RICHFIELD

PROJECT DOCUMENTATION (PHASE 1 TO 5)

Name & Surname:	**ICAS/ITS No:**
Aqeelah Rahman	402100588
Cameron Govender	402100163
Shekhar Maharaj	402101963
Allistair Dean Moodley	402101356
Naazia Ephraim	402101033

Qualification: BSc IT **Semester:** 2 **Module Name**: IT PROJECT 700

Date submitted: 2 December 2023

Table of Contents

1. Introduction

We understand the crucial importance of having a solid and effective student management platform in a time when education is undergoing continuous changes. As a solution to this urgent need, we put forth the ambitious task of optimising the current student management software, iEnabler. This effort intends to use student feedback to create a user-centric, seamless system, thereby improving the educational experience for both students and instructors.

It is essential to emphasise that this revamp is more than just an IT project; it represents a strategic investment in the institution's future. We hope to reduce administrative procedures and increase our capacity for competition in the rapidly changing environment of higher education by modernising this crucial system.

1.1 Objectives

The primary objective of the iEnabler frontend enhancement project is to elevate the user interface (UI) and user experience (UX) of the student management program. To achieve this, the project encompasses a multi-faceted approach. Initial phases involve comprehensive user research and analysis to discern user needs and preferences, yielding user personas and targeted requirements.

Subsequent usability testing aims to evaluate both current and proposed designs, generating valuable feedback for improvement. Accessibility standards are prioritized to ensure inclusivity, and a commitment to responsive design ensures a seamless user experience across diverse devices. Streamlining navigation and establishing a consistent design language contribute to an overall improved user flow and visual coherence. Performance optimization measures, including responsive design, seek to address any delays in user interactions. The project also focuses on creating user documentation and training materials, as well as implementing a feedback mechanism within the application.

Security considerations and scalability are paramount, ensuring that changes do not compromise data security and that the frontend is designed to accommodate future growth. Clear stakeholder communication and a comprehensive onboarding process for users further contribute to project success. Post-implementation, an evaluation phase is planned to assess the effectiveness of the enhancements, utilizing user and stakeholder feedback for continuous refinement.

Expanding on the objectives:

Project Area	**Specific**	**Measurable**	**Achievable**	**Relevant**	**Time-bound**
Modern Technologies	Implement ReactJS to create interactive and dynamic components for the application.	Achieve a user engagement increase of at least 20% based on user feedback and analytics.	We have the necessary expertise and resources to implement ReactJS effectively.	Enhancing the user experience aligns with our project's objectives.	Complete the ReactJS implementation and measure results within the next 3 months.

CI/CD Pipelines	Establish CI/CD pipelines for automating testing and deployment processes.	Achieve a 10% reduction in deployment time and a 10% decrease in post-release issues.	We have the required tools and infrastructure for CI/CD implementation.	Ensuring timely updates and code reliability is crucial for project success.	Have CI/CD pipelines fully operational within the next 3 months and measure improvements over the following months.
Containerization	Utilize Docker for containerization to ensure consistent deployment and reduce compatibility issues.	Decrease deployment related issues by 10% and improve deployment consistency.	We have experience with Docker and the infrastructure to support it.	Ensuring secure and consistent deployment aligns with project goals.	Implement Docker for containerization within the next 3 months and assess its impact on deployments over the next 3 months.
User-Centric Design	Develop a user-centric and intuitive interface for accessing financial documents and academic records.	Achieve a user satisfaction rating of at least 85% based on user testing and feedback.	We have a design team with expertise in user-centric design principles.	Enhancing user accessibility and understanding is a key project objective.	Complete the user-centric design implementation and gather user feedback within the next 3 months.
Version Control	Implement Git for version control to enable collaborative development and efficient code management.	Reduce code conflicts by 20% and improve code traceability.	We have the necessary training and infrastructure to implement Git effectively.	Efficient code management and collaboration are essential for project success.	Implement Git for version control within the next month and evaluate its impact on code management within 3 months.

1.2 Hypotheses

Our research suggests two hypotheses:

Simple Hypothesis 1 (H1)

Improved transparency, convenience, and engagement will lead to improved academic performance and general satisfaction.
This is made possible by implementing an improved iEnabler system that gives students access to their grades, financial information, and course details. This version of iEnabler will do away with manual inquiries and waiting periods by giving students quick access to their grades, financial information, and course information in real-time. This increased openness will enable students to keep track of their development and financial situation constantly, encouraging a proactive approach to their education.

Simple Hypothesis 2 (H2)

Implementation of this version of iEnabler system, providing students access to their grades, financial information, and course details, will result in improved academic performance and general satisfaction.
The revamped iEnabler will simplify administrative procedures and lighten the workload for staff and students. From any location with internet access, students can check their grades and class schedules, pay their tuition, and access information about financial aid. This ease will promote more effective interactions, saving both staff and students time and effort. Students are likely to become more engaged and responsible for their studies if they have immediate access to information about their finances and academic performance. This iEnabler will include functions like progress tracking and reminders for significant dates. Human errors will be much less likely if procedures like grade calculation, fee management, and course registration are automated. This accuracy will stop unneeded problems from arising from data entry errors, making things easier for both staff and students.
The theory contends that implementing a more user-friendly iEnabler platform with capabilities like grade checking, access to financial information, and course information will boost openness, convenience, engagement, and overall academic success. Improvements in communication, a decrease in administrative mistakes, and the possibility of a more individualized learning environment are all anticipated advantages. Metrics like increased student use of this version of iEnabler, improved academic performance, and favourable feedback from students, faculty, and administrative staff can all be used to gauge the success of these hypotheses.

1.3 Justification

There are several compelling reasons to start our iEnabler system's overhaul, with a particular emphasis on rebuilding its existing frontend. We can effectively fix the problems currently troubling users by using this strategy. Students will be able to easily access their academic and financial data thanks to the redesigned design and improved functionality of the iEnabler system, which will also dramatically raise student involvement. The redesign will significantly reduce user annoyance, which has been a recurrent issue. By making these upgrades, we hope

to strengthen Richfield's reputation for offering efficient, useful, and user-friendly services and establish ourselves as a leader in contemporary educational technology.
Modern web development methods will also have other advantages in the process of revamping the website. The iEnabler platform will be simpler to manage as a result, which will lower the possibility of usability and functionality problems in the future. In addition, it will make expansion easier if the need arises in the future, perfectly matching with our dedication to scalability. This all-encompassing approach to redesigning iEnabler not only takes on the problems that are already there, but also makes sure that our educational technology is flexible and keeps up with the changing demands of our organization and its users.

1.4 Expectations

- An aesthetically pleasing and easy frontend interface for the revamped iEnabler tool.
- An improved user experience which will hopefully result in a higher level of user engagement & satisfaction.
- An improvement in accessibility and responsiveness across various devices and screen sizes.
- Version control, CI/CD, and containerization procedures that have simplified the development and deployment processes.
- Collaboration and effective communication between project stakeholders.

1.5 Conclusion

By undertaking this comprehensive redesign project, we aim to transform the iEnabler tool into a modern and efficient platform that meets the needs of today's students. The adoption of ReactJS, Git, CI/CD pipelines, containerized environments, and an agile workflow will ensure a successful transition to a more user-friendly and functional frontend. This endeavour will not only address the current challenges but also lay the foundation for future enhancements and improvements.

2. Planning Phase

2.1 Identification of Need

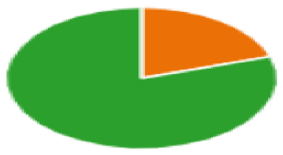
The identification of the need for the iEnabler System Revamp project arises from the following factors:

User Dissatisfaction:

- Users of the current iEnabler system have expressed significant dissatisfaction with its usability, navigation, and overall user experience.

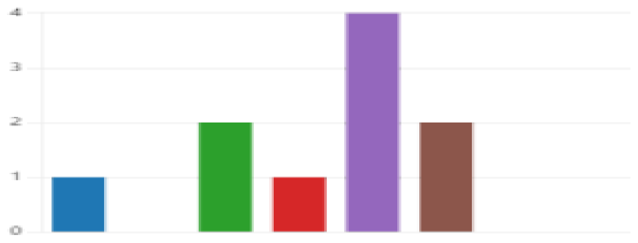
1. How frequently do you use iEnabler?

🔵	Never	0
🟠	Weekly	1
🟢	Monthly	4
🔴	Daily	0



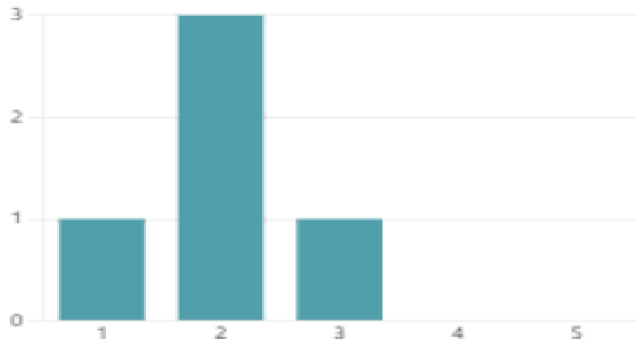
2. Which feature(s) do you use most often?

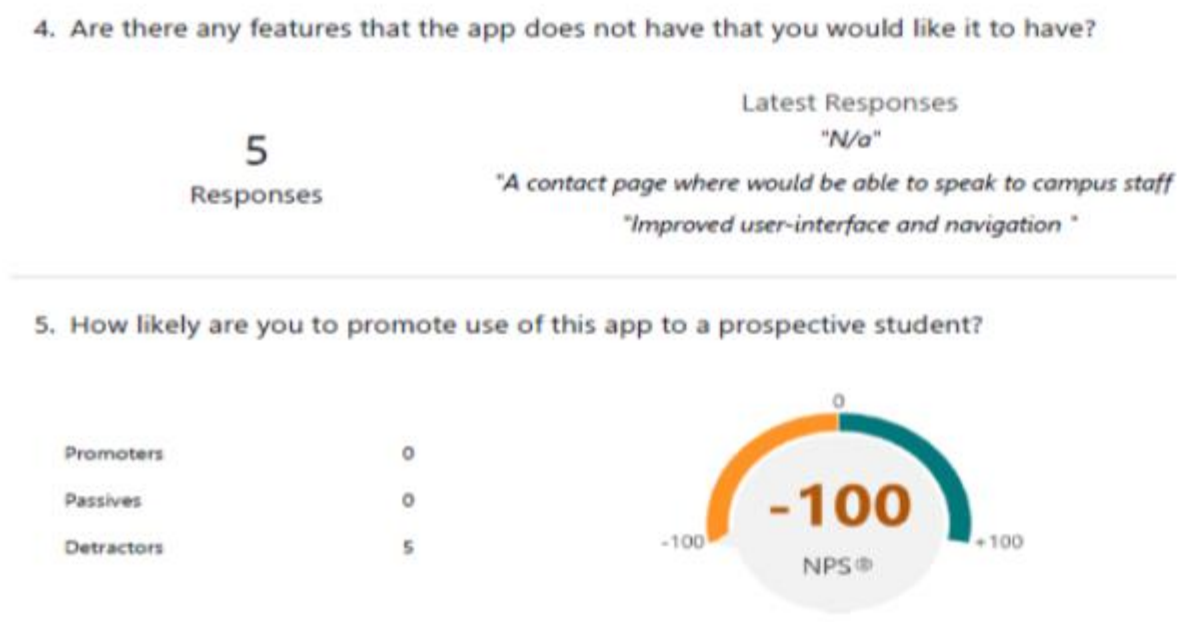
🔵	Application	1
🟠	Registration	0
🟢	Student Administration	2
🔴	Student Enquiry	1
🟣	Student Finance	4
🟤	Student Final Results	2
●	PayGate Online Payment	0
●	Higher Degrees	0



3. How user-friendly is the iEnabler app? (1 Star meaning extremely difficult to use)

2.00
Average Rating





4. Are there any features that the app does not have that you would like to have?

5
Responses

Latest Responses
"N/a"
"A contact page where would be able to speak to campus staff
"Improved user-interface and navigation "

5. How likely are you to promote use of this app to a prospective student?

Promoters	0
Passives	0
Detractors	5

Operational Inefficiencies:

- The existing system suffers from data entry errors, lack of personalization, and scalability issues, which hinder operational efficiency.

Strategic Importance:

- Modernizing the student management platform aligns with the institution's strategic goals of staying competitive in the rapidly evolving higher education landscape.

(Exner, 2023)

2.2 Preliminary Investigation

The preliminary investigation involves an initial assessment of the project's feasibility and potential impact.

2.3 Feasibility Study

Technical Feasibility

Assessment:

- The project aims to utilize modern web development technologies such as ReactJS, CI/CD pipelines, containerization, and version control, which are technically feasible and readily available.

Expected Outcomes:

- The technical feasibility study indicates that the project can be successfully implemented with the available tools and technologies.

Operational Feasibility

Assessment:

- The project will significantly improve operational efficiency by reducing manual processes, errors, and enhancing user engagement.

Expected Outcomes:

- The operational feasibility study indicates that the project aligns with the institution's goals of streamlining operations.

Economic Feasibility

Assessment:

- The project will incur initial development costs but is expected to result in long-term cost savings due to improved efficiency.

Expected Outcomes:

- The economic feasibility study shows that the benefits of the project outweigh the initial investment.

2.4 Project Objectives

The primary objectives of the iEnabler System Revamp project are as follows:

- To modernize the iEnabler frontend to meet contemporary web standards.
- To improve user experience, accessibility, and user engagement.
- To reduce operational inefficiencies and enhance system scalability.

2.5 Project Scope

The project scope encompasses the redesign of the iEnabler frontend, including the implementation of ReactJS, CI/CD pipelines, containerization, and version control.

2.6 Project Stakeholders
Project team:

- Project Manager: [Shekhar]
- Development Team:
- Allistair
- Aqeelah
- Cameron
- Naazia
- Shekhar

Users:

- Students
- Faculty
- Administrative Staff

2.7 Project Constraints
Time Constraint:

- The project must adhere to the specified timeline, with a final due date of August 28, 2023.

Technology Constraint:

- The project will utilize the specified modern technologies, IE Git, React, CSS, etc

2.8 Risk Assessment
Technical Feasibility Risks:

Risk	**Impact**	**Mitigation**
Technical Feasibility	Delays in technology selection and potential rework	Conduct thorough technology evaluations before committing. Have contingency plans for alternative technologies if needed.
Resource Availability	Resource shortages can lead to project delays and decreased productivity	Cross-train team members to cover for each other. Maintain a contingency plan for resource allocation.
Scope Creep	Scope creep can lead to project delays and increased development efforts	Clearly define the project scope and have a formal change control process in place to manage scope changes.

Inadequate Requirements Gathering	Delays in the project schedule and increased development efforts	Engage stakeholders actively in requirements gathering, use techniques like workshops, and conduct thorough reviews.
Lack of Change Management	User adoption and satisfaction could suffer, affecting project success	Implement a change management plan to prepare users for the system changes, provide training, and address concerns.
Lack of Documentation	Difficulties in project management, handover, and future maintenance	Maintain comprehensive project documentation and ensure knowledge transfer within the team.
Security and Privacy Concerns	Legal and reputational damage, project delays	Include security and privacy assessments in the planning phase and implement robust security measures.

A risk register will be maintained to effectively manage new and existing risks.

2.9 Project Scheduling

Key	**Value**
#1	Software Requirement Specification (SRS): - Gather detailed functional and non-functional requirements for the revamped iEnabler system. - Document user stories and use cases.
#2	Data Modelling: - Design the database schema and data models for the new system. - Create entity-relationship diagrams.
#3	Technology Setup: - Set up development environments and tools. - Implement containerization using Docker.
#4	User Interface Design: - Design the user-centric and intuitive frontend interface.

	- Create wireframes and prototypes.
#5	Development: - Write code for the new iEnabler frontend using ReactJS. - Implement version control with Git.
#6	Testing: - Perform unit testing, integration testing, and user acceptance testing. - Identify and resolve defects.
#7	User Training and Documentation: - Develop user training materials. - Conduct training sessions for staff and users. - Create user documentation for the new system.
#8	Deployment: - Deploy the revamped iEnabler system to production. - Monitor system performance and stability.
#9	Quality Assurance and Review: - Conduct a quality assurance review to ensure project goals are met. - Gather feedback from users and stakeholders.
#10	Project Closure: - Conduct a final project review. - Document lessons learned. - Close out the project and hand over deliverables.

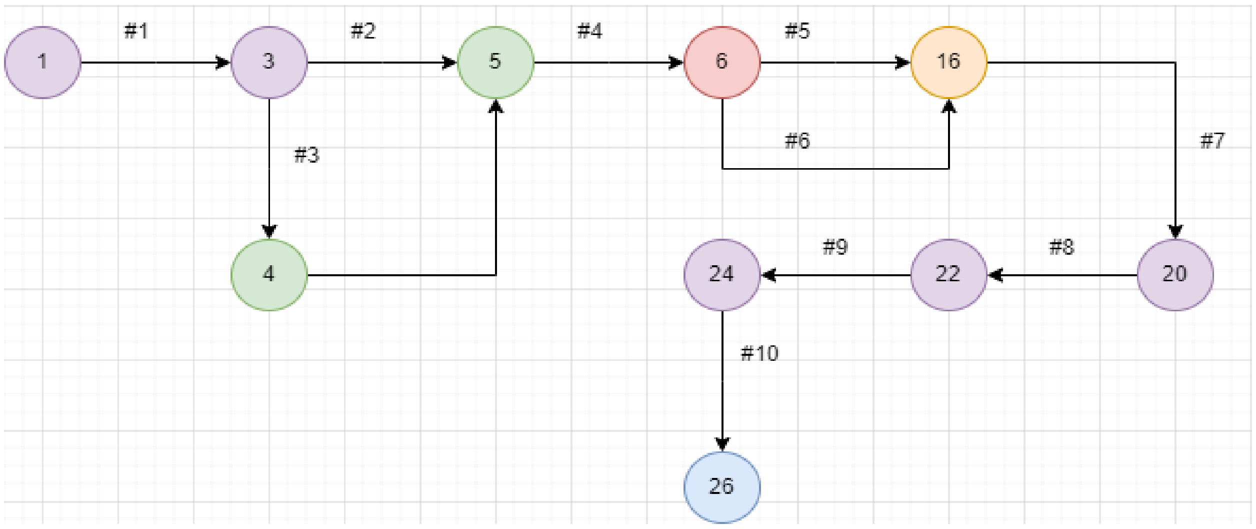


Figure 1 PERT Chart

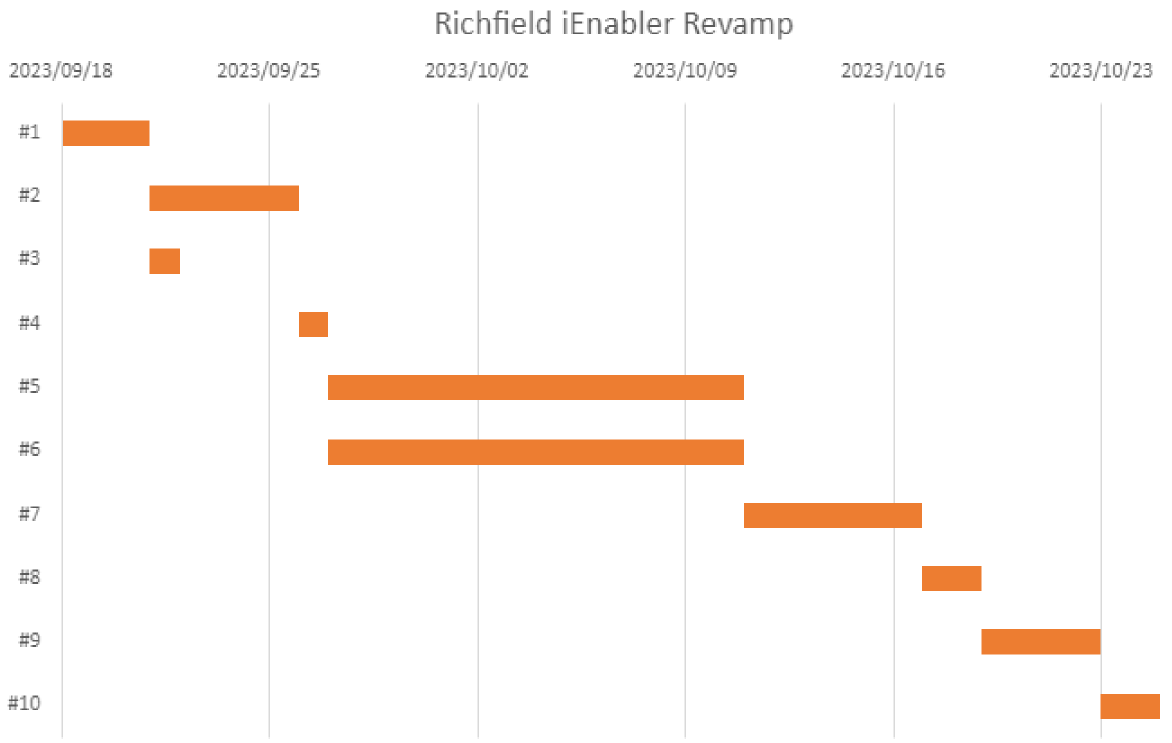


Figure 2: Gantt Chart

2.10 Software Requirement Specification

2.10.1 Introduction

Purpose

The purpose of this SRS document is to define the functional and non-functional requirements of the iEnabler System Revamp project.

