EduBlock

HSGamer

11/8/22

Table of contents

# Preface

This is the final report of EduBlock

# 1. Acknowledgement

# 2. Definition and Acronyms

# 3. Project Introduction

## 3.1 Overview

### 3.1.1 Project Information

### 3.1.2 Project Team

## 3.2 Background

## 3.3 Existing Systems

## 3.4 Business Opportunity

## 3.5 Software Product Vision

## 3.6 Project Scope & Limitations

# 4. Project Management Plan

## 4.1 Overview

### 4.1.1 WBS & Estimation

### 4.1.2 Project Objectives

### 4.1.3 Project Risks

## 4.2 Management Approach

### 4.2.1 Project Process

### 4.2.2 Quality Management

### 4.2.3 Training Plan

| Area | Participants | When, Duration | Waiver Criteria |
| --- | --- | --- | --- |
| Blockchain | Project Team | 20/07/2022, 2 weeks | Mandatory |
| Hyperledger Fabric | Project Team | 01/10/2022, 2 weeks | Mandatory for Backend Team |
| Java, Javalin | Project Team | 01/09/2022, 1 weeks | Mandatory |

## 4.3 Master Schedule

## 4.4 Project Organization

## 4.5 Project Communication

### 4.5.1 Communication Plan

| Item | Target | Purpose | When | Type |
| --- | --- | --- | --- | --- |
| Discord | Project Team | Review meeting & Status report | Monday, Friday | Voice, Remote |
| Google Meeting | Project Team & Supervisor | Review meeting, Sprint revision & Closeup | Wednesday | Voice, Remote |
| Messenger | Project Team & Supervisor | Meeting planning, Q&A & Status report | Everyday | Text, Remote |
| FU Library | Project Team | Pair programming & Code review | Planned | Offline |

### 4.5.2 External Interfaces

## 4.6 Configuration Management

### 4.6.1 Tools & Infrastructures

#### 4.6.1.1 Common

| Type | Tool |
| --- | --- |
| Version Control | Git, GitHub |
| UML | PlantUML, Graphviz |
| Deployment | Docker |
| Project Management | Quarto, GitHub Projects |

### 4.6.2 Backend

| Type | Tool |
| --- | --- |
| Programming Language | Java |
| Library | Javalin, Fabric SDKs, HSCore, Guava, Genson, EvalEx, JWT |
| Compiler | JDK, Lombok |
| UI | TinyLog, JLine, Fabric CLI |
| DBMS | H2, Hibernate, Minifabric |
| IDE / Editor | IntelliJ IDEA, VSCode |

### 4.6.3 Frontend

| Type | Tool |
| --- | --- |
| Programming Language | TypeScript |
| Library | React |
| Compiler | Node |
| UI | React |
| IDE / Editor | VSCode |

### 4.6.4 Document Management

We use Quarto to build documents from Markdown files and use GitHub to manage the files and their changes. A participant will create a new branch to edit the files, create pull requests and wait for the project manager to review the changes and merge to the main branch. Then, it’ll be built in three outputs: a website using GitHub Pages for visualization, A PDF document & A MS-Word document.

### 4.6.5 Source Code Management

We manage the source code by using GitHub. Endpoints of the project will be upload into separated repositories. Once the code is changed, the participant will create a new branch, create a relevant pull request, and wait for code owners to review and merge to the main branch.

# 5. Software Requirement Specification

## 5.1 Overall Description

### 5.1.1 Product Overview

This is the software requirement specification for the project “EduBlock”. EduBlock is an web-application that will help the school to manage their student’s records, more specifically, the students and teachers can reduce paper’s work to manage their records. Lately, the school has been using paper to manage their student’s record, which is not efficient and not environmental friendly. EduBlock will help the school to manage their student’s records in a more efficient way, although there are some other 3rd party applications that can help school to keep their student’s records, but it is not really efficient and safe, our application use blockchain technology to make sure the data is safe and secure. Every step of the process that need to be work with the records will be tracked by EduBlock, so the school can easily track the data and make sure the data is not being tampered.

### 5.1.2 Business Rules

* The application will be used by the students, teachers, staff and admin.

