EduBlock

etases

12/10/22

Table of contents

# Acknowledgement

Thanks to the following people for their help and support:

* [FPT Edu Experience Space](https://www.facebook.com/FEExpSpace) for providing an opportunity for us to do a team building activity and to have a chance to learn basic Blockchain technology.
* [Quách Luyl Đa](https://www.facebook.com/luyldaquach), our mentor & supervisor, for his guidance and support. He encouraged us to do our best and gave us a lot of useful advice.
* Fellow members at [ETASES](https://github.com/etases) to work together, share knowledge and experience, and to learn from each other. They did their best to make this project more complete and better.
* You, the reader, for your interest in our project.

A digital version of this report can be found here: <https://etases.github.io/EduBlock.Report/>

# Definition and Acronyms

Definition and Acronyms

| Acronym | Definition |
| --- | --- |
| BR | Business Rule |
| BA | Business Analysis |
| SRS | Software Requirement Specification |
| UC | Use Case |
| UAT | User Acceptance Test |
| API | Application Programming Interface |
| GUI | Graphical User Interface |
| RS | Request Server |
| CN | Chain Node |
| OCR | Optical Character Recognition |
| PM | Project Manager |
| Fabric | Hyperledger Fabric |
| FE | Feature |
| LI | Limitation |
| SI | Software Interface |
| HI | Hardware Interface |
| UI | User Interface |
| CRUD | Create, Read, Update & Delete |

# 1. Project Introduction

## 1.1 Overview

### 1.1.1 Project Information

* Project name: BlockChain application in academic record management to support online University/College admissions
* Project code: EduBlock
* Group name: ETASES
* Software type: Web app

### 1.1.2 Project Team

#### 1.1.2.1 Supervisor

Table 1.1: Supervisors

| Full Name | Email | Phone Number | Title |
| --- | --- | --- | --- |
| Quach Luyl Da | daql@fpt.edu.vn | 0976703075 | Lecture |

#### 1.1.2.2 Team Members

Table 1.2: Team Members

| Full Name | Email | Phone Number | Title |
| --- | --- | --- | --- |
| Huynh Quang Tien | TienHQCE150130@fpt.edu.vn | 0976608340 | Leader |
| Le Xuan Tu | TuLXCE150344@fpt.edu.vn | 0939774512 | Member |
| Nguyen Dang Khoa | KhoaNDCE140165@fpt.edu.vn | 0382554293 | Member |
| Cao Hoang Anh Uy | UyCHACE150661@fpt.edu.vn | 0706456981 | Member |
| Nguyen Minh Khoi | KhoiNMCE150103@fpt.edu.vn | 0338616352 | Member |

## 1.2 Background

The number of academic institutions still use manual processes to store and transfer academic records like transcripts and certifications between institutions and to potential organizations, despite the fact that many large institutions are now adopting the modern practice of maintaining electronic academic records. The process can take up to several days, just for students who want to review their own transcripts, so a common transfer for a student can take anywhere from a few weeks to a month. Due to the time required to process and submit appeal requests using the widely used paper method, more serious errors could happen, and the process could take several months. In addition to the significant wait time and the possibility of physical damage or loss of records during storage and transportation, there is also the risk of credential tampering by fraudulent parties. The cost of processing time, manual work effort, postage, and transit fees, as well as the storage and shipping of physical records, are also very expensive.

The emerging solutions are primarily based on email-based solutions or the transfer of PDF files while remaining limited by nationality, privacy and security barriers. Although the popularity of cryptocurrencies and NFTs has led to the implementation of Blockchain as a host of applications in the financial sector, the field is more diverse in both technical and application areas [1]. As distributed applications are increasingly applied in various fields such as data storage (including handling medical records and healthcare [2,5]), Cloud and Grid Computing [3], e-vote [4], Service for IoT [6], Banking system[7] and foremost is the field of Education. Academic institutions can benefit from blockchain technology to provide a decentralized and immutable ledger to confirm the integrity of academic records [10]. Then, solutions for storing and anti-fraud of online electronic degrees have also been conceived to bring the initial benefits of applying Blockchain technology [8][9]. While these solutions provide a more modern approach to the storage and the transfer of academic records, there are still limitations in terms of widespread adoption, auditability, and scalability. A successful solution for storing and exchanging electronic school records will include Security and Privacy, Scalability and at the same time benefit from the advantages of blockchain technology as Distributed, Transparency further described.