Scope

This document outlines the requirements for the revamped iEnabler student management system, focusing on the planning phase. It includes details on user requirements, system capabilities, and constraints.

2.10.2 System Overview

System Description

The iEnabler System Revamp aims to modernize the frontend of the existing student management platform to improve user experience, accessibility, and functionality. The system will use existing data models and schemas and will only display data.

2.10.3 System Features

User-Centric Interface: The system should provide an intuitive and user-centric interface for students, making it easy to access academic and financial information.

2.10.4 Functional Requirements

User Authentication

- Users should be able to log in with their credentials.
- Authentication mechanisms should be in place to secure user data.

User Dashboard

Users should have personalized dashboards displaying relevant information such as grades, schedules, and financial data.

Access to Academic Data

- Students should be able to access their course information, academic records, and grades in real-time.
- Faculty and administrative staff should have access to student records and academic data.

Access to Financial Data

- Students should be able to view their financial statements, pay tuition fees, and access financial aid information.
- Administrative staff should have access to financial data for record-keeping.

2.10.5 Non-Functional Requirements

Performance - The system should provide high performance and responsiveness, even during peak usage times.

Security - Security measures should be in place to protect user data and prevent unauthorized access.

Usability- The user interface should be intuitive and easy to navigate, promoting a positive user experience.

Reliability - The system should be reliable, with minimal downtime for maintenance.

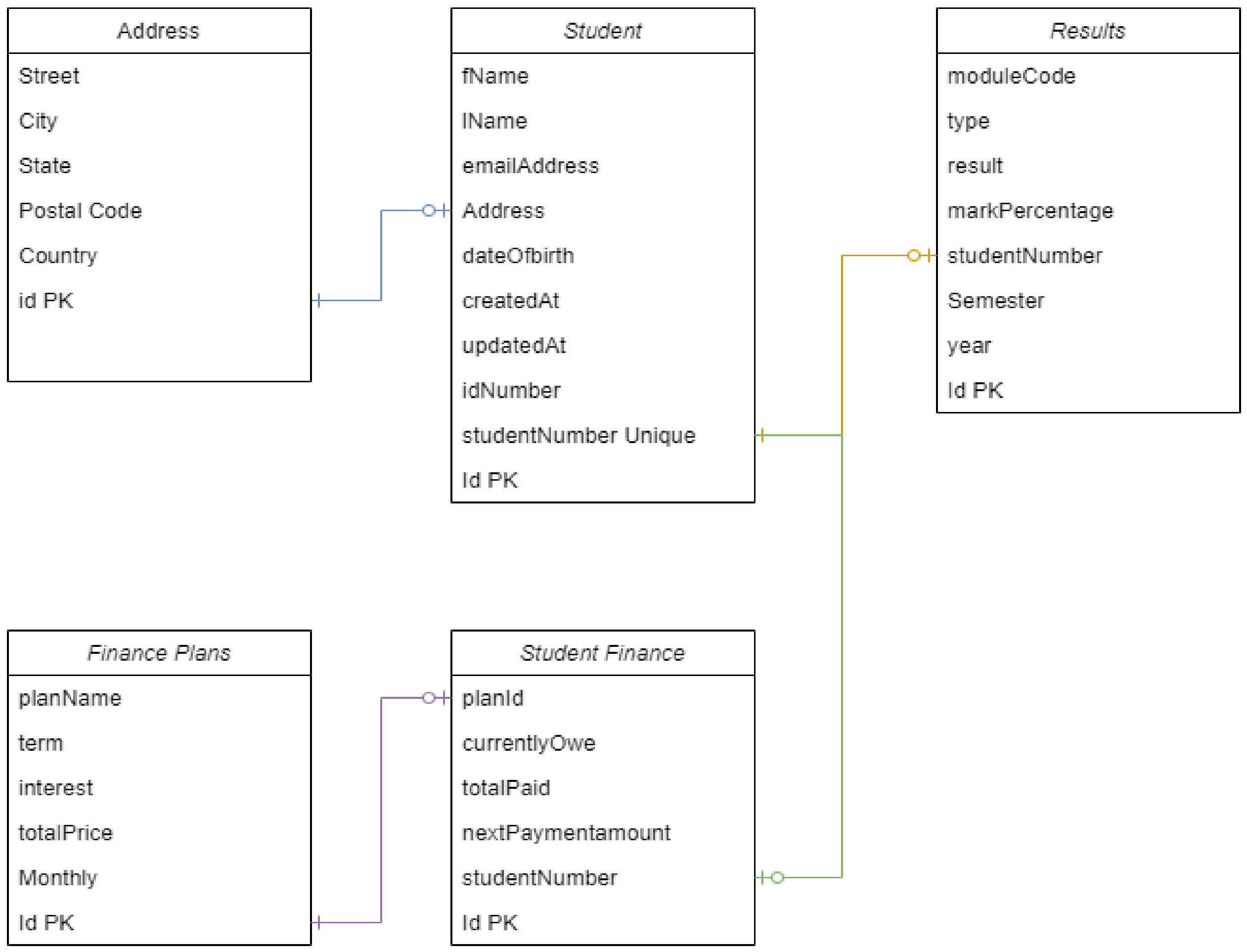
2.10.6 Constraints

- The project should adhere to the specified timeline with a final due date of August 28, 2023.
- The project will use the specified technologies and tools
    - ReactJs
    - Git
    - Html5/Css3
    - NodeJs (For building and initialization)
    - Supabase
        - This database will be used for testing purposes only. It is a Postgres database. It was chosen to allow us to easily simulate a real working environment where the original backend code would be used
    - Existing API's and Schemas created by previous developers

2.11 Data Models

For the purposes of testing this web application and ensuring that frontend data retrieval and processing works efficiently, we would need to create a mock schema/database to simulate how the retrieval process would work on the frontend. The schema that we create will be simplistic in nature but will allow us to fetch necessary data.

Richfield student database simulation:


Address
Street
City
State
Postal Code
Country
id PK
Student
fName
lName
emailAddress
Address
dateOfbirth
createdAt
updatedAt
idNumber
studentNumber Unique
Id PK
Results
moduleCode
type
result
markPercentage
studentNumber
Semester
year
Id PK
Finance Plans
planName
term
interest
totalPrice
Monthly
Id PK
Student Finance
planId
currentlyOwe
totalPaid
nextPaymentamount
studentNumber
Id PK

3. Analysis Phase

3.1 Introduction

This section outlines the analysis phase of the Software Development Life Cycle (SDLC) for the redesign of the frontend of the iEnabler tool at Richfield Graduate Institute of Technology. The purpose of this phase is to thoroughly understand the current system, gather relevant information, identify constraints and weaknesses, and define the functional and non-functional requirements for the proposed system.

3.2 Information Gathering Methodology

Observation

- As students, we (This team) utilize the iEnabler system, and have recognised that the system has some flaws particularly on the user interface. Even though the functionality of the system works as needed, it is important to provide an easy-to-use interface that provides users with a joyful experience on the site. This will show students, parents of students and staff the standard of work that Richfield offers.

Participatory

- Engage stakeholders, including students and staff, in hands-on sessions to interact with the current frontend.
- Encourage participants to perform common tasks and note their feedback regarding usability, responsiveness, and overall experience.

Interviews

- Conduct one-on-one or group interviews with stakeholders, including students, faculty, and administrators.
- Focus on understanding their needs, challenges with the existing frontend, and expectations from the redesigned platform.

3.3 Data Analysis

Data Integrity

- Ensure the accuracy, completeness, and reliability of the data collected by validating sources and cross-referencing information with other team members.

Constraints

- Time Constraint:
  - The project must adhere to the specified timeline, with a final due date of August 28, 2023.
- Technology Constraint:
  - The project will utilize the specified modern technologies, IE Git, React, CSS, etc

3.4 Weakness of the current system

- Outdated Interface:
    - The current interface is outdated and not aligned with modern web standards, leading to a suboptimal user experience.
- Limited Responsiveness:
    - The frontend is not sufficiently responsive across various devices and screen sizes, hindering accessibility.
- Complex Navigation:
    - Navigation within the tool is convoluted and can be confusing for users, affecting usability and efficiency.
    - Accessibility: If the system lacks accessibility features, it needs to be made more inclusive.

3.5 Functional Requirements

User Authentication and Authorization:

- Implement secure login mechanisms for users (students, faculty, administrators).
- This feature is functional on the current site but needs to be heavily re-styled and additional features such as a forgot password (mentioned below).

Themed Dashboard:

- Create a user-friendly dashboard for easy access to essential features and information relevant to each user that sticks to the colours and fonts of Richfield.

Responsive Layout:

- The dashboard should be presentable on all platforms. To achieve this, a mobile-first design approach will be implemented

Forgot password - Direct Email to Student Advisor:

- As a student, we have all forgotten our Moodle, or iEnabler, passwords so many times and each time, we must formulate an email to request a password reset. We will add a ‘Forgot Password’ button to the login page that formats an email template to request a new password using the user's default email service. Part of the forgot password process will include:
- Get the students student-number
- Get the students student-advisor (drop-down of names, with images)
- Open the student default email provider with the formatted template, which will be ready to send.

3.6 Non-Functional Requirements

Performance:

- Ensure the platform's responsiveness and loading times meet modern web performance standards.

Scalability:

- Design the frontend to handle a growing number of users and data without compromising performance.

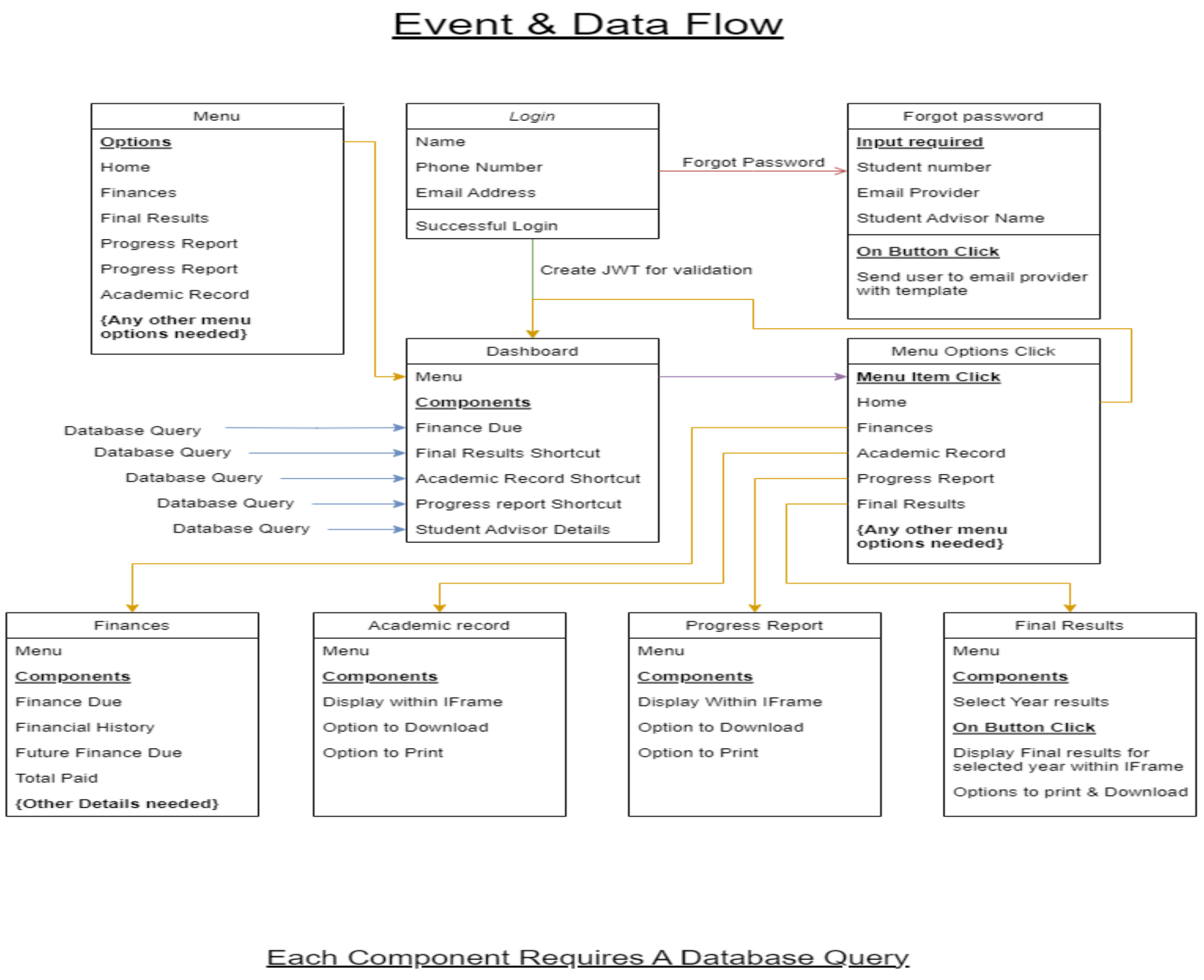
Security:

- Implement robust security measures to protect user data and ensure compliance with data privacy regulations.

Usability:

- Design an intuitive and easy-to-use interface that requires minimal training for users.

3.7 Data Modelling for the proposed system



4. System Design

4.1 Introduction

In response to the evolving needs of students and the growing demand for user-friendly and modern web interfaces, this section outlines the proposed system design for the comprehensive redesign of the frontend of the iEnabler tool used by Richfield Graduate Institute of Technology in Durban, South Africa. The redesign aims to enhance the user experience, usability, accessibility, and overall functionality of the platform by adopting modern web development technologies and methodologies.

4.2 System Design (Description of Proposed System)

The proposed system involves a complete revamp of the existing frontend of the iEnabler tool. The redesign will focus on user-centric design, utilizing modern web technologies like ReactJS to create a dynamic and interactive interface. Key elements of the redesign include implementing Git version control, establishing CI/CD pipelines (Through AWS Amplify), adopting an agile workflow for iterative development and frequent feedback loops.

4.3 Design

4.3.1 Software Architectural Design

The software architecture will follow a modular and component-based structure, leveraging ReactJS to create interactive components for the frontend. The architecture will encompass the Model-View-Controller (MVC) pattern to separate concerns and improve maintainability.

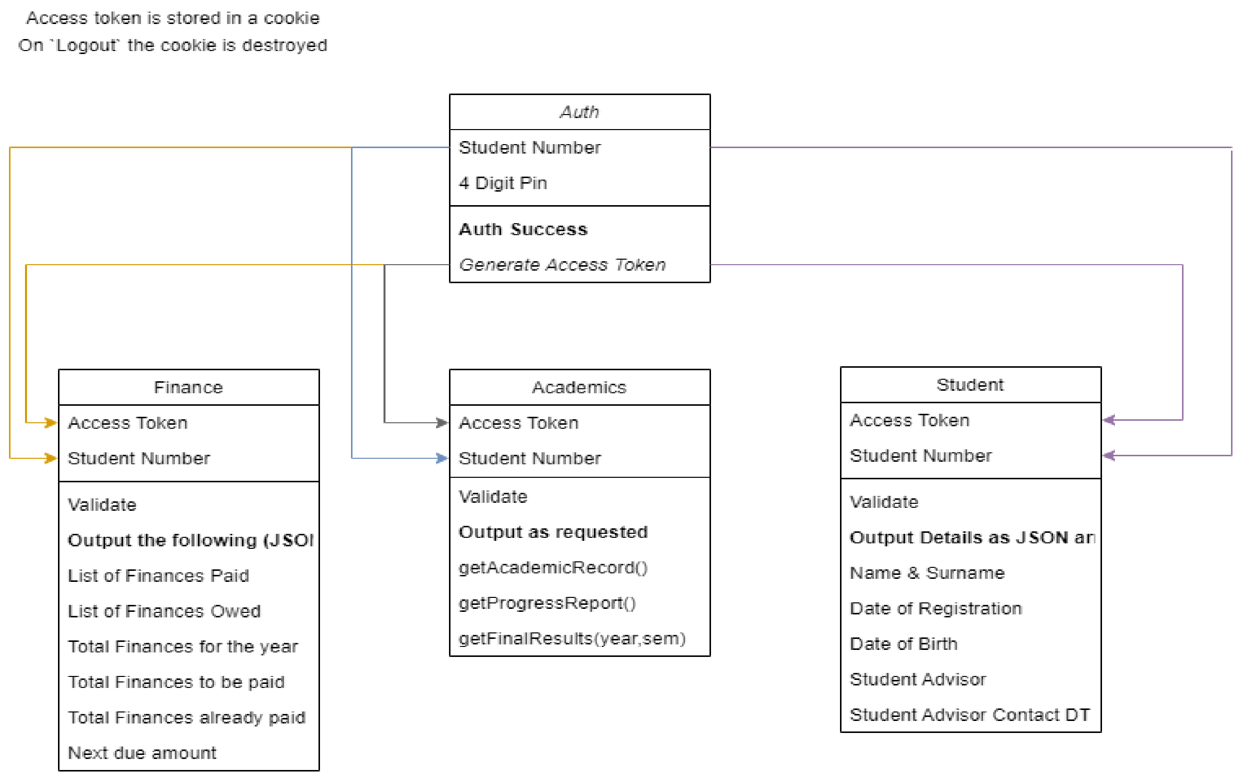
4.3.2 Hardware Architectural Design

The system being a web application, the hardware architecture primarily relies on the server infrastructure to host and serve the frontend. Cloud-based servers will be employed to ensure scalability and reliability.