## 5.2 User Requirements

* The Academic record management web-app has four active actors: Student, Teacher, Staff and Administrator.
* Students can view their academic record.
* Teachers can manage their class and view their students’ academic record.
* Staff can manage the classroom and view the academic record of the students, assign or delete teacher from the class, assign student to class, create new class.

## 5.3 Functional Requirements

### 5.3.1 System Functional Overview

The system is designed to provide a platform for students to view their academic records, teachers to view their students’ academic records, staff to view the academic records of students, create new class.

### 5.3.2 **Admin Features**

#### 5.3.2.1 UC-1 Admin Login

* **Description:** Admin can login with their username and password.
* **Actors:** Admin.
* **Preconditions:** Admin has an account.
* **Postconditions:** Admin can access the system.
* **Flow of Events:**
  + Admin go to EduBlock.
  + Admin enters their username and password.
  + System verifies the username and password.
  + System displays the dashboard.
* **Exceptions:**
  + If the username or password is incorrect, the system will display an error message.

#### 5.3.2.2 UC-2 Admin view list of accounts

* **Description:** Admin can view list of all accounts.
* **Actors:** Admin
* **Preconditions:** Admin is logged in.
* **Postconditions:** System show list of all accounts.
* **Flow of Events:**
  + Admin go to EduBlock.
  + Admin login with username and password.
  + Admin click on “Account”.
  + System show list of all accounts.
* **Alternate Flow:**
  + System displays notification “No account found” if there is no account.
* **Exception:**
  + System displays notification “No account found” if there is no account.

#### 5.3.2.3 UC-3 Admin view account details

* **Description:** Admin can view account details.
* **Actors:** Admin
* **Preconditions:** Admin is logged in.
* **Postconditions:** System show account details.
* **Flow of Events:**
  + Admin go to EduBlock.
  + Admin login with username and password.
  + Admin click on “Account”.
  + Admin click on “Details” (human icon) on actions column.
  + System show account details.
* **Alternate Flow:**
  + System displays notification “No account found” if there is no account.
* **Exception:**
  + System displays notification “No account found” if there is no account.

#### 5.3.2.4 UC-4 Admin create (multiple) account

* **Description:** Admin can create (multiple) account for each role such as staff, student, teacher.
* **Actors:** Admin
* **Preconditions:** Admin is logged in.
* **Postconditions:** System create (multiple) account.
* **Flow of Events:**
  + Admin go to EduBlock.
  + Admin login with username and password.
  + Admin click on “Account”.
  + Admin click on “Create” button.
  + Admin fill in the form.
  + Admin can click “Add Account” button to add more account.
  + Admin click on “Create” button.
  + System create account(s).
* **Alternate Flow:**
  + System displays notification if the form is not filled correctly.
* **Exception:**
  + System displays notification if the form is not filled correctly.

#### 5.3.2.5 UC-5 Admin search account

* **Description:** Admin can search account by text, username, email, id, first name and last name.
* **Actors:** Admin
* **Preconditions:** Admin is logged in.
* **Postconditions:** System show list of accounts that match the search criteria.
* **Flow of Events:**
  + Admin go to EduBlock.
  + Admin login with username and password.
  + Admin click on “Account”.
  + Admin click on “Search” button.
  + Admin input text to search account.
  + Admin add search criteria.
  + System show list of accounts that match the search criteria.
* **Alternate Flow:**
  + System displays notification “No account found” if there is no account that match the search criteria.
* **Exception:**
  + System displays notification “No account found” if there is no account that match the search criteria.

### 5.3.3 **Staff Features**

#### 5.3.3.1 UC-6 Staff Login

* **Description:** Staff can login with their username and password.
* **Actors:** Staff.
* **Preconditions:** Staff has an account.
* **Postconditions:** Staff can access the system.
* **Flow of Events:**
  + Staff go to EduBlock.
  + Staff enters their username and password.
  + System verifies the username and password.
  + System redirect Staff to dashboard.
* **Exceptions:**
  + If the username or password is incorrect, the system will display an error message.