The goal of our proposed system is to address the limitations of existing solutions by utilizing Blockchain technology to provide a secure, verifiable, and tamper-proof method of storing, accessing, managing, and exchanging electronic school records between institutions.

## 1.3 Existing Systems

### 1.3.1 Blockcerts

Blockcerts is an open standard platform for developing, issuing, and verifying blockchain-backed certificates that Learning Machine and the MIT Media Lab jointly developed. The business can assess the validity of documents and identify fraudulent information by generating records like academic transcripts and certificates on a blockchain. Grades, transcripts, and even degrees can all be kept on a Blockcerts blockchain enabling immutable access to previous academic performance.

### 1.3.2 APPII

The blockchain, smart contracts, and machine learning technologies used by APPII are used to validate the academic credentials of potential students and lecturers. Users set up a profile and complete their academic CV, which includes their academic background and transcripts. The user’s background is subsequently verified by APPII using blockchain, and their data is then locked into its blockchain.

### 1.3.3 Parchment

Students, academic institutions, and employers can use Parchment’s digital certificate services. Higher education institutions use the platform to evaluate academic excellence, process applications, and generate immutable degrees, while K-12 educators use the blockchain of the company to upload any significant developmental progress. Additionally, all educational information is permanently accessible to students, and they may readily tell prospective employers about their academic accomplishments.

### 1.3.4 Conclusion

From all systems mentioned above, in this design we presented a solution for managing and storing electronic academic records as a replacement for the traditional academic record based on distributed storage technology used by Blockchain, where the data is stored in a block and the blocks are connected on a chain by hashing. Our network enables us to manage data in the network using transactions via smart contracts. From there next, we show how to set up a multi-tier network and processes. Our network enables us to decentralize organizations and system users through arranging chain nodes, verifying transactions with smart contracts, archiving modification history and restoring data of a node using data from other nodes.

From this design, it can be concluded and proposed to organize a Permissioned Blockchain network with a multi-tier design. The main advantage of applying Permissioned Blockchain technology is its resistance to many threats and cyber attacks, rely on the hashing mechanism and the nodes on the Blockchain can prevent data breaches. And moreover, it offers a host of unique features such as improved reliability, better fault tolerance, faster and more efficient operation, and scalability.

And thus, the management of documents for the field of education has the potential to be significantly impacted by the integration of Blockchain, the hyperledger framework, and smart contract technologies across academic records.

## 1.4 Business Opportunity

Although many high schools in Vietnam still keep paper records for post-graduation and enrolment in college or university, many now employ an online system to track students’ academic progress and inform parents of any latest outcomes. Because they must do it for both the paper records and the internet system, teachers find it challenging to update the information on their students. There is also a minimum level of transparency for students who wish to verify their information at any time because the internet system is centralized and only administrators and teachers have access to it. As a result, a system is required to help teachers and students manage student records in an easier, quicker, and more effective manner.

## 1.5 Software Product Vision

With the use of this system, students may simply keep track of changes to their grades in their academic records, reducing teacher grade entry errors. In order to gain rapid admission to graduate programs and colleges, students can also more conveniently retrieve their transcripts. High security and restrictions on data editing also assist in limiting the issue of phony points that are inaccurate representations of reality. Additionally, it eliminates challenges with entering grades into instructors’ school records because doing so will be quicker and easier with the aid of the system.

## 1.6 Project Scope & Limitations

A record management system will always be the best in terms of security and purity because it must, of course, assure data security. When interacting with and storing the data, always keep it intact and unaltered. Additionally, it must be user-friendly, with an interface that is clear and unambiguous and avoids misinterpretations of the translation or the information on the screen. Without the responsible user’s consent, data editing procedures cannot be carried out at will.

### 1.6.1 Major Features

FE-01: Using the blockchain platform, store data.

FE-02: Transcripts can be updated by converting photos to alphanumeric data.

FE-03: May replace paper school records entirely (electronic school records but have the nature of paper school records).

FE-04: Enhancing the effort teachers put into entering grades.