4.3.3 Network Architectural Design

The system will utilize a simplistic network architecture, via a cloud provider platform. Due to this project being a student project and not a full application to be deployed, we (as a team) believe that a free hosting option would be ideal. Amazon Web Services offers a 12 month free period for their AWS Amplify sub-platform. AWS Amplify allows its users to seamlessly deploy React applications on a pre-built CI/CD pipeline complete with its own containerization. As the development team, we need only set up the GitHub repository and the appropriate branches for development and for production.

4.3.4 Class Diagram



4.4 Physical Design

AWS Amplify sets up all load-balancing, development pipelines and server configurations. We would only need to purchase a domain name if we do not wish to use the generated domain.

4.5 Database Design

We will be using Supabase for data storage. Supabase is a Postgres, Firebase alternative. Supabase is much cheaper than Firebase, while still maintaining some of its core security features, authentication and ease of use. Supabase has an officially supported JavaScript (JS) Api to control all aspects of the database; including but not limited to:

- CRUD operations
- Authentication
- Forgot password
- Row Level Security (RLS) control
- Live Search

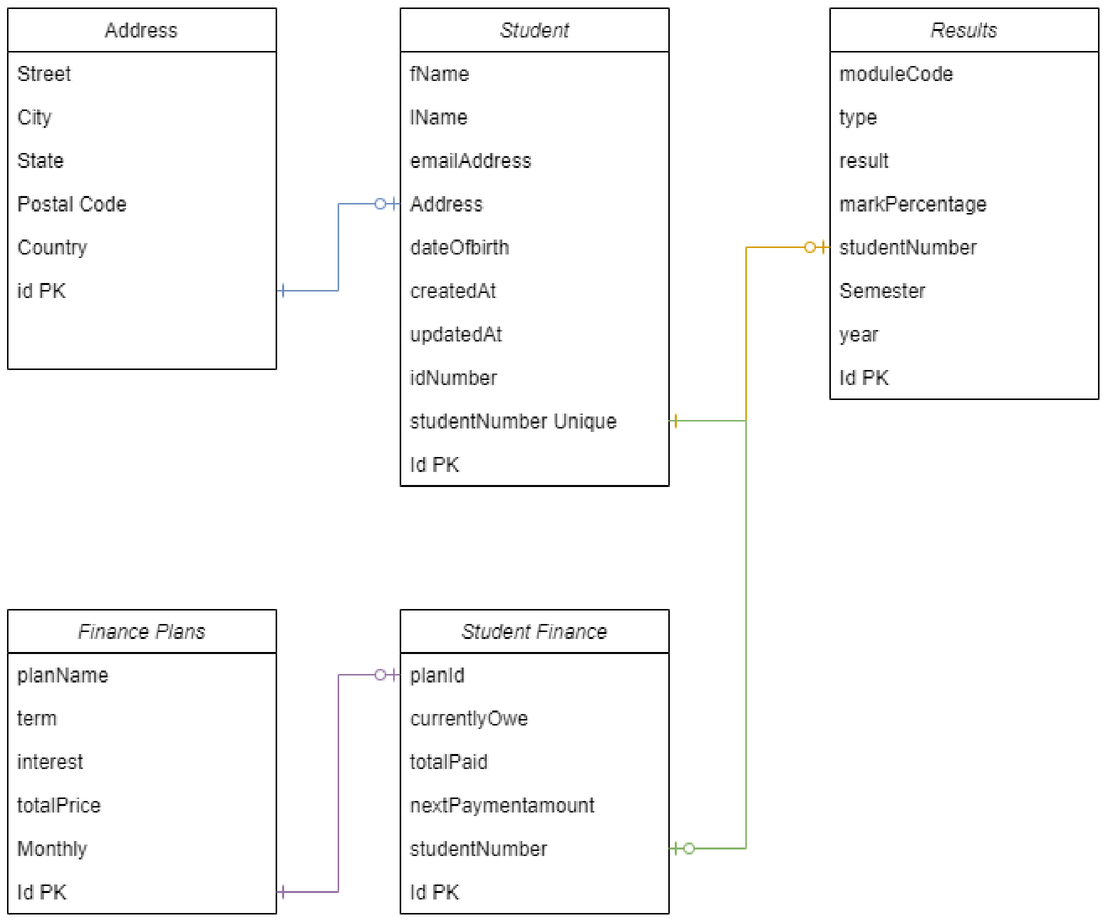


Figure 3: Database design

4.6 Program Design (Pseudo Code)
Server

```
.env
REACT_SUPABASE_URL = "https://www.supabase.com/my/project/path/"
REACT_SUPABASE_ANON_KEY = "sdblb9y0fvblhv7807807-8bjdvfzllbhds(&vt83rtvgkcvx dfgd"
```

```
database.js
// Create database connection
Import supabase_api;

Class database{
        constructor(url,key){
                Return SupaBaseConnection(url,key);
}

destroy(SupaBaseConnection){
        SupaBaseConnection.destroy();
}
}
```

```
Auth.js
Import database.js

Class auth{
        Function login(studentNumber, Pin){
                Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY  )
If SupaBaseConn
Then setCookie("Token",SupaBaseConn.authorise(studentNumber+"@my.richfield.ac.za",
Pin))
endif;
}
Function logout(){
getCookie("Token").destroy()
}
}
```

```
Academics.js
Import database.js

Class Academics {
        constructor(token){
                validateToken()}

Function getFinalMarks(year){
        Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY)
SupaBaseConn.getFinalMarks(year)
}
Function getAcademicRecord(){
        Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY)
SupaBaseConn.getAcademicRecord()
}
Function getProgressReport(){
        Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY)
SupaBaseConn.getProgressReport()
}
}
```

```
Finances.js
Import database.js

Class Finances{
        constructor(token){
                validateToken()
}

Function getFeeDetail(year){
        Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY)
SupaBaseConn.getFeeDetail(year)
}
Function getAgeAnalysis(){
        Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY)
SupaBaseConn.getAgeAnalysis()
}
Function getDepositDetail(){
            Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY)
SupaBaseConn.getDepositDetail()
}

Function getBursaryDetail(){
            Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY)
SupaBaseConn.getBursaryDetail()
}
}
```

```
Student.js
Import database.js

Class Finances{
            constructor(token){
                        validateToken()
}

Function getStudentDetails(year){
            Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY)
SupaBaseConn.getStudentDetails()
}
Function getStudentAddress(){
            Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY)
SupaBaseConn.getStudentAddress()
}
Function getStudentAdvisor(){
            Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY)
SupaBaseConn.getStudentAdvisor()
}

Function resetStudentPassword(){
Const newPassword = randomlyGeneratePassword();
            Const SupaBaseConn =
database(REACT_SUPABASE_URL,REACT_SUPABASE_ANON_KEY)
SupaBaseConn.resetStudentPassword(newPassword)
Return newPassword
}
}
```

FrontEnd

/* The frontend will follow an island/component based development process to minimize code redundancy*/

Folder Structure (Draft)

| - islands

| - - NavBar.jsx

| - - Login.jsx

| - - PinAuth.jsx

| - components

| - - RegularButton.jsx

| - - NavButton.jsx

| - - PinAuth.jsx

| - views

| - - Dashboard.jsx

| - - Finances.jsx

| - - StudentDetails.jsx

| - - AcademicRecord.jsx

| - - ProgressReport.jsx

| - - FinalResults.jsx

| - - auth

| - - - Login.jsx

| - lib

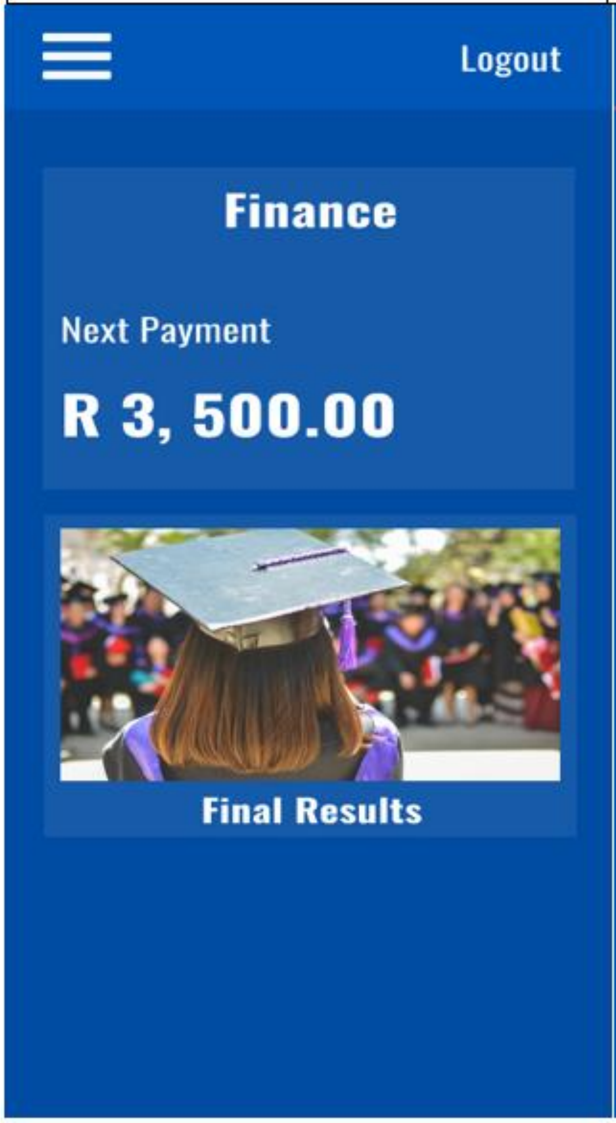
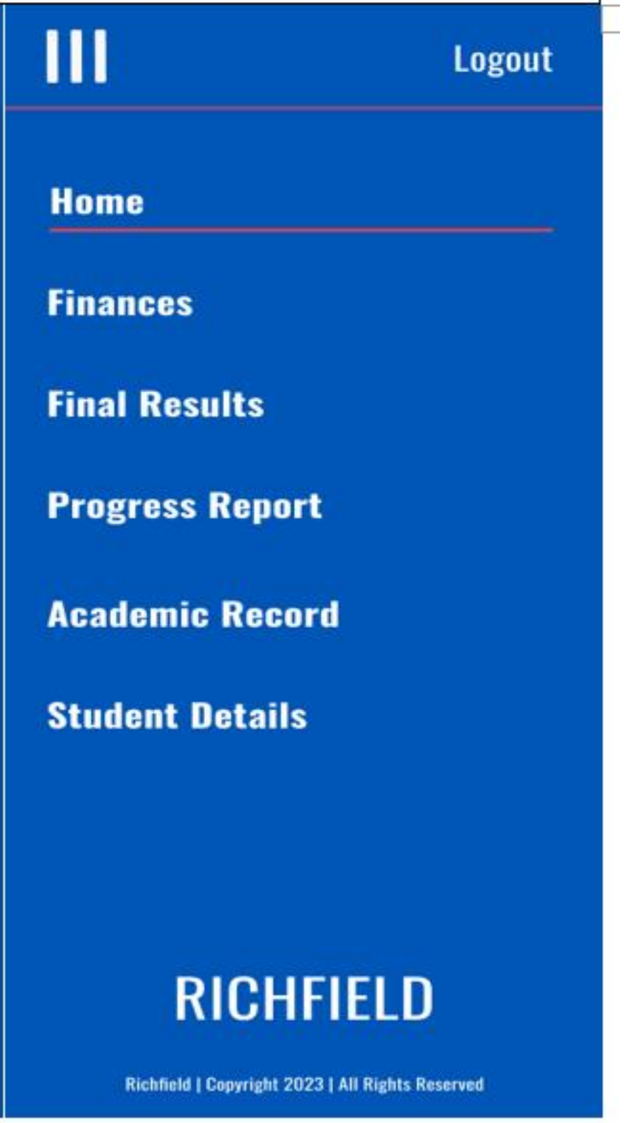
| - - assets (Images, videos, etc)

| - - style (All style sheets)

4.7 Interface Design

4.7.1 Menu Interface Design

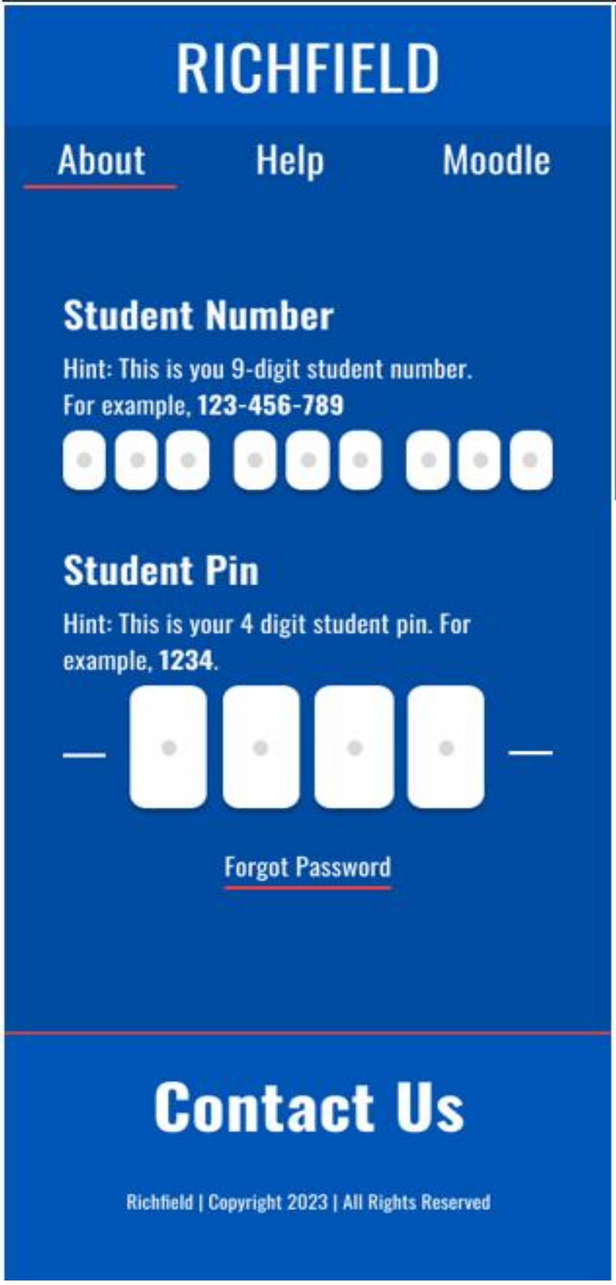
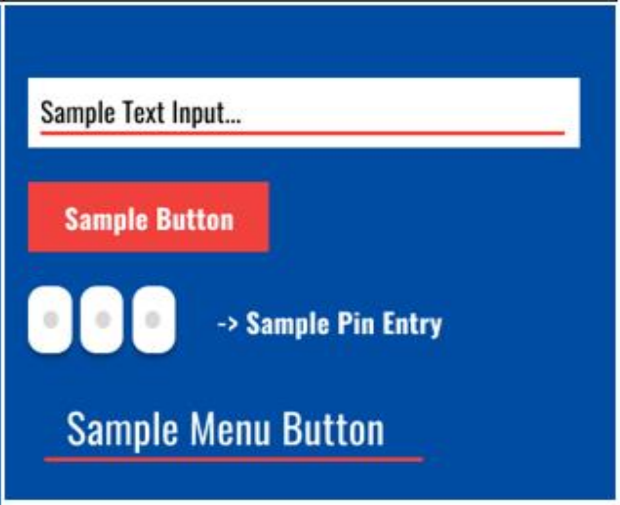
The menu interface will be designed to be intuitive and easily navigable, providing a seamless user experience. It will include categorization and organization of features for user accessibility. Samples of the menus are displayed below.

Dashboard Menu	Navigation Menu
 Logout Finance Next Payment R 3, 500.00 Final Results	 Logout Home Finances Final Results Progress Report Academic Record Student Details RICHFIELD Richfield | Copyright 2023 | All Rights Reserved

Menus are drafted in mobile format; this follows our mobile-first development flow. All elements will expand horizontally as the screen size increases. Where needed, orientation of content will be adjusted. For example, a vertical menu on mobile will become a horizontal menu on mobile with a scroll feature.

4.7.2 Input Design

The input design will focus on creating forms and input fields that are user-friendly, with appropriate validations and feedback to guide users during data entry.

Login Input	Sample Inputs
	

Hint: This is you 9-digit student number. For example, **123-456-789**

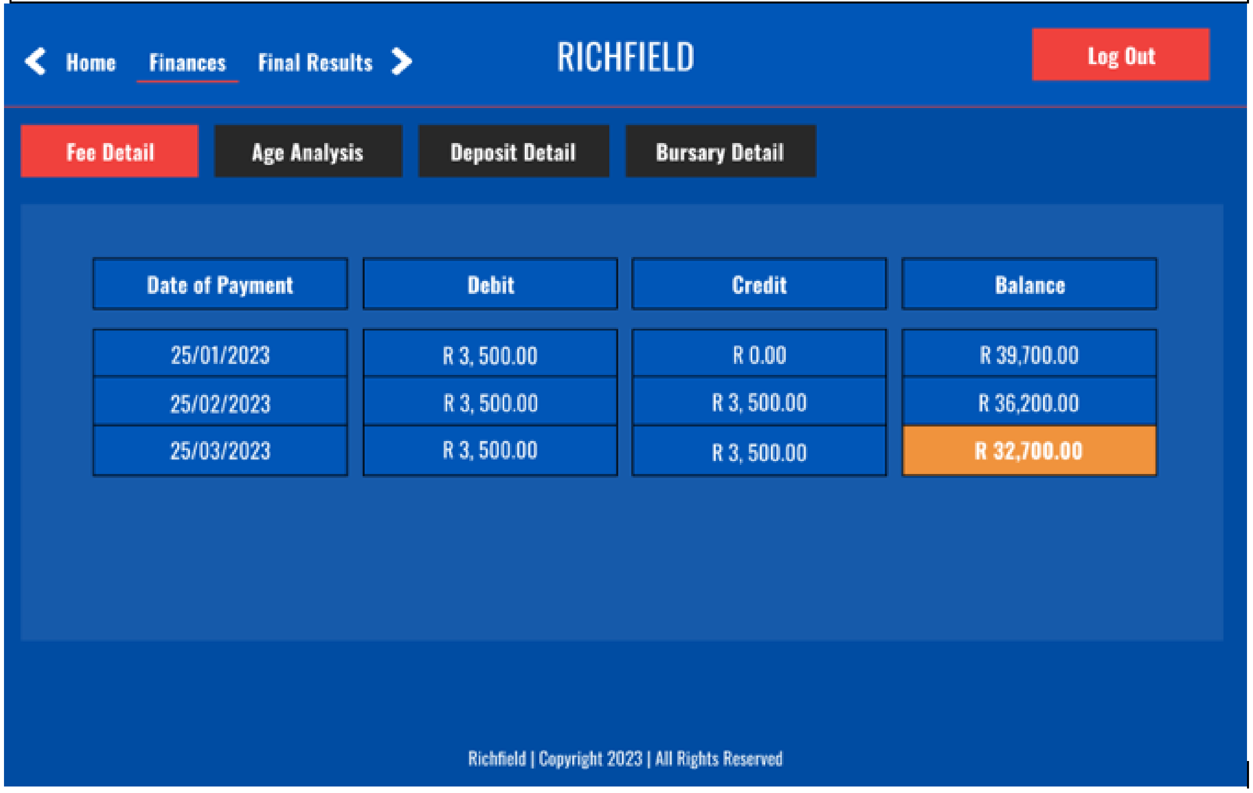
Hint: This is your 4 digit student pin. For example, **1234**.