#### 5.3.3.2 UC-7 Staff view list of accounts

* **Description:** Staff can view list of all accounts.
* **Actors:** Staff
* **Preconditions:** Staff is logged in.
* **Postconditions:** System show list of all accounts.
* **Flow of Events:**
  + Staff go to EduBlock.
  + Staff login with username and password.
  + Staff click on “Account”.
  + System show list of all accounts.
* **Alternate Flow:**
  + System displays notification “No account found” if there is no account.
* **Exception:**
  + System displays notification “No account found” if there is no account.

#### 5.3.3.3 UC-8 Staff view account list by role

* **Description:** Staff can view list of accounts by role.
* **Actors:** Staff
* **Preconditions:** Staff is logged in.
* **Postconditions:** System show list of accounts by role.
* **Flow of Events:**
  + Staff go to EduBlock.
  + Staff login with username and password.
  + Staff click on “Account”.
  + Staff click on role’s name to view list of accounts by role.
  + System show list of accounts by role.
* **Alternate Flow:**
  + System displays notification “No account found” if there is no account.
* **Exception:**
  + System displays notification “No account found” if there is no account.

#### 5.3.3.4 UC-9 Staff view account details

* **Description:** Staff can view account details.
* **Actors:** Staff
* **Preconditions:** Staff is logged in.
* **Postconditions:** System show account details.
* **Flow of Events:**
  + Staff go to EduBlock.
  + Staff login with username and password.
  + Staff click on “Account”.
  + Staff click on “Details” (human icon) on actions column.
  + System show account details.
* **Alternate Flow:**
  + System displays notification “No account found” if there is no account.
* **Exception:**
  + System displays notification “No account found” if there is no account.

#### 5.3.3.5 UC-10 Staff search account

* **Description:** Staff can search account by text, username, email, id, first name and last name.
* **Actors:** Staff
* **Preconditions:** Staff is logged in.
* **Postconditions:** System show list of accounts that match the search criteria.
* **Flow of Events:**
  + Staff go to EduBlock.
  + Staff login with username and password.
  + Staff click on “Account”.
  + Staff click on “Search” button.
  + Staff input text to search account.
  + Staff add search criteria.
  + System show list of accounts that match the search criteria.
* **Alternate Flow:**
  + System displays notification “No account found” if there is no account that match the search criteria.
* **Exception:**
  + System displays notification “No account found” if there is no account that match the search criteria.

#### 5.3.3.6 UC-11 Staff view class list

* **Description:** Staff can view list of all classes.
* **Actors:** Staff
* **Preconditions:** Staff is logged in.
* **Postconditions:** System show list of all classes.
* **Flow of Events:**
  + Staff go to EduBlock.
  + Staff login with username and password.
  + Staff click on “Classroom”.
  + System show list of all classes.
* **Alternate Flow:**
  + System displays notification “No class found” if there is no class.
* **Exception:**
  + System displays notification “No class found” if there is no class.

### 5.3.4 **Teacher Features**

### 5.3.5 **Student Features**

## 5.4 Non-Functional Requirements

### 5.4.1 External Interfaces

* Custom Fabric network

### 5.4.2 Quality Attributes

Our application ensures the following quality attributes:

* **Usability:** The application is easy to use and understand. The application is designed to be intuitive and easy to use. The application is designed to be used by both teachers and students.
* **Reliability:** The application is designed to be reliable. The application is designed to be used with blockchain technology to ensure data integrity.
* **Performance:** The application is designed to be fast and responsive.
* **Security:** The application is designed to be secure. The application is designed to be used with blockchain technology to ensure data integrity.
* **Maintainability:** The application is designed to be easy to maintain, update, and extend.
* **Portability:** The application is designed to be portable.
* **Scalability:** The application is designed to be scalable and can be extended to support more users and more features.
* **Interoperability:** The application is designed to be interoperable with other applications.
* **Reusability:** The application is designed to be reusable.
* **Testability:** The application is designed to be easy to test.