FE-05: Utilization dependability for admissions parties.

### 1.6.2 Limitations & Exclusions

LI-1: It is impossible to synchronize student counts between institutions due to the dispersed nature of the data.

LI-2: The only time to use the system is at the end of the year because it only saves the semester’s overall grade (can be expanded later)

LI-3: There is no option to switch schools (due to not processing student codes synchronously)

# 2. Project Management Plan

## 2.1 Overview

### 2.1.1 WBS & Estimation

Table 2.1: Work Breakdown Structure (WBS) and Estimation

| ID | WBS Item | Complexity | Estimated Effort (man-days) |
| --- | --- | --- | --- |
| 1 | Analysis |  | 20 |
| 1.1 | SRS | Medium | 5 |
| 1.2 | UML | Simple | 5 |
| 1.3 | Technology Research | Medium | 10 |
| 2 | Design |  | 20 |
| 2.1 | Architecture Design | Medium | 5 |
| 2.2 | Database Design | Medium | 5 |
| 2.3 | Detailed Design | Medium | 10 |
| 3 | Development |  | 80 |
| 3.1 | Backend | Medium | 30 |
| 3.2 | Frontend | Medium | 30 |
| 3.3 | Blockchain | Complex | 10 |
| 3.4 | OCR | Complex | 10 |
| 4 | Testing |  | 15 |
| 4.1 | Unit Testing | Simple | 3 |
| 4.2 | Integration Testing | Medium | 5 |
| 4.3 | System Testing | Medium | 7 |
| 5 | Documentation |  | 20 |
| 5.1 | Final Report | Medium | 10 |
| 5.2 | Final Product | Medium | 10 |
| # | Total |  | 155 |

### 2.1.2 Project Objectives

* Timeliness: 70%
* Allocated Effort (man-days): 90 days
* Quality Coverage:
  + Reviewing: 100%
  + Unit Test: 95%
  + Integration Test: 95%
  + System Test: 98%
  + User Acceptance Test: 80%

### 2.1.3 Project Risks

Table 2.2: Project Risks

| Risk | Impact | Probability | Mitigation |
| --- | --- | --- | --- |
| The team members are not familiar with the technologies used in the project | High | High | The team members will be trained before the project starts |
| The team members can not work together | High | Medium | Prepare a team building activity before the project starts |
| There is not enough time to research the technologies used in the project | High | Low | Find a mentor to help the team members / Find a workaround or similar technology that can be used in the project |

## 2.2 Management Approach

### 2.2.1 Project Process

|  |  |  |
| --- | --- | --- |
| |  | | --- | | (a) Agile |      |  | | --- | | (b) Scrum |   Figure 2.1: Software Development Process Model |

Since parts of the requirements are not clear, we will use a process model of the Agile family called Scrum. Using this model, we can understand requirements, adapt changes quickly and deliver target products in a short time to test & review them for next iterations. The project will be divided into 4 iterations (sprints). Each sprint will be a month long and include 2 phases: planning and development. The planning phase will be used to estimate tasks from the product backlog, create a sprint backlog and assign tasks to the team members to develop in the development phase. At the end of the sprint, the team will review the sprint, discuss the problems and changes during the sprint and plan for the next sprint.

### 2.2.2 Quality Management

* Reviewing: There will be a meeting every week for the team and the supervisor to review the status of the tasks in the current sprint. The team will show the progress of the tasks and discuss the problems and changes during the sprint. The supervisor will take notes and give advices to the team.
* Defect Prevention: Each change of the code should be reviewed by the code master or the team leader before applying to the main project. They will check for bugs, code smells, anti-patterns and other problems.
* Unit Testing: Each module of the system should be tested before applying to the main project. The team can test the module by themselves or use a testing framework / software to test the module.
* Integration Testing: The team will test the system after each sprint to make sure that the system is working properly. There will be an automatic testing software that includes test scenarios for the system.
* System Testing: The team will test the completed system based on the requirements and the test scenarios. The team can test the system by themselves or use a testing framework / software to test the system.