Each form input will have an “invisible” field below it to display errors with the input. All inputs will stick to a “Material-Kit” sort of theme in addition to the predefined themes from the Richfield website.

4.7.3 Output Design

The output design will ensure that the information presented to users is clear, well-structured, and visually appealing, enhancing the overall user experience. The sample below visualizes this.

Sample Content Output



Date of Payment	Debit	Credit	Balance
25/01/2023	R 3, 500.00	R 0.00	R 39,700.00
25/02/2023	R 3, 500.00	R 3, 500.00	R 36,200.00
25/03/2023	R 3, 500.00	R 3, 500.00	**R 32,700.00**

Richfield | Copyright 2023 | All Rights Reserved

Note: In the actual project, data tables will be used. The DataTables API (JavaScript) allows developers to “instantly” style a typical HTML table and fit it with pagination, basic modern styling and a search/filter feature.

More about DataTables here.

4.8 Security Backup Design

This section will address software security concerns, detailing measures and protocols in place to ensure data security, privacy, and backup strategies to safeguard against potential threats.

4.8.1 Strategy

The database chosen (Supabase) regularly creates backups (On the premium version) of tables in the database. Manual backups are also an option as well as CSV exports to save locally as well as on other back devices, such as cloud storage or a hard drive. Branch protection can also be implemented on our version control system of choice, IE Git via GitHub.

Additionally, our hosting service, AWS, provides containerized weekly backups of our system that can be accessed from our dashboard.

4.8.2 Data Security

To ensure data security this project will follow OOP principles such as encapsulation, Inheritance and Abstraction. We will also be implementing access control where possible within the code. Additionally, although the database is only for test purposes, we will be using Row-Level-Security to further enhance security and limit access on a database level.

Row-Level-Security is a feature in Supabase that allows specific access (CREATE, READ, UPDATE and/or DELETE) under specific conditions, which may be:

- Only authenticated users can access
- Only admin users can access
- Only users with two factor authentication can access

For the purposes of this application, we will only allow read and update access to authenticates users (IE, users who have logged in)

4.8.3 Privacy

In order to ensure that the users' data is kept private to them and is not accessible to others, this program will make use of access tokens in order to reduce the risk of somebody being able to access the user's data without their consent. The access token would last approximately 1 hour. In addition to this, any items that are "Sensitive Information" will require a token check before being displayed. Such items may include final results, students' details, fee details, etc.

To implement the protection of sensitive information, we used authentication checks on each page before the content is loaded to make sure that the user is logged in before loading any data. If the user is not logged in, then they are kicked from the system and routed back to the login screen.

5. Implementation Phase

5.1 Introduction

In the fifth and final section of this project, we have implemented the revised frontend of the iEnabler system. Unfortunately, due to time constraints, we were not able to implement backend api integration with Supabase. Meaning that this project currently does not use dynamic data and instead uses pre-set data for demo purposes only. Our team soon realized, towards the end of the software development cycle, that since Supabase creates an API automatically, we would not need a separate server folder and hence the project structure has also changed in order to remove unwanted folders and files. The revised iEnabler system has been developed with a mobile-first development process, hosting through AWS Amplify (for CI/CD) and we have taken some inspiration from Richfield’s primary website for the aesthetics of the new iEnabler system.

Without further ado, let’s dive into this project!

5.2 Coding
Due to this being a fairly large project, we have elected to only display the main views and app.js files here and provide a link to our public GitHub repository to view the rest of the code. During development, we have elected to let one person make all commits, after we have sent the code to this person. The person elected was:

Shekhar Maharaj - 402101963

Please see below, the code and paths of the “main” files of the application.

./src/components/PinInput.jsx

```
import React, { useState } from 'react';

export default function PinInput(props) {

  /*
    Required Properties
    - OnSubmit (function)
    - autoSubmit (bool)
    - pinCount (int)
    - submitBtn
  */

  const [allowOnChange, setAllowOnChange] = useState(true);

  // Events
  const onKeyPressed = (e)=>{
    if (e.keyCode === 8 || e.keyCode === 46) {
      // Do not allow onChange event
      setAllowOnChange(false);
      // Do something when the delete or backspace key is pressed
      console.log('Delete or Backspace key pressed');
      // Focus on previous sibling if available
      if(e.target.previousSibling){
        // Disable curremt pin input
        e.target.disabled = true;
        let previousElement = e.target.previousSibling;
        previousElement.focus();
        previousElement.select();
      }
    }else if(e.keyCode === 13){
      if(e.target.value !== "" && e.target.nextSibling){
        let nextElement = e.target.nextSibling;
```

```
            nextElement.disabled = false;
            nextElement.focus();
        }
    }
};
const onChange = (e)=>{
    // If input on target is greater than 1 character in length, then remove the last character
    if(e.target.value.toString().length > 1){
        e.target.value = parseInt(e.target.value.toString().slice(1));
    }

        // Check if the target has any next siblings
        if(e.target.nextSibling){
            setAllowOnChange(true);
            if(allowOnChange){
                let nextElement = e.target.nextSibling;
                nextElement.disabled = false;
                nextElement.focus();
            }
        }else{
            // Handle auto-submit logic
            if(props.autoSubmit && e.target.value !== ""){
                onSubmit(e.target);
            }
        }

}
const onSubmit = (e)=>{
    // Handle submit code here
    let userInput = [e.value];

    while(e.previousSibling){
        userInput.push(e.previousSibling.value);
        e = e.previousSibling;
    }

    userInput = userInput.reverse().join('');
    console.log("userInput: "+userInput);

    // Call the pre-defined onsubmit property after any necessary expressions
    props.OnSubmit(userInput);
};
```

```
  return (
    <div className='PinInput-Container' style={{ flexDirection: props.flexDirection === "row" ?
"row" : "column" }}>

      <div className='Pins-Row'>
        {Array.from({ length: props.pinCount }, (_, i) => (
          <input key={`pin-${i}`} type="number" min="0" max="9" onKeyUp={onKeyPressed}
onChange={onChange} className='PinInput-Pin' id={`pin-${i}`} disabled={i !== 0}
placeholder='&#x25CF;' />
        ))}
      </div>
      <div className='PinInput-SubmitBtn-Container'>
        {props.submitBtn}
      </div>
    </div>
  );
}
```

./src/islands/NavBar.jsx

```
import NavBtn from "../components/NavBtn";
import FilledBtn from "../components/FilledBtn";
import "./../lib/style/nav.css";
import React, { useState ,useEffect,useRef,number} from 'react';

export default function NavBar(props) {
  const [width, setWidth] = useState(window.innerWidth);
  const [isMobile, setIsMobile] = useState(width <= 1024);
  const [showMobileNav, setShowMobileNav] = useState(false);

  const navWidget = <>
    <NavBtn highlight={props.highlight} href="/Dashboard" text="Dashboard" />
    <NavBtn highlight={props.highlight} href="/Academics" text="Academics" />
    <NavBtn highlight={props.highlight} href="/Finances" text="Finances" />
    <NavBtn highlight={props.highlight} href="/StudentInfo" text="Student Info" />
  </>

  const navItemsListRef = useRef(null); // Ref for the nav-items-list

  // Events
  const toggleMobileNav = (e) => {
    setShowMobileNav(!showMobileNav);
  };

  const handleWindowSizeChange = () => {
    setWidth(window.innerWidth);
    setIsMobile(width <= 1024);
  };

  const scrollLeft = () => {
    if (navItemsListRef.current) {
      navItemsListRef.current.scrollLeft -= 100; // Adjust the scroll distance as needed
    }
  };

  const scrollRight = () => {
    if (navItemsListRef.current) {
      navItemsListRef.current.scrollLeft += 100; // Adjust the scroll distance as needed
    }
  };

  useEffect(() => {
    window.addEventListener('resize', handleWindowSizeChange);
    return () => {
      window.removeEventListener('resize', handleWindowSizeChange);
    };
  }, []);

  return (
    <>
      <nav>
        {!isMobile ? (
          <div id="nav-items-container">
            <button onClick={scrollLeft}>
                <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="white"
className="bi bi-chevron-left" viewBox="0 0 16 16">
                    <path fillRule="evenodd" d="M11.354 1.646a.5.5 0 0 1 0 .708L5.707 8l5.647
5.646a.5.5 0 0 1-.708.708l-6-6a.5.5 0 0 1 0-.708l6-6a.5.5 0 0 1 .708 0z"/>
                </svg>
            </button>
            <div id="nav-items-list" ref={navItemsListRef}>
              {navWidget}
            </div>
            <button onClick={scrollRight}>
                <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="white"
```

```
className="bi bi-chevron-right" viewBox="0 0 16 16">
                    <path fillRule="evenodd" d="M4.646 1.646a.5.5 0 0 1 .708 0l6 6a.5.5 0 0 1 0 .708l-
6 6a.5.5 0 0 1-.708-.708L10.293 8 4.646 2.354a.5.5 0 0 1 0-.708z"/>
                </svg>
            </button>
          </div>
        ) : null}
        {!isMobile ? null : (
            <div id="nav-toggle-custom-icon" onClick={toggleMobileNav}>
                <svg className="hb" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 10 10"
stroke="#eee" strokeWidth=".6" fill="rgba(0,0,0,0)" strokeLinecap="round" style={{cursor:
"pointer", width: "50px"}}>
                    <path d="M2,3L5,3L8,3M2,5L8,5M2,7L5,7L8,7">
                        <animate dur="0.2s" attributeName="d"
values="M2,3L5,3L8,3M2,5L8,5M2,7L5,7L8,7;M3,3L5,5L7,3M5,5L5,5M3,7L5,5L7,7"
fill="freeze" begin="start.begin" />
                        <animate dur="0.2s" attributeName="d"
values="M3,3L5,5L7,3M5,5L5,5M3,7L5,5L7,7;M2,3L5,3L8,3M2,5L8,5M2,7L5,7L8,7"
fill="freeze" begin="reverse.begin" />
                    </path>
                        <rect width="10" height="10" stroke="none">
                            <animate dur="2s" id="reverse" attributeName="width" begin="click" />
                        </rect>
                        <rect width="10" height="10" stroke="none">
                            <animate dur="0.001s" id="start" attributeName="width" values="10;0"
fill="freeze" begin="click" />
                            <animate dur="0.001s" attributeName="width" values="0;10" fill="freeze"
begin="reverse.begin" />
                        </rect>
                    </svg>
            </div>
        )}
        <div id="logo">
          <span>RICHFIELD</span>
        </div>
        {isMobile ? null :
          <div id="logout-container">
            <FilledBtn onClick={()=>{window.location.href = "/"}} label="Log Out"/>
          </div>
        }
      </nav>
      {isMobile ? (
        <div id="nav-items-container" className={showMobileNav ? 'md-active' : 'md-hide'}>
      <div id="nav-items-list">
        {navWidget}
      </div>
     </div>
    ) : null}
   </>
  );
}
```

./App.js

```
import './App.css';
import {  createBrowserRouter,
  RouterProvider,} from "react-router-dom";
// Import Layout
import Layout from './components/Layout';
// Import Pages
import Login from "./views/auth/Login";
import Academics from "./views/Academics"
import Dashboard from "./views/Dashboard";
import Finances from "./views/Finances";
import StudentInfo from './views/StudentInfo';


function App() {

    // initialize a browser router and define paths
    const router = createBrowserRouter([
      {
        element: <Layout />,
        children:
        [
          {
            path: "/",
            element: <Login />,
          },
          {
            path: "/Dashboard",
            element: <Dashboard />,
          },
          {
            path: "/Finances",
            element: <Finances />,
          },
          {
            path: "/Academics",
            element: <Academics />,
          },
          {
            path: "/StudentInfo",
            element: <StudentInfo />,
          },
        ]
      }
    ])

  return (
    <RouterProvider router={router} />
  );
}

export default App;
```

./src/views/auth/Login.jsx

```
import PinInput from "../../components/PinInput";

import NavBtn from "../../components/NavBtn";

import FilledBtn from "../../components/FilledBtn";

import { useState, useRef } from "react";

import { createClient } from '@supabase/supabase-js'

import Swal from 'sweetalert2';

import autoLogin from './../../util/autoLogin';

import emailjs from '@emailjs/browser';

import forgotPassword from "../../util/forgotPassword";


export default function Login(){

  autoLogin();

  const supabaseUrl = process.env.REACT_APP_SUPABASE_URL;
  const supabaseKey = process.env.REACT_APP_SUPABASE_KEY;

  const supabase = createClient(supabaseUrl, supabaseKey)

  // const form = useRef();

  const [studentNumber, setStudentNumber] = useState(null);

  const [studentPin, setStudentPin] = useState(null);


  const saveStudentNumber = async (input)=>{

    setStudentNumber(input.toString());


    if(studentPin !== null){

      // Make API request

      console.log(studentNumber);

      console.log(studentPin);

      const { data, error } = await supabase.auth.signInWithPassword({

        email: `${studentNumber}@my.richfield.ac.za`,

        password: studentPin

      });


      console.log(data);

      console.log(`LOGIN\nStudent Number: ${studentNumber}\nPin:
${input.toString()}\n${error}`);

      if(error){

        Swal.fire({

          title: 'Error!',

          text: `Invalid login credentials! Please try again later`,

          icon: 'error',

          confirmButtonText: 'Okay'

        })
      }else{
        localStorage.setItem('studentNumber',studentNumber);
        localStorage.setItem('token', data.session.access_token);
        window.location.href = "/Dashboard";
      }
    }
  }
  const login = async (input)=>{
    setStudentPin(input.toString());

    // Make API request
    if(studentNumber !== null){
      console.log(studentNumber);
      console.log(studentPin);

      const { data, error } = await supabase.auth.signInWithPassword({
        email: `${studentNumber}@my.richfield.ac.za`,
        password: input.toString()
      });

      console.log(data);
      console.log(`LOGIN\nStudent Number: ${studentNumber}\nPin:
${input.toString()}\n${error}`);
      if(error){
        Swal.fire({
          title: 'Error!',
          text: `Invalid login credentials! Please try again later`,
          icon: 'error',
            confirmButtonText: 'Okay'
          })
        }else{
          localStorage.setItem('studentNumber',studentNumber);
          localStorage.setItem('token', data.session.access_token);
          window.location.href = "/Dashboard";
        }
      }
  };
  const showAbout = ()=>{
    Swal.fire({
      title: 'About Richfields\' iEnabler System',
      html: `
        <p>iEnabler is Richfield College's student management system, providing a user-friendly interface for students to access their grades, exam and assignment marks, and financial information.<br/><br/> This comprehensive tool ensures transparency and convenience, allowing students to easily navigate and print academic records and financial details. As a key companion in the academic journey, iEnabler reflects Richfield's commitment to leveraging technology for efficient student management.</p>
      `,
      icon: 'info',
      confirmButtonText: 'Got it!'
    })
  }
  const showHelp = ()=>{
    Swal.fire({
      title: 'Frequently Asked Questions',
      html: `
        <div>
```

```
                <h2>What is iEnabler?</h2>

                <p>iEnabler is Richfield College's student management system, designed to provide students with easy access to their grades, exam and assignment marks, and financial information.</p>


                <h2>How do I access iEnabler?</h2>

                <p>You can access iEnabler through the college's official website. Use your student credentials to log in and explore the platform.</p>


                <h2>Can I print my academic records through iEnabler?</h2>

                <p>Yes, iEnabler allows you to print your academic records and financial information for your convenience.</p>


                <h2>Is iEnabler mobile-friendly?</h2>

                <p>Absolutely! iEnabler is designed to be responsive, making it accessible and user-friendly on various devices, including mobile phones and tablets.</p>


                <h2>How can I get support for iEnabler?</h2>

                <p>For support, you can contact the college's IT department or refer to the provided user guide available on the iEnabler platform.</p>

            </div>

          `,

          icon: 'info',

          confirmButtonText: 'Okay'

      })

   }


   const form = useRef();

   const [popupVisible, setPopupVisible] = useState(false);
```

```
  const [resetLink,setResetLink] = useState("null");

  // Logic to send email to student admin
  const sendEmail = (e)=>{
    e.preventDefault();
    // forgotPassword("shekothegreat1@gmail.com")
    emailjs.sendForm('service_mhvgide', 'template_lqhyc0a', form.current,
'W05BD8jtQx8F6mINv')
    .then((result) => {
      console.log(result.text);
    }, (error) => {
      console.log(error.text);
    });
  }

  // Markup to generate forgot-password form
  var formMarkup = <form className="forgot-form" onSubmit={sendEmail} ref={form}>
    <h1 className="form-title">Forgot Password</h1>

    <div className="form-input">
      <label htmlFor="to_name">Name</label>
      <input type="text" name="to_name" id="to_name" />
    </div>

    <div className="form-input">
      <label htmlFor="student_number">Student Number</label>
      <input type="number" name="student_number" id="student_number"
value={studentNumber ? studentNumber : null}/>
```

```
    </div>


    <div className="form-input">
      <label htmlFor="contact_email" title="This is the email that you prefer to be contacted
on">Preffered Contact Email</label>
      <input type="email" name="" id="" />
      <p>You can leave this blank if it is your richfield email. That is {studentNumber ?
studentNumber : "402000111"}@my.richfield.ac.za</p>
    </div>


    <input type="hidden" name="reset_link" id="reset_link" value={resetLink}/>
    <input type="hidden" name="send_to" id="send_to"
value="shekothegreat1@gmail.com"/>


    <div className="form-actions">
      <input className="FakeFilledBtn" type="submit" value="Send" />
      <FilledBtn label="Close"  onClick={() => setPopupVisible(false)} />
    </div>
  </form>;


  return(
    <>
      {
      // Determine whether or not to show the forgot password popup
      popupVisible &&
        <div style={{zIndex:6000,position: 'fixed', top: 0, left: 0, width: '100%', height: '100%',
backgroundColor: 'rgba(0, 0, 0, 0.6)', display: 'flex', justifyContent: 'center', alignItems:
'center'}}>
          <div className="cardBox">
```

```
              <div>
                {formMarkup}
              </div>
            </div>
          </div>
        }

        <header id="login-header">
          RICHFIELD
        </header>

        <div className="content">
          <nav>
            <NavBtn href={"#"} onclick={showAbout} text={"About"} />
            <NavBtn href={"#"} onclick={showHelp} text={"Help"} />
            <NavBtn href={"https://learning.richfield.ac.za/HET/login/index.php"}
text={"Moodle"} newTab={true}/>
          </nav>

          <div id="auth-box">
            <div id="student-number-container">
              <span>Student Number</span>
              <p>Hint: This is you 9-digit student number. For example, <strong>123-456-
789</strong></p>
              <PinInput pinCount={9} flexDirection="column" autoSubmit={true}
OnSubmit={saveStudentNumber}/>
            </div>
            <div id="student-pin-container">
              <span>Student Pin</span>

              <p>Hint: This is your 4 digit student pin. For example,
<strong>1234</strong></p>

              <div id="student-pin">

                <div className="decorative-line"></div><PinInput pinCount={4}
flexDirection="column" autoSubmit={true} OnSubmit={login}/><div className="decorative-
line"></div>

              </div>

              <div id="forgot-password-container">


                <FilledBtn label="Forgot Password?" onClick={()=>{setPopupVisible(true)}}
id="forgot-password-btn"/>

              </div>

            </div>

          </div>

        </div>


        <footer>

          <div className="contact-cta">

            <a href="https://www.richfield.ac.za/quick-application/"
target="_blank"><strong>Contact Us</strong></a>

          </div>

          <div className="legal">

            <span>

            Richfield | Copyright 2023 | All Rights Reserved

            </span>

          </div>

        </footer>

      </>
    );
}
```