## 5.5 Other Requirements

# 6. Software Design Description

## 6.1 Overall Description

### 6.1.1 Assumptions

### 6.1.2 Design Constraints

### 6.1.3 Technology Suggestion

## 6.2 System Architecture Design

### 6.2.1 Overall Architecture

|  |
| --- |
| Figure 6.1: Overall architecture |

| Component | Description |
| --- | --- |
| Chain Node (CN) | A node of the blockchain. This stores the records and handles the history and transaction requests from the Request Server (Change/View the score, information, etc.) |
| Request Server | The off-chain backend of a CN. This stores the pending requests from the user and is the only way to call a request to the CN. Each Request Server may have a different way to handle user requests (Voting, Direct Request, etc.) |
| Frontend Server | Provide the UX/UI for interacting with the Request Server |

### 6.2.2 System Architecture

### 6.2.3 Package Diagram

|  |
| --- |
| Figure 6.2: Package Diagram of Request Server |

| Package Name | Description |
| --- | --- |
| root | Main classes |
| api | The abstract classes & interfaces |
| entity | The entities of the database |
| handler | The handlers of the endpoints of the REST API server |
| internal | Internal classes used by other packages |
| internal/student | The instances of the Student Updater |
| model | The input / output objects |
| model/input | The input objects for the handlers |
| model/output | The output objects returned from the handlers |
| model/fabric | The models used internally by the student updater |

## 6.3 System Detailed Design

### 6.3.1 Class Diagram

|  |
| --- |
| Figure 6.3: Class Diagram of the Request Server |

### 6.3.2 Sequence Diagram

## 6.4 Class Specification

## 6.5 Data & Database Design

### 6.5.1 Database Design

|  |
| --- |
| Figure 6.4: Database Design of the Request Server |

#### 6.5.1.1 Account

| Field Name | Type | Size | Unique | Not Null | Flag | Notes |
| --- | --- | --- | --- | --- | --- | --- |
| ID | bigint |  | x | x | PK |  |
| USERNAME | character varying | 255 | x | x |  |  |
| HASHEDPASSWORD | character varying | 255 |  | x |  |  |
| SALT | character varying | 255 |  | x |  |  |
| ROLE | character varying | 255 |  | x |  |  |
| CREATEDAT | timestamp |  |  | x |  |  |

#### 6.5.1.2 Profile

| Field Name | Type | Size | Unique | Not Null | Flag | Notes |
| --- | --- | --- | --- | --- | --- | --- |
| ACCOUNT\_ID | bigint |  | x | x | PK, FK |  |
| ADDRESS | character varying | 255 |  | x |  |  |
| AVATAR | character varying | 255 |  | x |  |  |
| BIRTHDATE | timestamp |  |  | x |  |  |
| EMAIL | character varying | 255 |  | x |  |  |
| FIRSTNAME | character varying | 255 |  | x |  |  |
| LASTNAME | character varying | 255 |  | x |  |  |
| MALE | boolean |  |  | x |  |  |
| PHONE | character varying | 255 |  | x |  |  |
| UPDATED | boolean |  |  | x |  | Used internally by student updater |

#### 6.5.1.3 Student

| Field Name | Type | Size | Unique | Not Null | Flag | Notes |
| --- | --- | --- | --- | --- | --- | --- |
| ACCOUNT\_ID | bigint |  | x | x | PK, FK |  |
| ETHNIC | character varying | 255 |  | x |  |  |
| FATHERJOB | character varying | 255 |  | x |  |  |
| FATHERNAME | character varying | 255 |  | x |  |  |
| GUARDIANJOB | character varying | 255 |  | x |  |  |
| GUARDIANNAME | character varying | 255 |  | x |  |  |
| HOMETOWN | character varying | 255 |  | x |  |  |
| MOTHERJOB | character varying | 255 |  | x |  |  |
| MOTHERNAME | character varying | 255 |  | x |  |  |

#### 6.5.1.4 Classroom

| Field Name | Type | Size | Unique | Not Null | Flag | Notes |
| --- | --- | --- | --- | --- | --- | --- |
| ID | bigint |  | x | x | PK |  |
| NAME | character varying | 255 |  | x |  |  |
| GRADE | character varying | 255 |  | x |  |  |
| HOMEROOMTEACHER\_ID | bigint |  |  | x | FK |  |
| START\_YEAR | integer |  |  | x |  |  |

