 1:
        return Colors.amber;
      case 2:
        return Colors.green;
      case 3:
        return Colors.red;
      case 4:
        return Colors.amber;
      case 5:
        return Colors.green;
      case 6:
        return Colors.amber;
      case 7:
        return Colors.green;
      case 8:
        return Colors.amber;
      case 9:
        return Colors.green;
      case 10:
        return Colors.blueAccent;
      default:
        return Colors.grey;
    }
  }

  Color getStatusBGColor(int status) {
    switch (status) {
      case 0:
        return Colors.orange;
      case 1:
        return Colors.amber;
      case 2:
        return Colors.green;
      case 3:
        return Colors.red;
      case 4:
        return Colors.amber;
      case 5:
        return Colors.green;
      case 6:
        return Colors.amber;
      case 7:
        return Colors.green;
      case 8:

        return Colors.amber;

      case 9:

        return Colors.green;

      case 10:

        return Colors.blueAccent;

      default:

        return Colors.grey;

    }

  }

}
```

AbstarctForm.dart

```
import 'dart:io';

import 'package:flutter/material.dart';

import 'package:supabase_flutter/supabase_flutter.dart';

import 'package:file_picker/file_picker.dart';



class AbstractForm extends StatefulWidget {

  const AbstractForm({super.key});



  @override

  State<AbstractForm> createState() => _AbstractFormState();

}
```

```
class _AbstractFormState extends State<AbstractForm> {

  final TextEditingController _projectTitleController = TextEditingController();

  final TextEditingController _projectCenterController =

      TextEditingController();

  final supabase = Supabase.instance.client;



  File? selectedFile; // Stores the selected file

  String? fileName; // Stores the file name

  String? fileUrl; // Stores the uploaded file URL;



  Future<void> storeAbstract() async {

    try {

      String? sid = supabase.auth.currentUser?.id;



      final group = await supabase

          .from('tbl_groupmember')

          .select()

          .eq('student_id', sid!)

          .maybeSingle()

          .limit(1);

      String? url = await uploadFile();

      String projectTitle = _projectTitleController.text;

      String projectCenter = _projectCenterController.text;

      await supabase.from('tbl_group').update({
```

```
      'group_abstract': url,

      'project_title': projectTitle,

      'group_center': projectCenter,

      'group_status': 1,

    }).eq('group_id', group!['group_id']);

    ScaffoldMessenger.of(context).showSnackBar(

      SnackBar(

        content: Text(

          "Abstract Submitted:",

          style: TextStyle(

            color: Colors.white, // Text color

            fontSize: 16,

            fontWeight: FontWeight.bold,

          ),

        ),

        backgroundColor: Colors.green, // Background color

        behavior: SnackBarBehavior.floating, // Floating style

        shape: RoundedRectangleBorder(

          borderRadius: BorderRadius.circular(10), // Rounded corners

        ),

        margin: EdgeInsets.all(20), // Add margin around Snackbar

        elevation: 6, // Add shadow effect

        duration: Duration(seconds: 5), // Display duration

      ),
    );

    Navigator.pop(context);

  } catch (e) {

    print("Error submitting abstract:$e");

  }

}


Future<void> pickFile() async {

  try {

    FilePickerResult? result = await FilePicker.platform.pickFiles(

      type: FileType.custom,

      allowedExtensions: ['pdf', 'doc', 'docx'], // Accept PDF & Word files

    );


    if (result != null) {

      setState(() {

        selectedFile = File(result.files.single.path!);

        fileName = result.files.single.name; // Store selected file name

      });


      ScaffoldMessenger.of(context).showSnackBar(

        SnackBar(content: Text("File Selected: $fileName")),

      );

    }
  } catch (e) {
    print("Error selecting file: $e");
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text("File selection failed!")),
    );
  }
}

Future<String?> uploadFile() async {
  if (selectedFile == null) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text("Please select a file first!")),
    );
    return null;
  }

  try {
    String uploadFileName =
        "abstracts/${DateTime.now().millisecondsSinceEpoch}.${fileName!.split('.').last}";

    // Upload file to Supabase storage
    await supabase.storage
        .from('Abstract')
        .upload(uploadFileName, selectedFile!);
    // Get the public URL of the uploaded file
    fileUrl = supabase.storage.from('Abstract').getPublicUrl(uploadFileName);


    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text("File Uploaded Successfully!")),
    );


    print("Uploaded File URL: $fileUrl");


    return fileUrl;
  } catch (e) {
    print("Error uploading file: $e");
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text("Upload failed!")),
    );
    return null;
  }
}


@override
Widget build(BuildContext context) {
  return Scaffold(
    body: Center(
child: Padding(
  padding: const EdgeInsets.all(20.0),
  child: Container(
    width: 500,
    height: 500,
    decoration: BoxDecoration(
        border: Border.all(color: Colors.black, width: 1.5),
        borderRadius: BorderRadius.circular(10)),
    child: Column(
      children: [
        SizedBox(height: 40),
        Text(
          "Abstract Form",
          style: TextStyle(fontSize: 25, fontWeight: FontWeight.bold),
        ),
        Padding(
          padding: const EdgeInsets.all(20.0),
          child: TextFormField(
            controller: _projectTitleController,
            decoration: InputDecoration(
                hintText: 'Enter project title',
                border: OutlineInputBorder(),
                prefixIcon: Icon(Icons.computer)),
          ),
        ),
        Padding(
          padding:
              const EdgeInsets.only(left: 20, right: 20, bottom: 20),
          child: TextFormField(
            controller: _projectCenterController,
            decoration: InputDecoration(
                hintText: 'Enter project center',
                border: OutlineInputBorder(),
                prefixIcon: Icon(Icons.hub)),
          ),
        ),
        ElevatedButton.icon(
          onPressed: pickFile, // Pick File Function
          icon: Icon(Icons.attach_file, color: Colors.white, size: 24),
          label: Text(
            "Select File",
            style: TextStyle(
                fontSize: 16,
                fontWeight: FontWeight.bold,
                color: Colors.white),
          ),
          style: ElevatedButton.styleFrom(
            backgroundColor: Color(0xFF004A61),
            padding: EdgeInsets.symmetric(horizontal: 20, vertical: 12),
            shape: RoundedRectangleBorder(
                borderRadius: BorderRadius.circular(10)),
            elevation: 5,
          ),
        ),
        if (fileName != null)
          Padding(
            padding: const EdgeInsets.all(10.0),
            child: Text(
              "Selected File: $fileName",
              style: TextStyle(color: Colors.green, fontSize: 14),
              textAlign: TextAlign.center,
            ),
          ),
        Padding(
          padding: const EdgeInsets.only(left: 80, right: 80, top: 25),
          child: ElevatedButton(
            style: ElevatedButton.styleFrom(
              backgroundColor: Color(0xFF017AFF),
              padding:
                  EdgeInsets.symmetric(horizontal: 30, vertical: 15),
              shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(5)),
                  ),
                  onPressed: storeAbstract, // Uploads the file when clicked
                  child: Text(
                    'Submit',
                    style: TextStyle(
                        color: Colors.white,
                        fontSize: 18,
                        fontWeight: FontWeight.bold),
                  ),
                ),
              ),
              SizedBox(height: 50),
            ],
          ),
        ),
      ),
    ),
  );
 }
}
```