./src/views/Academics.jsx

```
import { useState, useEffect } from "react";
import Academics_AcademicRecord from "../islands/Academics_AcademicRecord";
import Academics_ProgressReport from "../islands/Academics_ProgressReport";
import Academics_ExamResults from "../islands/Academics_ExamResults";
import "./../lib/style/academic.css";
import { createClient } from '@supabase/supabase-js';
import processAcademicData from '../util/handleAcademicRecord';
import handleExamData from "../util/handleExamResults";
import getStudentInfo from '../util/getStudentInfo';

export default function Academics(props){
    const [selectedOption, setSelectedOption] =
useState(localStorage.getItem('selectedOption') ? localStorage.getItem('selectedOption') :
"academic_record");
    const [results, setResults] = useState(null);
    const [examData, setExamData] = useState(null);
    const hasStudentInfo = getStudentInfo();
    let widget = null;

    const supabase = createClient(
        process.env.REACT_APP_SUPABASE_URL,
        process.env.REACT_APP_SUPABASE_KEY
    );
```

```
const fetchResults = async () => {
  const { data, error } = await supabase
    .from('results')
    .select('*').eq('student_number',localStorage.getItem('studentNumber'));
  if (error) {console.log("Error: ", error);}
  else {setResults(processAcademicData(data));setExamData(handleExamData(data))}
};

useEffect(() => {
  fetchResults();
}, []);

switch (selectedOption) {
  case "academic_record":
    widget = results == null ? <></> : <Academics_AcademicRecord results={results} />;
    break;
  case "progress_report":
    widget = results == null ? <></> :  <Academics_ProgressReport results={results} />
    break;
  case "exam_results":
    widget = results == null ? <></> :  <Academics_ExamResults results={examData} />
    break;
  default:
    widget = <>
      <h1>Oops...Something went wrong!</h1>
      <p>An error occured while loading the page content. Please conact your
administrator or try again later.</p>
```

```
        </>
        break;
    }

    return (
        <div id="finances">
            <div id="view-options">
                <button onClick={()=>{setSelectedOption("academic_record");}}
className={selectedOption === "academic_record" ? "option option-
selected":"option"}>Academic Record</button>
                <button onClick={()=>{setSelectedOption("progress_report");}}
className={selectedOption === "progress_report" ? "option option-
selected":"option"}>Progress Report</button>
                <button onClick={()=>{setSelectedOption("exam_results");}}
className={selectedOption === "exam_results" ? "option option-selected":"option"}>Exam
Results</button>
            </div>
            <div id="finance-content">
                {widget}
            </div>
        </div>
    );
}
```

./src/views/Dashboard.jsx

```
import FilledBtn from "../components/FilledBtn";
import "../lib/style/dashboard.css"
import finalResults from "../lib/assets/finalResults.jpg";
import academicRecord from "../lib/assets/academicRecord.jpg";
import progressReport from "../lib/assets/progressReport.jpg";
import profile from "../lib/assets/profile.jpg";

export default function Dashboard(props){
  return(
    <div id="dashboard">
      <div id="components">
        <div className="component" id="finance-due" onClick={()=>{window.location = 
"/Finance"}}>
          <div className="component-heading">
            <span>Finance</span>
          </div>
          <div className="component-content">
            <span>Next Payment</span>
            <span><strong>R 3, 500.00</strong></span>
          </div>
          {/* <div className="view-now-btn">
            <FilledBtn label="View" onClick={()=>{window.location.href = "/Finance"}}/>
          </div> */}
        </div>
        <div className="component orange" id="final-results" 
onClick={()=>{window.location = "/Academics"}}>
          <div className="component-content">
            <img src={finalResults} alt="Final Results"/>
          </div>
          <div className="component-heading">
            <span>Final Results</span>
          </div>
          {/* <div className="view-now-btn">
            <FilledBtn label="View" onClick={()=>{window.location.href = "/Final-
Results"}}/>
          </div> */}
        </div>
        <div className="component pink" id="academic-record" 
onClick={()=>{window.location = "/Academics"}}>
          <div className="component-content">
            <img src={academicRecord} alt="Academic Record"/>
          </div>
          <div className="component-heading">
            <span>Academic Record</span>
          </div>
          {/* <div className="view-now-btn">
            <FilledBtn label="View" onClick={()=>{window.location.href = "/Dashboard"}}/>
          </div> */}
```

```
            </div>
            <div className="component purple" id="progress-report"
onClick={()=>{window.location = "/Academics"}}>
                <div className="component-content">
                    <img src={progressReport} alt="Progress Report"/>
                </div>
                <div className="component-heading">
                    <span>Progress Report</span>
                </div>
                {/* <div className="view-now-btn">
                    <FilledBtn label="View" onClick={()=>{window.location.href = "/Dashboard"}}/>
                </div> */}
            </div>
        </div>
        <div id="bottom-content">
            <div className="component" id="student-advisor">
                <div className="component-heading">
                    <span>Student Advisor Details</span>
                </div>
                <div className="component-content">
                    <div id="details">
                    <p>You may contact your designated student advisor by using the following
details:</p>

                    <table>
                        <tbody>
                            <tr>
                                <td>
                                    Name:
                                </td>
                                <td>
                                    Khanyisile Ngobo
                                </td>
                            </tr>
                            <tr>
                                <td>
                                    Phone Number:
                                </td>
                                <td>
                                    (000)-000-0000
                                </td>
                            </tr>
                            <tr>
                                <td>
                                    Email Address:
                                </td>
                                <td>
                                    KhanyisileN@richfield.ac.za
                                </td>
                            </tr>
                            <tr>
                                <td>
                                    Office Hours
                                </td>
                                <td>
                                    Monday - Friday<br/>
                                    08:00AM - 05:00PM
                                </td>
                            </tr>
                            <tr>
                                <td>
                                    Average Response Time (Email)
                                </td>
                                <td>
                                    1 hour
                                </td>
                            </tr>
                        </tbody>
                    </table>
                    </div>
                    <div className="profile">
                        <img src={profile} alt="Profile" />
                    </div>
                </div>
                <div className="view-now-btn">
                    <FilledBtn label="Contact" onClick={(e)=>{e.preventDefault();window.location =
"mailto:recipient@example.com?subject=Your%20Prefilled%20Subject"}}/>
                </div>
            </div>
            <div id="quote">
                <em>"<strong>EDUCATION</strong> is the <strong>KEY</strong><br/>to
<strong>UNLOCK</strong> the <strong>GOLDEN DOOR</strong><br/> of
<strong>FREEDOM</strong>"</em>
            </div>
        </div>
      </div>
```

```
   );
}
```

./src/views/Finance.jsx

```
import { useState,useEffect } from "react";
import Finance_FeeDetail from "../islands/Finance_FeeDetail";
import Finance_AgeAnalysis from "../islands/Finance_AgeAnalysis";
import Finance_BursaryDetail from "../islands/Finance_BursaryDetail";
import Finance_DepositDetail from "../islands/Finance_DepositDetail";
import "./../lib/style/finances.css";
import { createClient } from '@supabase/supabase-js';

export default function Finances(){

  // All select option string use underscores for spaces and are always lower cased
  const [selectedOption, setSelectedOption] = useState("fee_detail");
  let widget = null;

  const [wait,setWait] = useState(true)
  const[finance,setFinance]= useState(null);

  useEffect(() => {
    const fetchSupabaseData = async () => {
     const supabase = createClient(
       process.env.REACT_APP_SUPABASE_URL,
       process.env.REACT_APP_SUPABASE_KEY
     );
```

```
      try {
        const { data, error } = await supabase
          .from('studentFinance')
          .select(`*`)
          .eq('studentNumber',localStorage.getItem('studentNumber'));
        if (error) {
            setWait(true);
            console.log(error)
        } else {
            console.log(data)
            setFinance(data);
            setWait(false);
        }
      } catch (error) {
        setWait(true);
        console.log(error)
      }
    };

    fetchSupabaseData();
  }, []);

// Determine widget to show
switch (selectedOption) {
    case "fee_detail":
        widget = <Finance_FeeDetail financeData={finance}/>;
    break;
    case "age_analysis":
      widget = <Finance_AgeAnalysis/>
    break;
    case "deposit_detail":
      widget = <Finance_DepositDetail/>
    break;
    case "bursary_detail":
      widget = <Finance_BursaryDetail/>
    break;

    default:
      widget = <>
        <h1>Oops...Something went wrong!</h1>
        <p>An error occured while loading the page content. Please conact your
administrator or try again later.</p>
      </>
      break;
  }

  return (
    <div id="finances">
      <div id="view-options">
        <button onClick={()=>{setSelectedOption("fee_detail");}} className={selectedOption
=== "fee_detail" ? "option option-selected":"option"}>Fee Detail</button>
        <button onClick={()=>{setSelectedOption("age_analysis");}}
className={selectedOption === "age_analysis" ? "option option-selected":"option"}>Age
Analysis</button>
        <button onClick={()=>{setSelectedOption("deposit_detail");}}
className={selectedOption === "deposit_detail" ? "option option-
selected":"option"}>Deposit Detail</button>
        <button onClick={()=>{setSelectedOption("bursary_detail");}}
```

```
className={selectedOption === "bursary_detail" ? "option option-
selected":"option"}>Bursary Detail</button>

         </div>

         <div id="finance-content">

            {

               wait ? "Loading...":

               widget

            }

         </div>

      </div>

   );

}
```

./src/views/StudentInfo.jsx

```
import { useState, useEffect } from "react";

import "./../lib/style/student.css"

import { createClient } from '@supabase/supabase-js';

import UpdateUserDetailsForm from "../islands/UpdateUserDetailsForm";


export default function StudentInfo(){


   const [student,setStudent] = useState(null);

   const [address,setAddress] = useState(null);

   const [wait,setWait] = useState(true)


   useEffect(() => {
    const fetchSupabaseData = async () => {
      const supabase = createClient(
        process.env.REACT_APP_SUPABASE_URL,
        process.env.REACT_APP_SUPABASE_KEY
      );

      try {
        const { data, error } = await supabase
          .from('student')
          .select(`*, address!inner(*)`)
          .eq('studentNumber',localStorage.getItem('studentNumber'));
        if (error) {
            setWait(true);
            console.log(error)
        } else {
            console.log(data[0])
            setStudent(data[0]);
            setWait(false);
        }
      } catch (error) {
        setWait(true);
      }
    };

    fetchSupabaseData();
  }, []);

return(
    <div id="student">
      <h1>Student Information</h1>
      { student != 'error' ?
      <>
        <div id="details-tools">
          {
            wait ? null :
            <button id="update-details-btn" data-student={JSON.stringify(student)}
onClick={UpdateUserDetailsForm}>
              <i>
                <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
fill="currentColor" class="bi bi-pencil-fill" viewBox="0 0 16 16">
                  <path d="M12.854.146a.5.5 0 0 0-.707 0L10.5 1.793 14.207 5.5l1.647-
1.646a.5.5 0 0 0 0-.708l-3-3zm.646 6.061L9.793 2.5 3.293 9H3.5a.5.5 0 0 1 .5.5v.5h.5a.5.5 0 0
1 .5.5v.5h.5a.5.5 0 0 1 .5.5v.5h.5a.5.5 0 0 1 .5.5v.207l6.5-6.5zm-7.468 7.468A.5.5 0 0 1 6
13.5V13h-.5a.5.5 0 0 1-.5-.5V12h-.5a.5.5 0 0 1-.5-.5V11h-.5a.5.5 0 0 1-.5-.5V10h-.5a.499.499
0 0 1-.175-.032l-.179.178a.5.5 0 0 0-.11.168l-2 5a.5.5 0 0 0 .65.65l5-2a.5.5 0 0 0 .168-.11l.178-
.178z"/>
                </svg>
              </i>
              Update
            </button>
          }
        </div>
        <table>
        <tbody>
          <tr>
            <td>
              Full Name
            </td>
    <td>{wait ? "...Loading" : `${student.fName} ${student.lName}`}</td>
  </tr>
  <tr>
    <td>
      Student Number
    </td>
    <td>{wait ? "...Loading" : `${student.studentNumber}`}</td>
  </tr>
  <tr>
    <td>
      Mobile Number
    </td>
    <td>{wait ? "...Loading" : `${student.contactNumber}`}</td>
  </tr>
  <tr>
    <td>
      Email Address
    </td>
    <td>{wait ? "...Loading" : `${student.emailAddress}`}</td>
  </tr>
  <tr>
    <td>
      Postal Address
    </td>
    <td>
      {wait ? "...Loading" : `${student.address[0].street}`}<br/>
      {wait ? "...Loading" : `${student.address[0].city}`}<br/>
      {wait ? "...Loading" : `${student.address[0].country},
${student.address[0].postalCode}`}
            </td>
          </tr>
          <tr>
            <td>
              Physical Address
            </td>
            <td>
            {wait ? "...Loading" : `${student.address[0].street}`}<br/>
              {wait ? "...Loading" : `${student.address[0].city}`}<br/>
              {wait ? "...Loading" : `${student.address[0].country},
${student.address[0].postalCode}`}
            </td>
          </tr>
          <tr>
            <td>
              Study Address
            </td>
            <td>
            {wait ? "...Loading" : `${student.address[0].street}`}<br/>
              {wait ? "...Loading" : `${student.address[0].city}`}<br/>
              {wait ? "...Loading" : `${student.address[0].country},
${student.address[0].postalCode}`}
            </td>
          </tr>
        </tbody>
      </table>
      </>
```

```
        :<>
          <h1>Error</h1>
          <p>An error occured while loading your student information. Please contact your student advisor or try again later.</p>
        </>
        }
      </div>
  );
}
```

src/lib/style/dashboard.css

```
/* Base components */
#dashboard{
  display: flex;
  flex-direction: column;
  gap: 30px;
  padding: 5px;
}
#components{
  display: flex;
  flex-direction: column;
  gap: 30px;
  padding: 5px;
}
.component{
  /* padding: 10px; */
  background-color: var(--blue-modal);
  min-height: 200px;
  display: grid;
  grid-template-areas: 'header' 'content';
}
#student-advisor{
  padding: 10px;
}
#student-advisor table tbody tr td{
```

```css
    text-align: left;
}
.component-content img{
    max-width: 100%;
}

/* Specific components */
.component{
    transition: all 1s;
}
.component .component-content{
    overflow: hidden;
}
.component:hover .component-content img{
    transition: all 1s;
    transform: scale(1.1,1.1);
}
#finance-due{
    padding: 5px;
}
#finance-due:hover .component-heading{
    transition: all 1s;
    font-size: 50px;
}
#finance-due:hover .component-content{
    transition: all 1s;
    font-size: 40px;
}
#finance-due{
    display: grid;
    grid-template-areas: 'header' 'content' 'button';
}
#finance-due .component-content{
    display: flex;
    flex-direction: column;
    gap: 1px;
}
#finance-due .component-content span:nth-child(1){
    font-size: 18px;
}
#finance-due .component-content span:nth-child(2){
    font-size: 25px;
}
```

```css
/* Sub components */
.component-heading{
   display: flex;
   flex-direction: row;
   justify-content: left;
   align-items: center;
   font-size: 30px;
   font-weight: bold;
   color: white;
   padding: 5px;
}
.component-content{
   color: white;
}
.view-now-btn{
   display: flex;
   justify-content: left;
   align-items: center;
   padding-top: 10px;
}
.view-now-btn .FilledBtn{
   min-width: 100px;
   justify-content: center;
}
.view-now-btn .FilledBtn .FilledBtn-Text{
   width: 100%;
   font-size: 19px;
}

#bottom-content{
   display: flex;
   flex-direction: column;
   gap: 20px;
}
#quote{
   width: 100%;
   display: flex;
   justify-content: center;
   font-size: 50px;
   align-items: center;
   color: white;
}
```

```css
.orange:hover{
   background-color: orange;
}
.pink:hover{
   background-color: rgb(255, 104, 129);
}
.purple:hover{
   background-color: rgb(154, 33, 154);
}
/* Desktop */
@media only screen and (min-width: 768px) {
   #dashboard{
      padding: 20px;
   }
   #components{
      flex-direction: row;
   }
   #components .component{
      width: 25%;
   }
   .component-content img{
      width: 100%;
      max-height: 250px;
   }
   #components .component .component-content{
      display: flex;
      justify-content: center;
      align-items: center;
   }
   .component-heading, .view-now-btn{
      justify-content: center;
   }
   #student-advisor{
      width: 50%;
   }
   #student-advisor .component-content{
      display: flex;
      flex-direction: row;
      gap: 5px;
   }
   #details{
      width: 100%;
      display: flex;
```

```
        flex-direction: column;
        justify-content: center;
        align-items: center;
    }
    .profile{
        padding: 50px;
        width: 100%;
        display: flex;
        justify-content: center;
        align-items: center;
    }
    .profile img{
        width: 200px;
        height: auto;
        border-radius: 100%;
    }
    #bottom-content{
        flex-direction: row;
    }
    #quote{
        width: 50%;
    }
}
@keyframes zoom-in-zoom-out {
    0% {
      transform: scale(1, 1);
    }
    50% {
      transform: scale(1.5, 1.5);
    }
}
```

./src/lib/style/login.css

```
#login-header{
    width: 100%;
    height: 70px;
    background-color: var(--blue-light);
    display: flex;
    justify-content: center;
    align-items: center;
    color: white;
    font-size: 35px;
```

```css
    font-weight: 400;
    letter-spacing: -0.5px;
    border-bottom: 2px solid var(--red);
    box-sizing: border-box;
}
/* Navigation */
nav{
    display: flex;
    justify-content: center;
    align-items: center !important;
    gap: 10px;
    overflow: auto;
    padding: 5px 0 5px 0;
    height: 60px;
}
nav .NavBtn span{
    letter-spacing: -1px;
}

/* Student input */
#student-number-container{
    padding: 30px;
}
#student-pin-container{
    padding: 0 30px 0 30px;
}
#student-number-container, #student-pin-container{
    color: white;
    font-weight: lighter;
}
#student-number-container span, #student-pin-container span{
    font-size: 25px;
    letter-spacing: 1px;
    font-weight: bold;
}
#student-number-container .PinInput-Pin{
    width: 20px;
    height: 35px;
}
#student-pin-container .PinInput-Pin{
    width: 30px;
    height: 45px;
}
```

```css
#student-number-container .PinInput-Container .Pins-Row .PinInput-Pin:nth-child(3),
#student-number-container .PinInput-Container .Pins-Row .PinInput-Pin:nth-child(6){
    margin-right: 5px;
}
#student-pin{
    display: flex;
    justify-content: center;
    align-items: center;
    gap: 10px;
}

/* Line deco */
.decorative-line{
    height: 3px;
    background-color: white;
    width: 20%;
}
/* Forgot Password */
#forgot-password-container{
    display: flex;
    justify-content: center;
    align-items: center;
    padding: 30px;
    box-sizing: border-box;
}
#forgot-password-container .FilledBtn .FilledBtn-Text{
    font-size: 15px;
}

/* Footer */
footer{
    width: 100%;
    background-color: var(--blue-light);
    color: white;
    padding: 30px;
    display: flex;
    gap: 30px;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    border-top: 2px solid var(--red);
    box-sizing: border-box;
}
```

```css
footer .contact-cta span{
  font-size: 30px;
}
footer .legal{
  font-size: 15px;
}
/* Desktop */
@media only screen and (min-width: 768px) {
  #login-header{
    padding-left: 20px;
  }
  #login-header,nav{
    justify-content: left;
  }
  #student-number-container, #student-pin-container{
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
  }
  #student-pin-container{
    padding-top: 60px;
  }
  #student-pin{
    width: 100%;
  }
  #student-number-container .PinInput-Pin{
    width: 30px;
    height: 45px;
  }
}
```