#### 6.5.1.5 Class Student

| Field Name | Type | Size | Unique | Not Null | Flag | Notes |
| --- | --- | --- | --- | --- | --- | --- |
| ID | bigint |  | x | x | PK |  |
| CLASSROOM\_ID | bigint |  |  | x | FK |  |
| STUDENT\_ID | bigint |  |  | x | FK |  |

#### 6.5.1.6 Class Teacher

| Field Name | Type | Size | Unique | Not Null | Flag | Notes |
| --- | --- | --- | --- | --- | --- | --- |
| ID | bigint |  | x | x | PK |  |
| CLASSROOM\_ID | bigint |  |  | x | FK |  |
| TEACHER\_ID | bigint |  |  | x | FK |  |
| SUBJECTID | bigint |  |  | x |  | Defined in the system’s config |

#### 6.5.1.7 Record

| Field Name | Type | Size | Unique | Not Null | Flag | Notes |
| --- | --- | --- | --- | --- | --- | --- |
| ID | bigint |  | x | x | PK |  |
| CLASSROOM\_ID | bigint |  |  | x | FK |  |
| STUDENT\_ACCOUNT\_ID | bigint |  |  | x | FK |  |

#### 6.5.1.8 Record Entry

| Field Name | Type | Size | Unique | Not Null | Flag | Notes |
| --- | --- | --- | --- | --- | --- | --- |
| ID | bigint |  | x | x | PK |  |
| RECORD\_ID | bigint |  |  | x | FK |  |
| REQUESTER\_ID | bigint |  |  |  | FK |  |
| TEACHER\_ID | bigint |  |  |  | FK |  |
| APPROVER\_ID | bigint |  |  |  | FK |  |
| APPROVALDATE | timestamp |  |  | x |  |  |
| REQUESTDATE | timestamp |  |  | x |  |  |
| FIRSTHALFSCORE | double precision |  |  | x |  |  |
| SECONDHALFSCORE | double precision |  |  | x |  |  |
| FINALSCORE | double precision |  |  | x |  |  |
| SUBJECTID | bigint |  |  | x |  | Defined in the system’s config |
| UPDATECOMPLETE | boolean |  |  | x |  | Used internally by student updater |

#### 6.5.1.9 Pending Record Entry

| Field Name | Type | Size | Unique | Not Null | Flag | Notes |
| --- | --- | --- | --- | --- | --- | --- |
| ID | bigint |  | x | x | PK |  |
| RECORD\_ID | bigint |  |  | x | FK |  |
| REQUESTER\_ID | bigint |  |  | x | FK |  |
| TEACHER\_ID | bigint |  |  | x | FK |  |
| REQUESTDATE | timestamp |  |  | x |  |  |
| FIRSTHALFSCORE | double precision |  |  | x |  |  |
| SECONDHALFSCORE | double precision |  |  | x |  |  |
| FINALSCORE | double precision |  |  | x |  |  |
| SUBJECTID | bigint |  |  | x |  | Defined in the system’s config |

### 6.5.2 Updater Key

| Field Name | Type | Size | Unique | Not Null | Flag | Notes |
| --- | --- | --- | --- | --- | --- | --- |
| ID | character varying | 255 | x | x | PK |  |
| STUDENT\_ACCOUNT\_ID | bigint |  |  | x | FK |  |

### 6.5.3 Data File Design

| File Name | Type | Notes |
| --- | --- | --- |
| db | Folder | The folder of The H2 Database files |
| updater | Folder | Contains the data files of the local student updater |

# 7. Software Testing Documentation

## 7.1 Overall Description

### 7.1.1 Test Model

We apply the V-model in our project, which is a development of the waterfall model. Testing is carried out concurrently with the software development cycle in the V-model, where a testing phase corresponds to a phase of software development.

### 7.1.2 Testing Levels

About the Testing levels in our project, we apply all those levels including Unit testing, Integration testing, System testing and Acceptance testing.

With Unit testing, we test each small module in the system, each class and function. Eg (………)

With Integration testing is a type of testing in which individual software modules or functions are logically integrated and tested in groups together. For instance, we can test the interoperability of two functions, add 1 item and search for the item to see if they interact well with each other, after successfully creating an item, we can proceed to search for the newly created item. or not.