### 2.2.3 Training Plan

Table 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 |

## 2.3 Master Schedule

Table 2.4: Master Schedule

| Deliverable | Due | Deliverable Scope |
| --- | --- | --- |
| Project Plan | 05/09/2022 |  |
| SRS | 15/09/2022 |  |
| Design 1 | 15/09/2022 | Architecture Design, Database |
| Design 2 | 30/09/2022 | Detailed Design |
| Backend | 30/09/2022 | Code, Test, Deployment |
| Frontend | 30/11/2022 | Code, Test, Deployment |
| Extra Services | 30/11/2022 | Blockchain, OCR |
| Final Report | 10/12/2022 |  |
| Final Product | 15/12/2022 |  |

## 2.4 Project Organization

### 2.4.1 Team Structure

|  |
| --- |
| Figure 2.2: Team Structure |

### 2.4.2 Roles

Table 2.5: Roles

| Role | Responsibilities |
| --- | --- |
| Project Manager | Supervise the project, Manage the team, Manage the schedule, Manage deliverables |
| Analysis Leader | Analyze the requirements, Create SRS, Manage the analysis team |
| Analysis Member | Analyze the requirements, Create SRS |
| Design Leader | Design the system, Create system design document, Manage the design team |
| Design Member | Design the system, Create system design document |
| Backend Leader | Develop the backend, Manage the backend team |
| Backend Member | Develop the backend |
| Frontend Leader | Develop the frontend, Manage the frontend team |
| Frontend Member | Develop the frontend |
| Test Leader | Test the system, Create test report, Manage the test team |
| Test Member | Test the system, Create test report |

## 2.5 Project Communication

### 2.5.1 Communication Plan

Table 2.6: 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 |
| FUCT Library | Project Team | Pair programming & Code review | Planned | Offline |

### 2.5.2 External Interfaces

Table 2.7: External Interfaces

| Function | Contact Person | Contact Address | Responsibility |
| --- | --- | --- | --- |
| Supervisor | Quach Luyl Da | daql@fpt.edu.vn | Review deliverables, Provide document template, Check project progress, Give advices to project team |

## 2.6 Configuration Management

### 2.6.1 Tools & Infrastructures

#### 2.6.1.1 Common

Table 2.8: Common Tools

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

### 2.6.2 Backend

Table 2.9: Backend Tools

| 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 |

### 2.6.3 Frontend

Table 2.10: Frontend Tools

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

### 2.6.4 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.

### 2.6.5 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.

# 3. Software Requirement Specification

## 3.1 Overall Description

### 3.1.1 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 environmentally. EduBlock will help the school to manage their student’s records in a more efficient way, although there are some other third-party apps that can help schools to manage student’s records nowadays, but it is not really efficient and safe. Our application uses blockchain technology to make sure the data is safe and secure. Every step of the process that needs to be worked with records will be tracked by EduBlock, so the school can easily track the data changes and make sure the data is not being tampered.

### 3.1.2 Business Rules

| **ID** | **Rules Description** |
| --- | --- |
| BR-1 | The application will be used by the students, teachers, staff, admin and third-party. |
| BR-2 | Only Staff have permission to manage classroom including create new class, edit class, assign or remove teacher from the class, assign student to class, remove student from class, edit student information. |
| BR-3 | Only Admin have permission to create new account(s). |
| BR-4 | Only Teacher who teach the subject can edit the grade of the student. Homeroom teacher can request subject teacher to change grade. |
| BR-5 | Student can only view their class, profile and academic record. Student can only request to re-check their record. |
| BR-6 | In Blockchain, the data is immutable, so the data cannot be changed once it is created. Because of this, the data can just be appended, can’t be edited or deleted, this will help ensure the student’s record is safe and secure. |
| BR-7 | A node if wants to join the network, it must have other nodes permission or the node must be approved by the admin. |
| BR-8 | In a private blockchain, every node knows each other, which node own the data. Data is shared between nodes so the data can be recovered if one node is down. |
| BR-9 | Other nodes can only read the data, they cannot change the data. |
| BR-10 | Third-party’s member can only view the academic record and statistics of the students by using verified keys. |

## 3.2 User Requirements

* The Academic record management web-app has five active actors: Student, Teacher, Staff, Administrator and Third-party’s member (i.e., parents, etc.).
* Admin can create accounts for each role such as staff, student, teacher.
* Students can view their academic record.
* Teachers can manage their class and their student’s academic record.
* Staff can manage the classroom and view academic records of students, assign or delete teachers from the class, assign students to class, create new classes and edit student information.
* Third-party’s member can view the academic record and statistic of the students by using verified key.