./src/util/auth.js

```javascript
import { createClient } from '@supabase/supabase-js';

const supabaseUrl = process.env.REACT_APP_SUPABASE_URL;
const supabaseKey = process.env.REACT_APP_SUPABASE_KEY;
const supabase = createClient(supabaseUrl, supabaseKey)

// Validate the user is logged in
export default async function validateUser(){
    let token = localStorage.getItem('token');

    // Kick user if toke is not set
    if(!token && window.location.href != "/") {
        window.location.href = "/";
    }

    const { data: { user } } = await supabase.auth.getUser(token)
    console.log("User auth check");

    // Kick user if JWT is not valid
    if(!user){
        window.location.href = "/";
    }
}
```

./src/util/autoLogin.js

```
import { createClient } from '@supabase/supabase-js';

const supabaseUrl = process.env.REACT_APP_SUPABASE_URL;
const supabaseKey = process.env.REACT_APP_SUPABASE_KEY;
const supabase = createClient(supabaseUrl, supabaseKey)

// Validate the user is has token
export default async function autoLogin(){
    let token = localStorage.getItem('token');

    // Login user if toke is not set
    if(token){
        const { data: { user } } = await supabase.auth.getUser(token)
        console.log(user);

        // Login user if JWT is not valid
        if(user){
            window.location.href = "/Dashboard";
        }
    }
}
```

./src/util/getStudentInfo.js

```
import { createClient } from '@supabase/supabase-js';

const supabaseUrl = process.env.REACT_APP_SUPABASE_URL;
const supabaseKey = process.env.REACT_APP_SUPABASE_KEY;
const supabase = createClient(supabaseUrl, supabaseKey)

// Validate the user is logged in
export default async function getStudentInfo(){
   let token = localStorage.getItem('token');

   // Kick user if toke is not set
   if(!token && window.location.href != "/") {
      window.location.href = "/";
   }

   const { data: { user } } = await supabase
   .from('student')
   .select(`*, address!inner(*)`)
   .eq('studentNumber',localStorage.getItem('studentNumber'));

   // Set Local storage is user data fetched
   if(user){
      localStorage.setItem('user',user);
      return true;
   }else{
      return false;
   }
}
```

./src/util/handleAcademicRecord.js

```
export default function processAcademicData(data){

   /*
   - semester [int] (1 or 2)
   - year [int]
   - moduleCode [text]
   - type [text] (Either assignment_1, assignment_2, assignment_3, cs_test,
supplementary_exam or exam)
   - result [text] (Either pass or fail)
   - markPercentage [int]
   - studentNumber [int] (ALWAYS 9 digits)
```

```
*/

/*
------------------------------------------------------------
OBJECT
------------------------------------------------------------
*/

// Initialize an empty array to store the processed data
const processedData = data.reduce((acc, curr) => {
  // Destructure the current item
  const { module_code, year, semester, type, mark_percentage } = curr;
  // Find the index of the current module in the accumulator
  const moduleIndex = acc.findIndex(module => module.module_code === module_code);

  // If the module is not found in the accumulator
  if (moduleIndex === -1) {
    // Push a new module object into the accumulator
    acc.push({
      module_code,
      years: [{
        year,
        semesters: [{
          semester,
          marks: [{
            type,
            mark_percentage
          }]
        }]
      }]
    });
  } else {
    // If the module is found, find the index of the current year
    const yearIndex = acc[moduleIndex].years.findIndex(y => y.year === year);

    // If the year is not found
    if (yearIndex === -1) {
      // Push a new year object into the module
      acc[moduleIndex].years.push({
        year,
        semesters: [{
          semester,
          marks: [{
```

```
                        type,
                        mark_percentage
                    }]
                }]
            });
        } else {
            // If the year is found, find the index of the current semester
            const semesterIndex = acc[moduleIndex].years[yearIndex].semesters.findIndex(s =>
s.semester === semester);

            // If the semester is not found
            if (semesterIndex === -1) {
                // Push a new semester object into the year
                acc[moduleIndex].years[yearIndex].semesters.push({
                    semester,
                    marks: [{
                        type,
                        mark_percentage
                    }]
                });
            } else {
                // If the semester is found, push the mark into the semester
                acc[moduleIndex].years[yearIndex].semesters[semesterIndex].marks.push({
                    type,
                    mark_percentage
                });
            }
        }
    }

    // Return the accumulator for the next iteration
    return acc;
}, []);

// Iterate over each module in the processed data
processedData.forEach(module => {
    // Iterate over each year in the module
    module.years.forEach(year => {
        // Iterate over each semester in the year
        year.semesters.forEach(semester => {
            // Calculate the total marks for the semester
            const totalMarks = semester.marks.reduce((total, mark) => total +
mark.mark_percentage, 0);
```

```
            // Assign the total marks to the semester
            semester.total = totalMarks;
        });
    });
  });

  // console.log('-----------------------------------------------');
  // console.log(processedData);
  // console.log('-----------------------------------------------');

  return processedData;
}
```

./src/util/logout.js

```
import { createClient } from '@supabase/supabase-js';

const supabaseUrl = process.env.REACT_APP_SUPABASE_URL;
const supabaseKey = process.env.REACT_APP_SUPABASE_KEY;
const supabase = createClient(supabaseUrl, supabaseKey)

// Validate the user is has token
export default async function logOut(){
  let token = localStorage.getItem('token');

  // Login user if toke is not set
  if(token){
    const { error } = await supabase.auth.signOut()
    console.log(error);

    // Login user if JWT is not valid
    if(!error){
      localStorage.removeItem("token");
      window.location.href = "/"
    }
  }else{
    window.location.href = "/"
  }
}
```

./src/islands/Academics_AcademicRecord.jsx

```
import validateUser from './../util/auth';
import {React,useState,useEffect} from 'react';
import { createClient } from '@supabase/supabase-js';
import processAcademicData from '../util/handleAcademicRecord';
import { usePDF } from "react-to-pdf";

export default function Academics_AcademicRecord(props){
  // Ensure user is logged in before showing sensitive data
  validateUser();
  const [showTools, setShowTools] = useState(true);
  const results = props.results
  const { toPDF, targetRef } = usePDF({filename: 'AcademicRecord.pdf'});
console.log(localStorage.getItem("student"))
  const [student,setStudent] = useState(null);
  const [wait,setWait] = useState(true)

  useEffect(() => {
    const fetchSupabaseData = async () => {
     const supabase = createClient(
      process.env.REACT_APP_SUPABASE_URL,
      process.env.REACT_APP_SUPABASE_KEY
     );

     try {
      const { data, error } = await supabase
       .from('student')
       .select(`*`)
       .eq('studentNumber',localStorage.getItem('studentNumber'));
      if (error) {
        setWait(true);
        console.log(error)
      } else {
        setStudent(data[0]);
        setWait(false);
        localStorage.setItem("student",JSON.stringify(data[0]))
      }
     } catch (error) {
      setWait(true);
     }
    };

    if(student === null){
      fetchSupabaseData();
```

```
        }
      }, []);

    return(
        <>
            <div id="record-tools">
                <button onClick={() => {toPDF()}}>
                    <span>
                        <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
fill="currentColor" class="bi bi-download" viewBox="0 0 16 16">
                            <path d="M.5 9.9a.5.5 0 0 1 .5.5v2.5a1 1 0 0 0 1 1h12a1 1 0 0 0 1-1v-2.5a.5.5
0 0 1 1 0v2.5a2 2 0 0 1-2 2H2a2 2 0 0 1-2-2v-2.5a.5.5 0 0 1 .5-.5z"/>
                            <path d="M7.646 11.854a.5.5 0 0 0 .708 0l3-3a.5.5 0 0 0-.708-.708L8.5
10.293V1.5a.5.5 0 0 0-1 0v8.793L5.354 8.146a.5.5 0 1 0-.708.708l3 3z"/>
                        </svg>
                    </span>
                    <span>Download</span>
                </button>
                <button onClick={()=>window.print()}>
                    <span>
                        <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
fill="currentColor" class="bi bi-printer" viewBox="0 0 16 16">
                            <path d="M2.5 8a.5.5 0 1 0 0-1 .5.5 0 0 0 0 1z"/>
                            <path d="M5 1a2 2 0 0 0-2 2v2H2a2 2 0 0 0-2 2v3a2 2 0 0 0 2 2h1v1a2 2 0 0 0
2 2h6a2 2 0 0 0 2-2v-1h1a2 2 0 0 0 2-2V7a2 2 0 0 0-2-2h-1V3a2 2 0 0 0-2-2H5zM4 3a1 1 0 0 1
1-1h6a1 1 0 0 1 1 1v2H4V3zm1 5a2 2 0 0 0-2 2v1H2a1 1 0 0 1-1-1V7a1 1 0 0 1 1-1h12a1 1 0 0
1 1 1v3a1 1 0 0 1-1 1h-1v-1a2 2 0 0 0-2-2H5zm7 2v3a1 1 0 0 1-1 1H5a1 1 0 0 1-1-1v-3a1 1 0 0
1 1-1h6a1 1 0 0 1 1 1z"/>
                        </svg>
                    </span>
                    <span>Print</span>
                </button>
            </div>
            <div id="academicrecord-content" ref={targetRef}>
                <div id="headings">
                    <span id="school">Richfield Graduate Institute of Technology</span>
                    <span>Academic Record</span>
                </div>
                <div id="student-info">
                    <div id="student-number">{localStorage.getItem('studentNumber')}</div>
                    <div id="student-names">
                        <span id="student-fname">{wait ? "Loading..." : `${student.fName}`}</span>
                        <span id="student-mnames"></span>
                        <span id="student-lname">{wait ? "Loading..." : `${student.lName}`}</span>
                    </div>
                    <div id="student-id-number">{wait ? "Loading..." : `${student.idNumber}`}</div>
                    <div id="student-dob">{wait ? "Loading..." : `${student.dateOfBirth}`}</div>
                </div>
                <div id="student-owes-institution">
                    You owe the institution <span id="student-owing-amount">ZAR 7674</span>
                </div>
                {Object.entries(results.reduce((acc, module) => {
                    module.years.forEach((year) => {
                        if (!acc[year.year]) {
                            acc[year.year] = [];
                        }
                        year.semesters.forEach((semester) => {
                            console.log(semester)
                            // ------------------------------------------------
                            // Mark an year mark calculations here
                            let fullPeriodMark = 0;
                            let finalMark = 0;
                            let examMark = 0;
                            for(let i=0; i<semester.marks.length;i++){
                                if(semester.marks[i].type === "assignment_3"){
                                    fullPeriodMark += semester.marks[i].mark_percentage * 0.34;
                                }else if(semester.marks[i].type !== "exam"){
                                    fullPeriodMark += semester.marks[i].mark_percentage * 0.33;
                                }else{
                                    examMark = semester.marks[i].mark_percentage;
                                }
                            }

                            finalMark = (fullPeriodMark * 0.40) + (examMark * 0.60);
                            //------------------------------------------------
                            acc[year.year].push({
                                module_code: module.module_code,
                                semester: semester.semester,
                                full_period_mark:fullPeriodMark.toFixed(1),
                                total: finalMark.toFixed(1)
                            });
                        });
                    });
                    return acc;
                }, {})).map(([year, modules], index) => {
                    return (
                        <div key={index} className="record">
                            <div className="year">
                                <div className="represented-year">
                                    {year}
                                </div>
                            </div>
                            <div className="qualification">
                                RBSIT - BSC IN INFORMATION TECHNOLOGY
                            </div>
                            <div className="records">
                                {modules.map((module, moduleIndex) => {
                                    return (
                                        <div key={moduleIndex} className={`record-item ${module.total >= 75
    ? 'pass-distinction-border' : module.total >= 60 ? 'pass-safe-border' : module.total >= 50 ?
    'pass-warning-border' : module.total >= 40 ? 'pass-border' : 'fail-border'}`}>
                                            <div className="subject"><span>SUBJECT:</span>
                                                <span className="code">
                                                        {module.module_code}
                                                </span>
                                                <span className="title">
                                                    {module.module_code}
                                                </span>
                                            </div>
                                            <div className="academic-period">
                                                SEMESTER {module.semester} (JAN - JUN)
                                            </div>
                                            <div className="marks">
                                                <table>
                                                    <thead>
                                                        <tr>
                                                            <th>Year Mark</th>
                                                            <th>Final Mark</th>
                                                            <th>Result</th>
                                                        </tr>
                                                    </thead>
                                                    <tbody>
                                                        <tr>
                                                            <td className="year-mark">
                                                                {module.full_period_mark}
                                                            </td>
                                                            <td className="final-mark">
                                                                {module.total}
                                                            </td>
```

```
                                                            <td className={`result ${module.total >= 75 ? 'pass-distinction-text' : module.total >= 60 ? 'pass-safe-text' : module.total >= 50 ? 'pass-warning-text' : module.total >= 40 ? 'pass-text' : 'fail-text'}`}>
                                                                {module.total >= 40 ? 'PASS' : 'FAIL'}
                                                            </td>
                                                        </tr>
                                                    </tbody>
                                                </table>
                                            </div>
                                        </div>
                                    );
                                })}
                            </div>
                        </div>
                    );
                })}
                {/* RECORD END */}
            </div>
        </>
    );
}
```

./src/islands/Academics_ExamResults.jsx

```
import { useState } from "react";
import ExamResultCard from "../components/ExamResultCard";
import validateUser from './../util/auth';
import { usePDF } from "react-to-pdf";

export default function Academics_ExamResults(props){
    validateUser();
    const { results: examData } = props;
    const [showResults, setShowResults] = useState(false);
    const [year, setYear] = useState(null);
    const { toPDF, targetRef } = usePDF({filename: 'ExamResults.pdf'});
    const student = JSON.parse(localStorage.getItem('student'));
    const loadResults = (event) => {
        const element = event.target;
        const yearToSet = element.classList.contains('exam-item') ?
            element.getAttribute("data-year") :
            element.parentElement.getAttribute("data-year");
        setYear(yearToSet);
        setShowResults(true);
    };

    const loadMenu = () => {
        setShowResults(false);
    };
    console.log(year ? examData : "Not set");

    const results = (
        <div id="exam-results">
            <div id="go-back">
                <button onClick={loadMenu}>
                    <svg xmlns="http://www.w3.org/2000/svg" width="25" height="25"
fill="currentColor" className="bi bi-arrow-left-short" viewBox="0 0 16 16">
                        <path fillRule="evenodd" d="M12 8a.5.5 0 0 1-.5.5H5.707l2.147 2.146a.5.5 0 0
1-.708.708l-3-3a.5.5 0 0 1 0-.708l3-3a.5.5 0 1 1 .708.708L5.707 7.5H11.5a.5.5 0 0 1 .5.5z"/>
                    </svg>
                    <span>Go Back</span>
                </button>
            </div>
            <div id="record-tools">
                <button onClick={() => {toPDF()}}>
                    <span>
                        <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
fill="currentColor" class="bi bi-download" viewBox="0 0 16 16">
                            <path d="M.5 9.9a.5.5 0 0 1 .5.5v2.5a1 1 0 0 0 1 1h12a1 1 0 0 0 1-1v-2.5a.5.5
0 0 1 1 0v2.5a2 2 0 0 1-2 2H2a2 2 0 0 1-2-2v-2.5a.5.5 0 0 1 .5-.5z"/>
                            <path d="M7.646 11.854a.5.5 0 0 0 .708 0l3-3a.5.5 0 0 0-.708-.708L8.5
10.293V1.5a.5.5 0 0 0-1 0v8.793L5.354 8.146a.5.5 0 1 0-.708.708l3 3z"/>
                        </svg>
                    </span>
                    <span>Download</span>
                </button>
                <button onClick={()=>window.print()}>
                    <span>
                        <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
fill="currentColor" class="bi bi-printer" viewBox="0 0 16 16">
                            <path d="M2.5 8a.5.5 0 1 0 0-1 .5.5 0 0 0 0 1z"/>
                            <path d="M5 1a2 2 0 0 0-2 2v2H2a2 2 0 0 0-2 2v3a2 2 0 0 0 2 2h1v1a2 2 0 0 0
2 2h6a2 2 0 0 0 2-2v-1h1a2 2 0 0 0 2-2V7a2 2 0 0 0-2-2h-1V3a2 2 0 0 0-2-2H5zM4 3a1 1 0 0 1
1-1h6a1 1 0 0 1 1 1v2H4V3zm1 5a2 2 0 0 0-2 2v1H2a1 1 0 0 1-1-1V7a1 1 0 0 1 1-1h12a1 1 0 0
1 1 1v3a1 1 0 0 1-1 1h-1v-1a2 2 0 0 0-2-2H5zm7 2v3a1 1 0 0 1-1 1H5a1 1 0 0 1-1-1v-3a1 1 0 0
```

```
1 1-1h6a1 1 0 0 1 1 1z"/>
                    </svg>
                  </span>
                  <span>Print</span>
                </button>
              </div>
              <div id="exam-details" ref={targetRef}>
                {examData[year] ?

                Object.entries(examData[year]['semesters']).map(([semester, semesterData], index)
=> (
                  Object.entries(semesterData).map(([subjectKey, subjectData], index) => {
                    console.log(subjectData)
                    /*
                    subjectData expected value:
                      [
                        {
                          "id": 29,
                          "type": "assignment_1",
                          "result": "pass",
                          "mark_percentage": 85,
                          "created_at": "2023-11-14T19:33:59.754483+00:00"
                        },
                        {
                          "id": 30,
                          "type": "assignment_2",
                          "result": "fail",
                          "mark_percentage": 40,
                          "created_at": "2023-11-14T19:33:59.754483+00:00"
                        },
                        {
                          "id": 31,
                          "type": "assignment_3",
                          "result": "pass",
                          "mark_percentage": 75,
                          "created_at": "2023-11-14T19:33:59.754483+00:00"
                        },
                        {
                          "id": 32,
                          "type": "exam",
                          "result": "pass",
                          "mark_percentage": 78,
                          "created_at": "2023-11-14T19:33:59.754483+00:00"
```

```
        }
    ]
*/

let halfPeriodMark = 0;
let fullperiodMark = 0;
let finalMark = 0;
let examMark = 0;
let result = null;

// Loop to iterate through subjectData
subjectData.forEach((data, index) => {
    // Space for calculations
    // TODO: Add your calculations here
    console.log("data is");
    console.log(data);

    if(data.type === "assignment_3"){
        fullperiodMark += data.mark_percentage * 0.34;
    }else if(data.type != "exam"){
        fullperiodMark += data.mark_percentage * 0.33;
    }else{
        examMark = data.mark_percentage;
    }
});

finalMark = (fullperiodMark * 0.40) + (examMark * 0.60);

if(finalMark > 49 && finalMark < 56){
    result = "pass-warning"
}else if(finalMark > 55 && finalMark < 60){
    result = "pass"
}
else if(finalMark > 59 && finalMark < 75){
    result = "pass-safe"
}
else if(finalMark > 74){
    result = "pass-distinction"
}
else if(finalMark < 50){
    result = "fail"
}
```

```
                return (
                    <ExamResultCard
                        key={subjectKey}
                        result={result}
                        examMonth={6}
                        subject={subjectKey}
                        halfPeriodMark={halfPeriodMark.toFixed(1)}
                        fullPeriodMark={fullperiodMark.toFixed(1)}
                        finalMark={finalMark}
                    />
                )
            })
        )) : "Nothing Found"

        }
      </div>
    </div>
  );

  const options = (
    <div id="exam-year-qualification-options">
      {Object.keys(examData).map((examYear, index) => (
        <div key={index} className="exam-item" onClick={loadResults} data-
year={examYear}>
          <div className="exam-year">{examYear}</div>
          <div className="exam-qualification">Bsc In Information Technology</div>
        </div>
      ))}
    </div>
  );

  return(
    <div id="examResults">
      <div id="student-details-exam-year">
        <span>{`${student.fName} ${student.lName}`}</span>
        <span>{localStorage.getItem('studentNumber')}</span>
      </div>
      {showResults ? results : options}
    </div>
  );
}
```