System testing is the last test phase to determine whether the system is about to deliver satisfying the requirements and goals. It tests the whole functionality and interface of the system. For instance, a database test for a system test is used to see if the data displayed on the system matches the data in the database.

Finally, with Acceptance Test, similar to System Test but usually tested by customers, the purpose is to see if the software meets the customer’s requirements or not.

### 7.1.3 Testing Types

Functional testing is checking if the system is working according to the business requirements and is performed in every level of testing. Non-Functional testing is similar to Functional testing in that both occur in all levels of testing. Non-functional testing is primarily concerned with the software’s other features, such as its security and if data is exposed by straightforward queries in any input field. Structural testing is often considered a type of white box testing. Instead than focusing on the software’s functionality, this method examines what is happening inside the program. Structural testing is also applicable at all testing levels. Changes Testing is done to determine whether or not the program is functioning correctly after bugs have been fixed.

## 7.2 Test Plan

### 7.2.1 Test Stages

### 7.2.2 Resources

#### 7.2.2.1 Human Resources

#### 7.2.2.2 Environment

### 7.2.3 Test Milestones

### 7.2.4 Deliverables

## 7.3 Test Cases

## 7.4 Test Reports

# 8. Release Package & User Guides

## 8.1 Deliverable Package

### 8.1.1 Source Codes & Documents

| No. | Items | Sub-Items | Type | Version |
| --- | --- | --- | --- | --- |
| **Code package** |  |  |  |  |
| 1 | EduBlock | EduBlock Client | New | 1.0 |
| 2 | Blockchain Network | Blockchain Chaincode | New | 1.0 |
| 3 | OCR | Record Table Processor | New | 1.0 |
| **Database** |  |  |  |  |
| 1 | Tables | accounts.sql | New | 1.0 |
|  |  | classrooms.sql | New | 1.0 |
|  |  | subjects.sql | New | 1.0 |
| **Documents** |  |  |  |  |
| 1 | Requirement | SRS\_v1.0.docx | New | 1.0 |
| 2 | Deployment | UserGuide\_v1.0.docx | New | 1.0 |

### 8.1.2 Known Issues, Limitations & Restrictions

## 8.2 Installation Guides

### 8.2.1 System Requirements

OS: any

CPU: at least 4 cores

RAM: at least 4Gb

NETWORK: required

SOFTWARE: Docker

### 8.2.2 Setup Files

* Dockerfile.backend
* Dockerfile.frontend
* Dockerfile.ocr

### 8.2.3 Installation Instruction

## 8.3 User Manual

### 8.3.1 Terms & Definitions

| No. | Term | Definition |
| --- | --- | --- |
| 01 | F.FT | Feature |
| 02 | R.ADM | Admin |
| 03 | R.STF | Staff |
| 04 | R.TCH | Teacher |
| 05 | R.STD | Student |
| 06 | R.ANY | Any role |

### 8.3.2 System Requirements

OS: any

CPU: any

RAM: at least 1Gb

NETWORK: required

### 8.3.3 Application Usage

#### 8.3.3.1 Overview

| No. | Feature | Role | Note |
| --- | --- | --- | --- |
| 01 | Create new account | R.ADM |  |
| 02 | View account list | R.ADM, R.STF |  |
| 03 | View profile | R.ANY | Each role have different behavior |
| 04 | Update profile | R.ADM, R.STF | Each role have different behavior |
| 05 | Update password | R.ANY | Each role have different behavior |
| 06 | Create new classroom | R.STF |  |
| 07 | View classroom list | R.STF, R.TCH, R.STD | Each role have different behavior |
| 08 | View classroom information | R.STF, R.TCH, R.STD |  |
| 09 | View students of classroom | R.STF, R.TCH, R.STD |  |
| 10 | View teachers of classroom | R.STF, R.TCH, R.STD |  |
| 11 | Update classroom information | R.STF |  |
| 12 | Update student in classroom | R.STF |  |
| 13 | Update teacher in classroom | R.STF |  |
| 14 | View update request list | R.TCH |  |
| 15 | Verify update request | R.TCH |  |
| 16 | Request update record | R.TCH, R.STD |  |

#### 8.3.3.2 Feature 01: Create new account

**Description:**

* Admin create account for other user usage

**Details:**

* R.ADM
  + Step 1: Click Account on the left navigation bar to navigate to account list page
  + Step 2: Click Create at the top left of the page to open a modal with form
  + Step 3: Input user First name, Last name and select a role for user
  + Step 4: (Optional) Click Add at the bottom left of the form to add more account and repeat from Step 1
  + Step 5: Click Create at the bottom right of the form to confirm the account creation

#### 8.3.3.3 Feature 02: View account list

**Description:**

* Admin, Staff view the account list to manage account information and find reference for other operations

**Details:**

* R.ADM
  + Step 1: Click Account on the left navigation bar to navigate to account list page
  + Step 2: (Optional) Click Search to reveal filter options below
  + Step 2.1: Select search field on the left
  + Step 2.2: Input search text on the right
  + Step 2.3: Click Search button at the right most to apply list filter
  + Step 3: View list of account
  + Step 4: (Optional) Click page number at the top right to view other accounts

#### 8.3.3.4 Feature 03: View profile

**Description:**

* Admin, Staff view user profile
* User view personal profile

**Details:**

* R.ANY (Personal)
  + Step 1: Click personal card at the bottom of the Vertical Navigation bar to navigate to the profile page
* R.ADM, R.STF
  + Step 1: Click Account on the left navigation bar to navigate to account list page
  + Step 2: Look for the specific account row in table
  + Step 3: Click Details in the Actions column to navigate to the profile page of that account

#### 8.3.3.5 Feature 04: Update profile

**Description:**

* Admin, Staff update personal profile
* Staff update Teacher, Student profile

**Details**

* R.ADM, R.STF (Personal)
  + Step 1: Click personal card at the bottom of the Vertical Navigation bar to navigate to the profile page
  + Step 2: Click Update in the profile section to open a modal with form
  + Step 3: Change the form data to desired value
  + Step 4: Click Confirm to save the changes.
* R.STF
  + Step 1: Click Account on the left navigation bar to navigate to account list page
  + Step 2: Look for the specific account row in table
  + Step 3: Click Update in the Actions column to open an update modal with form
  + Step 4: Change the form data to desired value
  + Step 5: Click Confirm to save the changes

#### 8.3.3.6 Feature 05: Update password

**Description:**

* Admin update other user password
* User self update password

**Details**

* R.ANY (Personal)
  + Step 1: Click personal card at the bottom of the Vertical Navigation bar to navigate to the profile page
  + Step 2: Click Update password at the top right of the page to open a modal with form
  + Step 3: Input the new password
  + Step 4: Click Confirm to save the new password
* R.ADM
  + Step 1: Click Account on the left navigation bar to navigate to account list page
  + Step 2: Look for the specific account row in table
  + Step 3: Click Update password in the Actions column to open an update modal with form
  + Step 4: Input the new password
  + Step 5: Click Confirm to save the new password

#### 8.3.3.7 Feature 06: Create new classroom

**Description:**

* Staff Create new classroom in the system

**Details**

* R.STF
  + Step 1: Click Classroom on the left navigation bar to navigate to classroom list page
  + Step 2: Click Create at the top left of the page to open a modal with form
  + Step 3: Change the form data to desired value
  + Step 4: Click Confirm at the bottom right of the modal to save the created classroom

#### 8.3.3.8 Feature 07: View classroom list

**Description:**

* Staff view the list of all the classroom in the system
* Teacher view the list of all the classroom being taught by that teacher
* Student view the list of all the classroom that student taking part in

**Details**

* R.STF
  + Step 1: Click Classroom on the left navigation bar to navigate to classroom list page
* R.TCH, R.STD
  + Step 1: The list of classroom is in the dashboard page

#### 8.3.3.9 Feature 08: View classroom information

**Description:**

* Staff view classroom information
* Teacher view information of the classroom being taught by that teacher
* Student view information of the classroom that student taking part in