./src/islands/Academics_ProgressReport.jsx

```
import validateUser from './../util/auth';
import DataManipulation from './../util/dataManipulation';
import { usePDF } from "react-to-pdf";

export default function Academics_ProgressReport(props){
    validateUser();
    const dataManipulation = new DataManipulation();
    const results = props.results;
    const { toPDF, targetRef } = usePDF({filename: 'ProgressReport.pdf'});
    const student = JSON.parse(localStorage.getItem('student'));

    // console.log(results)
    return(
        <>
            <div id="report-tools">
                <button onClick={() => {toPDF()}}>
                    <span>
                        <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" 
fill="currentColor" class="bi bi-download" viewBox="0 0 16 16">
                            <path d="M.5 9.9a.5.5 0 0 1 .5.5v2.5a1 1 0 0 0 1 1h12a1 1 0 0 0 1-1v-2.5a.5.5 
0 0 1 1 0v2.5a2 2 0 0 1-2 2H2a2 2 0 0 1-2-2v-2.5a.5.5 0 0 1 .5-.5z"/>
                            <path d="M7.646 11.854a.5.5 0 0 0 .708 0l3-3a.5.5 0 0 0-.708-.708L8.5 
10.293V1.5a.5.5 0 0 0-1 0v8.793L5.354 8.146a.5.5 0 1 0-.708.708l3 3z"/>
                        </svg>
                    </span>
                    <span>Download</span>
                </button>
                <button onClick={()=>window.print()}>
                    <span>
                        <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" 
fill="currentColor" class="bi bi-printer" viewBox="0 0 16 16">
                            <path d="M2.5 8a.5.5 0 1 0 0-1 .5.5 0 0 0 0 1z"/>
                            <path d="M5 1a2 2 0 0 0-2 2v2H2a2 2 0 0 0-2 2v3a2 2 0 0 0 2 2h1v1a2 2 0 0 0 
2 2h6a2 2 0 0 0 2-2v-1h1a2 2 0 0 0 2-2V7a2 2 0 0 0-2-2h-1V3a2 2 0 0 0-2-2H5zM4 3a1 1 0 0 1 
1-1h6a1 1 0 0 1 1 1v2H4V3zm1 5a2 2 0 0 0-2 2v1H2a1 1 0 0 1-1-1V7a1 1 0 0 1 1-1h12a1 1 0 0 
1 1 1v3a1 1 0 0 1-1 1h-1v-1a2 2 0 0 0-2-2H5zm7 2v3a1 1 0 0 1-1 1H5a1 1 0 0 1-1-1v-3a1 1 0 0 
1 1-1h6a1 1 0 0 1 1 1z"/>
                        </svg>
                    </span>
                    <span>Print</span>
                </button>
```

```
      </div>
      <div id="progressreport-content">
        <div id="header">
          <h1>Progress Report</h1>
        </div>
        <div id="student-details">
          <div id="student-names">
            <span id="student-fname">{student.fName}</span>
            <span id="student-mnames"></span>
            <span id="student-lname">{student.lName}</span>
          </div>
          <div id="student-number">{localStorage.getItem('studentNumber')}</div>
        </div>
        <div id="progress-reports">
          {results.map((module, index) => {
            return module.years.map((year, yearIndex) => {
              return year.semesters.map((semester, semesterIndex) => {
                let fullPeriodMark = 0;

                return (
                  <div key={`${index}-${yearIndex}-${semesterIndex}`}
className="report">
                    <div className="report-year">{year.year}</div>
                    {/* <div className="report-
qualification">{module.module_code}</div> */}
                    <div className="report-qualification">BSC IT</div>
                    <div className="report-content">
                      <div className="report-subject">
                        <span>SUBJECT: {module.module_code}</span>
                      </div>
                      <table className='grade'>
                        <tbody>
                        {semester.marks.map((mark, markIndex) => {
                          if(mark.type === "assignment_3"){
                            fullPeriodMark += mark.mark_percentage * 0.34;
                          }else if(mark.type != "exam"){
                            fullPeriodMark += mark.mark_percentage * 0.33;
                          }

                          return (
                            <td key={markIndex} className="report-grade">
                              <tr className={`progress-heading ${mark.type === "exam"
? 'exam':null}`}>{dataManipulation.formatString(mark.type)}</tr>
```

```
                                                            <tr>{mark.mark_percentage}</tr>
                                                        </td>
                                                    );
                                                })}
                                                </tbody>
                                            </table>
                                            <div className='breaker'></div>
                                            <div className="report-detail">
                                                <div className="academic-period">Semester
{semester.semester}</div>
                                                <div className="full-period-mark">Full Period Mark:
{fullPeriodMark.toFixed(1)}</div>
                                            </div>
                                            <div className="exam-admission">
                                                Exam Admission: {fullPeriodMark.toFixed(1) > 49 ? "Y":"N"}
                                            </div>
                                        </div>
                                    </div>
                                );
                            });
                        });
                    })}
                </div>
            </div>
        </>
    );
}
```

./src/util/handleExamResults.js

```
export default function handleExamData(data){
    // Create an empty object to store the sorted data
    let sortedData = {};

    // Loop through each item in the data array
    data.forEach(item => {
        // If the year does not exist in the sortedData object, add it
        if (!sortedData[item.year]) {
            sortedData[item.year] = {
                student_number: item.student_number,
                semesters: {}
            };
        }

        // If the semester does not exist in the year, add it
        if (!sortedData[item.year].semesters[item.semester]) {
            sortedData[item.year].semesters[item.semester] = {};
        }

        // If the module does not exist in the semester, add it
        if (!sortedData[item.year].semesters[item.semester][item.module_code]) {
            sortedData[item.year].semesters[item.semester][item.module_code] = [];
        }

        // Add the module data to the module array
        sortedData[item.year].semesters[item.semester][item.module_code].push({
            id: item.id,
            type: item.type,
            result: item.result,
            mark_percentage: item.mark_percentage,
            created_at: item.created_at
        });
    });

    // Return the sorted data
    return sortedData;
}
```

5.3 Testing
During the development of the systems, regular unit tests were performed and peer reviewed. As we prioritized certain components to develop first, the same components were also tested first to ensure that they are responsive and styled appropriately.

During development, a keen eye was also kept for any compilation and package errors that may occur with node. Especially when re-running the project. All tests were done on localhost before being pushed to the production server.

5.4 System Testing
The main goal of the project is to revamp the front-end of the iEnabler system so that it keeps up with modern web interfaces and is pleasant to view on all devices. With this in mind, we ran tests to ensure that this goal is met on all pages, islands and components.

5.5 Test Case

- Route to each page in mobile view
- Switch from mobile view to:
    - Tablet View
    - Desktop View
  - On each view, assess if the layout has been adjusted appropriately so that content is viewable and pleasantly presented with all functionalities still working. For example, buttons should still be clickable to route to another page.
- Reverse the order of tests. Now we start from the desktop view and switch down into mobile view, running the same tests as above.

While running these tests we also lookout for any obvious bugs and stylistic errors that may occur. Should we find any errors, they are reported on our group chat and a fix is then worked on.

5.6 Evaluation of the testing results
After running all tests, we have concluded that the code does work as intended, albeit with limited functionality and 'dummy data'. The styling is maintained through all views of the application. There are 0 runtime and compile-time errors. The application runs smoothly in all device views and the modular design allows for new components to be added with ease.

5.7 Installation
There are two ways to view this app. The first is to simple go to the URL of the demo:

https://production.d3nl9seejt63m7.amplifyapp.com/

Alternatively, you could install the application on your local machine by:

- Pulling the code from production branch of the GitHub repository
- Routing into the folder of the cloned repo, via your terminal like so, 'cd PATH_TO_CLONED_REPO'
- Once in your folder from the terminal you may run 'npm install' to install any NodeJS Packages that were used in this project.
- Now you may run 'npm start' to run this project on the default browser your local machine

Note: if you want to view the code of this project from VS Code, you may type in your terminal:

'code .'

Doing this from the root folder of the cloned repository will open VS Code with this folder already selected.

6. System Security measures (Implementation of security for the project developed)

The database chosen (Supabase) regularly creates backups (On the premium version) of tables in the database.

Manual backups are also an option as well as CSV exports to save locally as well as on other back devices, such as cloud storage or a hard drive. Branch protection can also be implemented on our version control system of choice, IE Git via GitHub.

Additionally, our hosting service, AWS, provides containerized weekly backups of our system that can be accessed from our dashboard.

6.1 Data Security

To ensure data security this project will follow OOP principles such as encapsulation, Inheritance and Abstraction. We will also be implementing access control where possible within the code. Additionally, although the database is only for test purposes, we will be using Row-Level-Security to further enhance security and limit access on a database level.

Row-Level-Security is a feature in Supabase that allows specific access (CREATE, READ, UPDATE and/or DELETE) under specific conditions, which may be:

- Only authenticated users can access
- Only admin users can access
- Only users with two factor authentication can access

For the purposes of this application, we will only allow read and update access to authenticates users (IE, users who have logged in)

6.2 Privacy

In order to ensure that the users' data is kept private to them and is not accessible to others, this program will make use of access tokens in order to reduce the risk of somebody being able to access the user's data without their consent. The access token would last approximately 1 hour. In addition to this, any items that are considered to be "Sensitive Information" will require a token check before being displayed. Such items may include final results, students' details, fee details, etc.

To implement the protection of sensitive information, we used authentication checks on each page before the content is loaded to make sure that the user is logged in before loading any data. If the user is not logged in, then they are kicked from the system and routed back to the login screen.

7. Cost Estimation of the Project along with Cost Estimation Model

7.1 Cost Estimation

Potential	Description
Personnel Costs	- Project Manager: Manages overall project, coordinates efforts. - UI/UX Designers: Designs enhanced UI. - Frontend Developers: Implements UI enhancements.
Technology and Tools	Costs associated with acquiring or upgrading necessary tools and technologies for frontend development and design.
Training	Costs related to training sessions for team members on new technologies, design principles, and project requirements.
Usability Testing	Costs associated with conducting usability tests, including participant recruitment, testing tools, and analysis.
Documentation	Costs related to the creation and maintenance of project documentation, including user manuals and technical documentation.
Contingency	A percentage of the total project cost allocated as a contingency fund to address unforeseen issues or changes in project scope.
Change Management	Costs associated with implementing change management strategies, including user training and communication efforts.
Security Measures	Costs related to security assessments, implementation of security measures, and any necessary tools or services.
Infrastructure	Costs associated with server hosting, domain management, and other infrastructure needs.
Miscellaneous	Any other miscellaneous costs that may arise during the project.

7.2 Cost Estimation Model

The Parametric Estimation model within the Cost Estimation framework relies on statistical relationships extracted from historical data and various variables to project costs for this project. The equation for this estimation model is Cost Estimate = Parameter Value × Quantity.

Based on the same personnel completing all the various tasks on this project, we calculate the cost as follows:

= Average Cost per Hour × Estimated Hours for Personnel × Number of Personnel

R148.67 (averages cost per hour) x 24 (working 6 hours a week for 4 weeks) x 5 (number of personal)

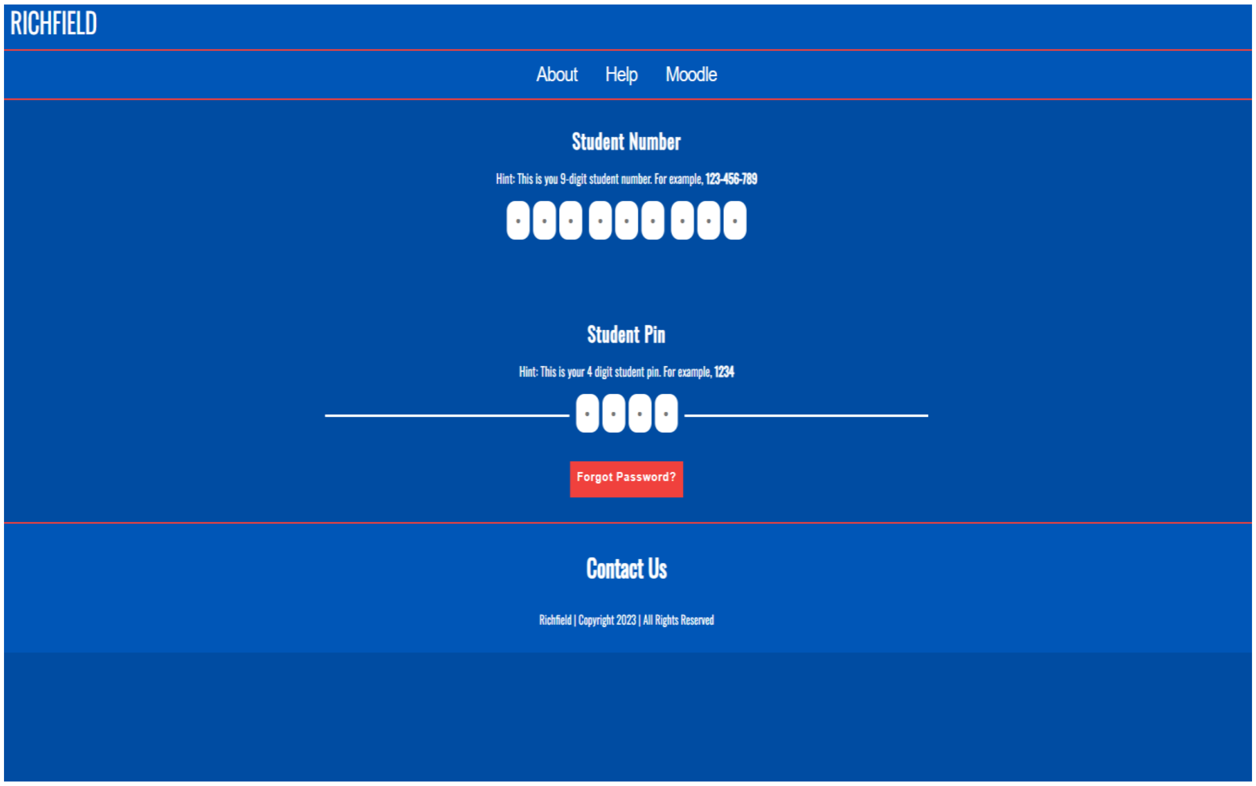
= R17 840.40 x 3 (months)

Total cost:

= R53 521.20

(McGarvie, 2023)

8. Reports (sample layouts should be placed)



Student Number

Hint: This is you 9-digit student number. For example, 123-456-789

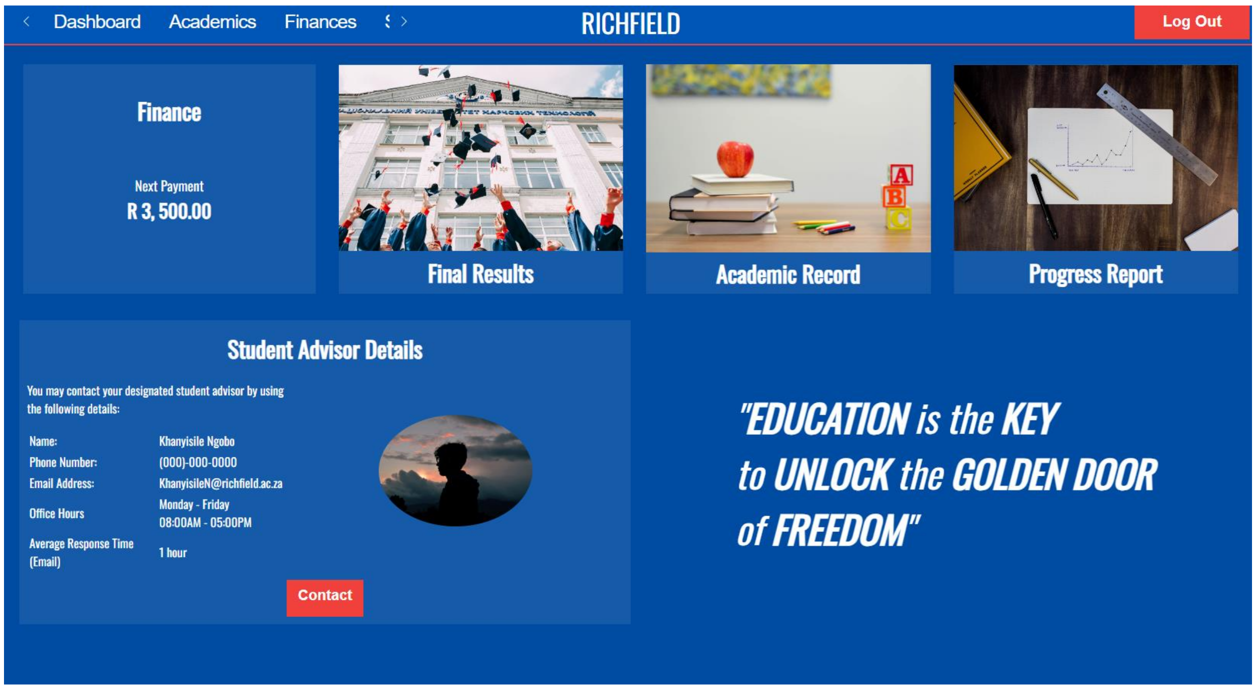
Student Pin

Hint: This is your 4 digit student pin. For example, 1234

Contact Us

Richfield | Copyright 2023 | All Rights Reserved

Figure 4: Updated log in page



Finance

Next Payment

R 3, 500.00

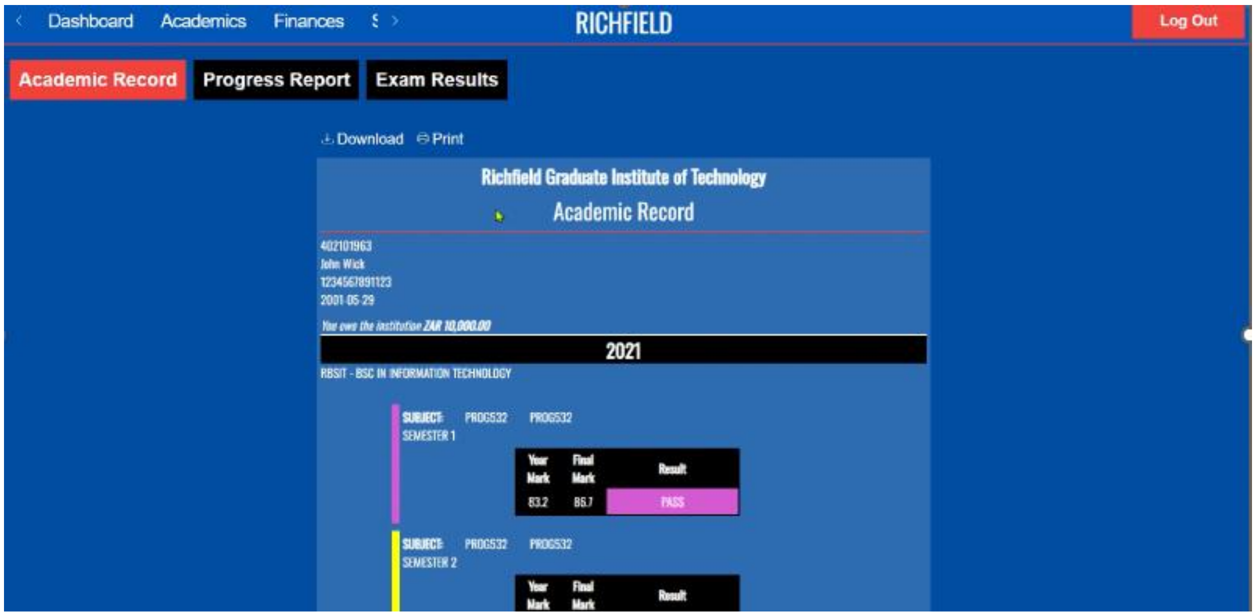
Student Advisor Details

You may contact your designated student advisor by using the following details:

Name:	Khanyisile Ngobo
Phone Number:	(000)-000-0000
Email Address:	KhanyisileN@richfield.ac.za
Office Hours	Monday - Friday 08:00AM - 05:00PM
Average Response Time (Email)	1 hour

"EDUCATION is the KEY to UNLOCK the GOLDEN DOOR of FREEDOM"

Figure 5: Enhanced main dashboard



Richfield Graduate Institute of Technology

Academic Record

402101963
John Wick
1234567891123
2001-05-29

*You owe the institution **ZAR 10,000.00***

2021

RBSIT - BSC IN INFORMATION TECHNOLOGY

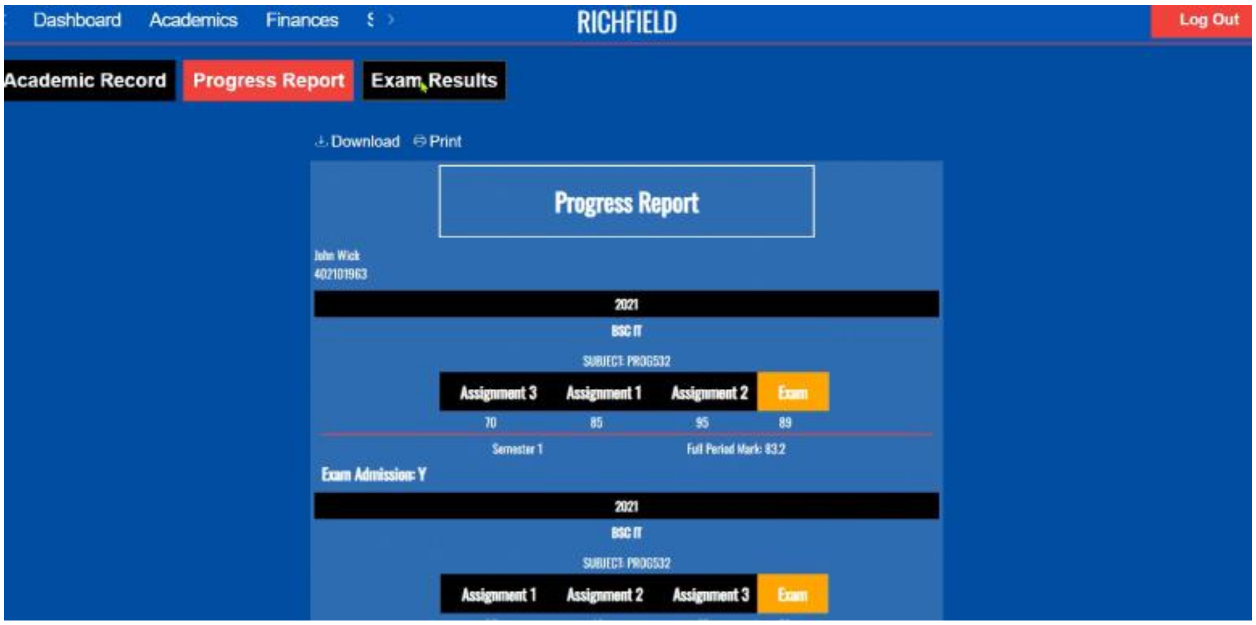
SUBJECT: PROG532 PROG532
SEMESTER 1

Year Mark	Final Mark	Result
83.2	86.7	PASS

SUBJECT: PROG532 PROG532
SEMESTER 2

Year Mark	Final Mark	Result

Figure 6: Enhanced academics page with download and print functionality



Progress Report

John Wick
402101963

2021

BSC IT

SUBJECT: PROG532

Assignment 3	Assignment 1	Assignment 2	Exam
70	85	95	89

Semester 1 Full Period Mark: 83.2

Exam Admission: Y

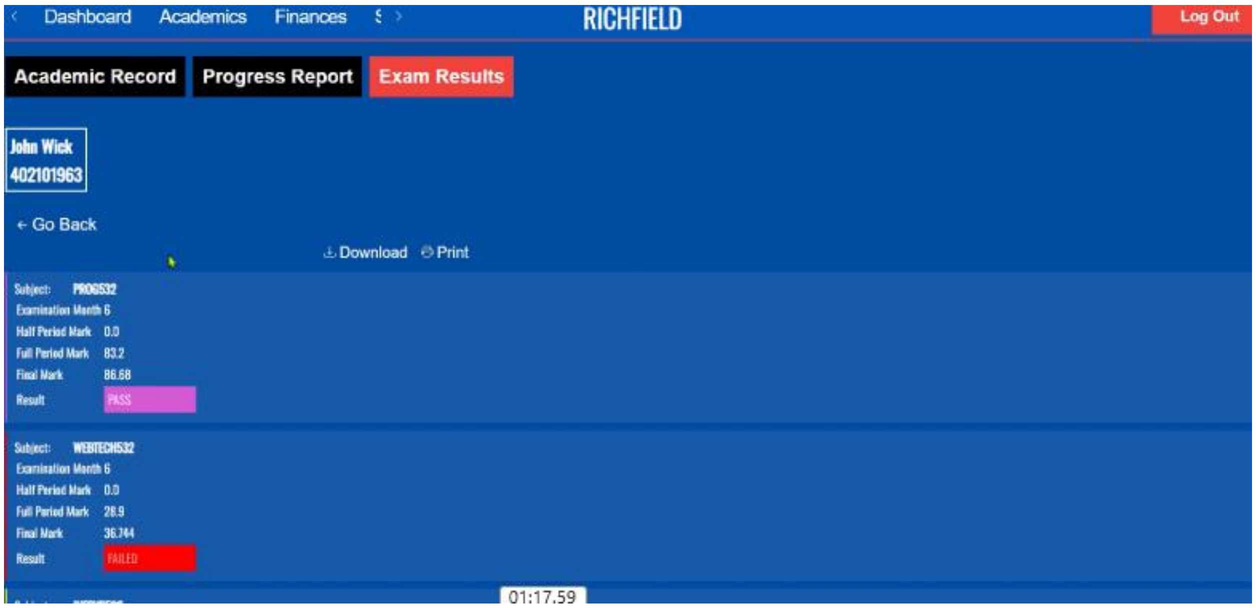
2021

BSC IT

SUBJECT: PROG532

Assignment 1	Assignment 2	Assignment 3	Exam

Figure 7: Updated Progress Report page

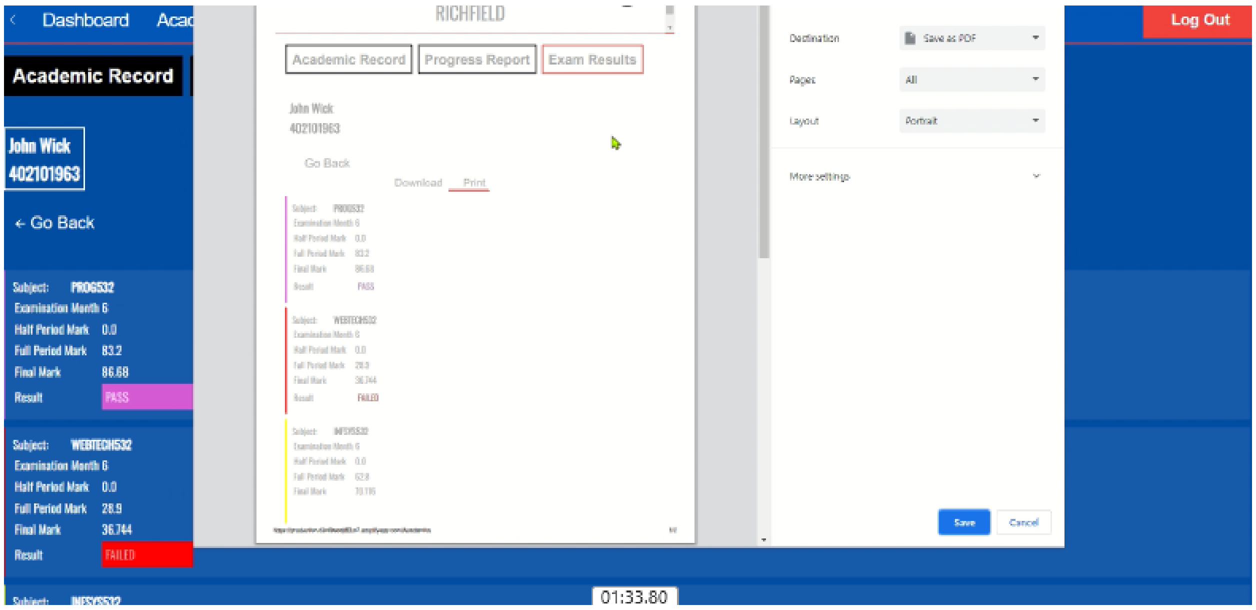


John Wick
402101963

Subject:	PROG532
Examination Month	6
Half Period Mark	0.0
Full Period Mark	83.2
Final Mark	86.68
Result	PASS

Subject:	WEBTECH532
Examination Month	6
Half Period Mark	0.0
Full Period Mark	28.9
Final Mark	36.744
Result	FAILED

Figure 8: Updated Exam results page



Destination: Save as PDF
Pages: All
Layout: Portrait

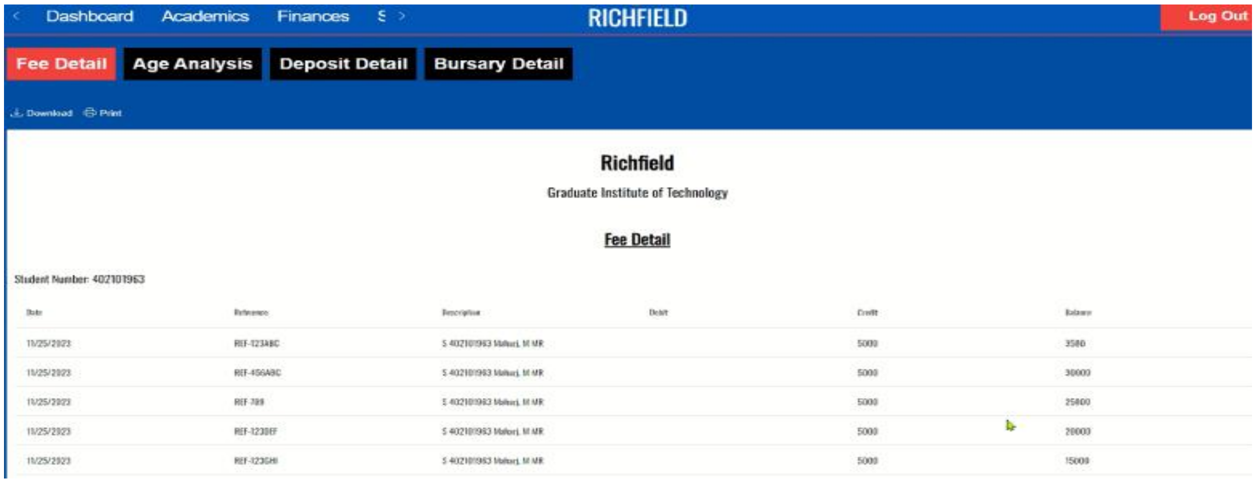
John Wick
402101963

Subject:	PROG532
Examination Month	6
Half Period Mark	0.0
Full Period Mark	83.2
Final Mark	86.68
Result	PASS

Subject:	WEBTECH532
Examination Month	6
Half Period Mark	0.0
Full Period Mark	28.9
Final Mark	36.744
Result	FAILED

Subject:	INFSYS532
Examination Month	6
Half Period Mark	0.0
Full Period Mark	62.8
Final Mark	70.116

Figure 9: Print functionality



Richfield

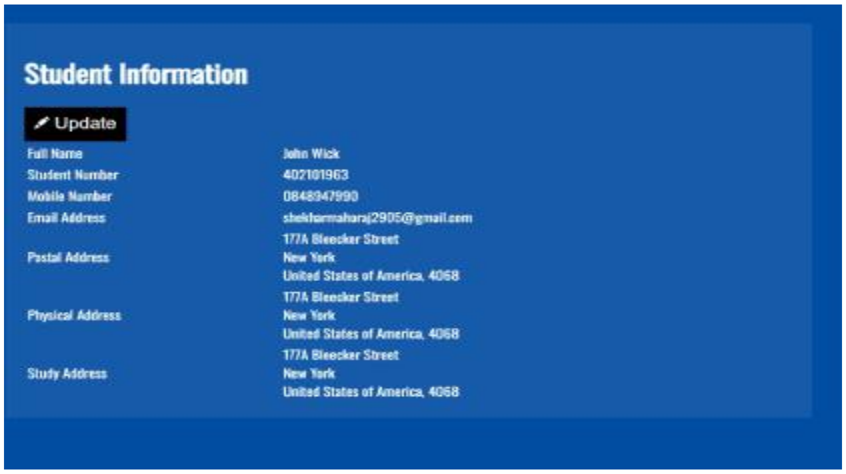
Graduate Institute of Technology

Fee Detail

Student Number: 402101963

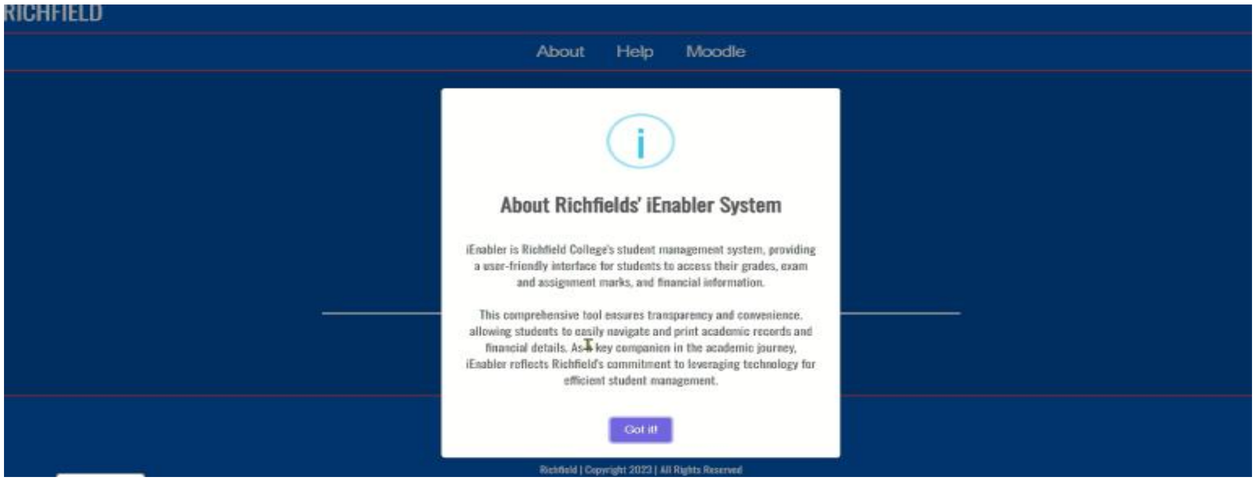
Date	Reference	Description	Debit	Credit	Balance
11/25/2023	REF-123ABC	S 402101963 Mahari, M MR		5000	3580
11/25/2023	REF-456ABC	S 402101963 Mahari, M MR		5000	30000
11/25/2023	REF-789	S 402101963 Mahari, M MR		5000	25800
11/25/2023	REF-123DEF	S 402101963 Mahari, M MR		5000	20000
11/25/2023	REF-123GHI	S 402101963 Mahari, M MR		5000	15000

Figure 10: Finance tabs updated



Full Name	John Wick
Student Number	402101963
Mobile Number	0848947990
Email Address	shekharmaharaj2905@gmail.com
Postal Address	177A Bleecker Street New York United States of America, 4068
Physical Address	177A Bleecker Street New York United States of America, 4068
Study Address	177A Bleecker Street New York United States of America, 4068

Figure 11: Student info with update function



About Richfields' iEnabler System

iEnabler is Richfield College's student management system, providing a user-friendly interface for students to access their grades, exam and assignment marks, and financial information.

This comprehensive tool ensures transparency and convenience, allowing students to easily navigate and print academic records and financial details. As a key companion in the academic journey, iEnabler reflects Richfield's commitment to leveraging technology for efficient student management.

Figure 12: Dynamic buttons with additional information

9. Future scope and further enhancement of the Project

Addressing the current shortcomings in the user-friendliness of the iEnabler student management program's frontend sets the stage for prospective improvements. Building on completed upgrades to layout, login interface, and functionality, there are various avenues for future development. These include conducting user feedback sessions and usability testing to refine the user interface continually. Future enhancements involve introducing personalization features for user-centric experiences, exploring advanced navigation, ensuring mobile responsiveness, and integrating with other educational systems. Additional improvements encompass data visualization tools, enhanced communication features, accessibility enhancements, gamification elements, and the integration of artificial intelligence for smarter functionalities. A focus on continuous training and support ensures users are adept at leveraging the evolved features. Collectively, these enhancements aim to transform iEnabler into a sophisticated, user-friendly, and feature-rich platform, meeting the diverse needs of students, educators, and administrators in the educational landscape.

Bibliography

Exner, K. (2023, November 28). *The Complete Guide To Collecting Meaningful User Feedback*. Retrieved from The Product Manager: https://theproductmanager.com/topics/user-feedback/

McGarvie, E. (2023, May 15). *Software Developer Salary in South Africa*. Retrieved from We Are Developers: https://www.wearedevelopers.com/magazine/software-developer-salary-in-south-africa