**Details**

* R.STF
  + Step 1: Click Classroom on the left navigation bar to navigate to classroom list page
  + Step 2: Look for the specific classroom row in table
  + Step 3: Click Details in the Actions column to navigate to the classroom details page
* R.TCH, R.STD
  + Step 1: Click Dashboard on the left navigation bar to navigate to classroom list page
  + Step 2: Look for the specific classroom row in table
  + Step 3: Click Details in the Actions column to navigate to the classroom information page

#### 8.3.3.10 Feature 09: View students of classroom

**Description:**

* Staff view all the student in a specific classroom
* Teacher view all the student in the classroom being taught by that teacher
* Student view all the student in the classroom that student taking part in

**Details**

* R.STF, R.TCH, R.STD
  + Step 1: Follow Feature 08 to navigate to the classroom information page
  + Step 2: Click Student which is the center tab at the top of the page to navigate to student list of that classroom

#### 8.3.3.11 Feature 10: View teachers of classroom

**Description:**

* Staff view all the teacher in a specific classroom
* Teacher view all the teacher in the classroom being taught by that teacher
* Student view all the teacher in the classroom that student taking part in

**Details**

* R.STF, R.TCH, R.STD
  + Step 1: Follow Feature 08 to navigate to the classroom information page
  + Step 2: Click Teacher which is right most tab at the top of the page to navigate to teacher list of that classroom

#### 8.3.3.12 Feature 11: Update classroom information

**Description:**

* Staff update a specific classroom information

**Details**

* R.STF
  + Step 1: Follow Feature 08 to navigate to the classroom information page
  + Step 2: Click Details which is right most tab at the top of the page to navigate to details page of that classroom
  + Step 3: Click Update at the bottom of the page to open a modal with form
  + Step 4: Change the form data to desired value
  + Step 5: Click Confirm at the bottom of the modal to save changes

#### 8.3.3.13 Feature 12: Update student in classroom

**Description:**

* Staff change the student of a specific classroom

**Details**

* R.STF
  + Step 1: Follow Feature 09 to navigate to the classroom student page
  + Step 2: Look for a specific account row in the table (May skip to Step 4)
  + Step 3: Click Remove in the Actions column to remove student from classroom
  + Step 4: Click Add at the top left of the page to open a modal with form
  + Step 5: Change the form data to desired value
  + Step 6: Click Confirm at the bottom right of the modal to save changes

#### 8.3.3.14 Feature 13: Update teacher in classroom

**Description:**

* Staff change the teacher of a specific classroom

**Details**

* R.STF
  + Step 1: Follow Feature 10 to navigate to the classroom teacher page
  + Step 2: Look for a specific account row in the table (May skip to Step 4)
  + Step 3: Click Remove in the Actions column to remove teacher from classroom
  + Step 4: Click Add at the top left of the page to open a modal with form
  + Step 5: Change the form data to desired value
  + Step 6: Click Confirm at the bottom right of the modal to save changes

#### 8.3.3.15 Feature 14: View update request list

**Description:**

* Teacher view list of request for updating record value

**Details**

* R.TCH
  + Step 1: Click Request on the left navigation bar to navigate to request list page

#### 8.3.3.16 Feature 15: Verify update request

**Description:**

* Teacher verify request waiting for verification

**Details**

* R.TCH
  + Step 1: Follow Feature 14 to navigate to the request list page
  + Step 2: Look for the specific request row in table
  + Step 3: Click Approve or Reject to approve or reject the request

#### 8.3.3.17 Feature 16: Request update record

**Description:**

* Teacher or student of same classroom request updating record for that student

**Details**

* R.TCH
  + Step 1: Follow Feature 09 to navigate to student profile page
  + Step 2: Look for the record need update in the record table at the bottom of the page
  + Step 3: Click Update in the Actions column to open a modal with form
  + Step 4: Change the form data to desired value
  + Step 5: Click Confirm to send the request
* R.STD
  + Step 1: Click personal card at the bottom of the Vertical Navigation bar to navigate to the profile page
  + Step 2: Look for the record need update in the record table at the bottom of the page
  + Step 3: Click Update in the Actions column to open a modal with form
  + Step 4: Change the form data to desired value
  + Step 5: Click Confirm to send the request

### 8.3.4 Troubleshooting

# References

# Appendix