### 3.2.1 System Actors

| **ID** | **Actor** | **Description** |
| --- | --- | --- |
| 1 | ADMIN | Admin is the person who has the highest authority in the system. Admin can manage the account of the other actors. |
| 2 | STAFF | Staff is the person who has the authority to manage the classroom. Staff can assign or remove teachers from the class, assign students to class and remove students from class. |
| 3 | TEACHER | Teacher is the person who has the authority to manage their class. Teachers can view their student’s academic record; subject teachers can send requests to edit student grade. |
| 4 | STUDENT | Student is the person who has the authority to view their academic record. |
| 5 | THIRD PARTY | Third party is the person who has the authority to view the academic record and statistics of the students by using verified a key. |

### 3.2.2 Use cases

#### 3.2.2.1 Diagrams

|  |
| --- |
| Figure 3.1: Use case diagram for Admin |

|  |
| --- |
| Figure 3.2: Use case diagram for Staff |

|  |
| --- |
| Figure 3.3: Use case diagram for Teacher |

|  |
| --- |
| Figure 3.4: Use case diagram for Student |

|  |
| --- |
| Figure 3.5: Use case diagram for Third Party |

#### 3.2.2.2 Details

[Use Case List](./docx/UseCase.docx)

## 3.3 Functional Requirements

### 3.3.1 System Functional Overview

The system is designed to provide a platform for school to manage their student’s record, information with high security, fast and private.

### 3.3.2 Features

#### 3.3.2.1 **Account Features**

* **User Login**
  + Use cases: UC-1, UC-13, UC-32, UC-47
  + Description: The system shall allow user to login to EduBlock using their account.
* **Create Account**
  + Use cases: UC-4
  + Description: System shall allow admin to create account for staff, teacher, student, parent.
* **View list of all accounts**
  + Use cases: UC-2, UC-14
  + Description: System shall allow admin and staff to view list of accounts.
* **View account’s detail**
  + Use cases: UC-3, UC-15, UC-23, UC-34, UC-40, UC-51
  + Description: The system shall allow user to view account’s detail.
* **Update profile**
  + Use cases: UC-6, UC-24, UC-29
  + Description: The system shall allow only admin and staff to update account’s profile.
* **Update password**
  + Use cases: UC-7, UC-29, UC-34
  + Description: The system shall allow user to reset their password.

#### 3.3.2.2 **Class Features**

* **View classroom list**
  + Use cases: UC-10, UC-17, UC-36, UC-48
  + Description: The system shall allow user to view list of classes filter by their role.
* **View classroom detail**
  + Use cases: UC-11, UC-19, UC-37, UC-49
  + Description: The system shall allow user to view class detail.
* **View student in class**
  + Use cases: UC-11, UC-21, UC-38
  + Description: The system shall allow user who have the right to view list of students in class.
* **View teacher in class**
  + Use cases: UC-11, UC-26, UC-39, UC-50
  + Description: The system shall allow user who have the right to view list of teachers in class.
* **Create classroom**
  + Use cases: UC-18
  + Description: The system shall allow staff to create new class
* **Update classroom detail**
  + Use cases: UC-20
  + Description: The system shall allow only staff to update class detail.
* **Add student to class**
  + Use cases: UC-22
  + Description: The system shall allow only staff to add student to class.
* **Assign teacher to class**
  + Use cases: UC-27
  + Description: The system shall allow only staff to assign teacher to class.
* **Remove student from class**
  + Use cases: UC-25
  + Description: The system shall allow only staff to remove student from class.
* **Remove teacher from class**
  + Use cases: UC-28
  + Description: The system shall allow only staff to remove teacher from class. #### **Record Features**
* **View student’s record** (view, print)
  + Use cases: UC-23, UC-30, UC-40, UC-41, UC-51, UC-52, UC-55
  + Description: The system shall allow user to view student’s record and print the record.
* **Send request to change or re-check student’s record**
  + Use cases: UC-46, UC-53
  + Description: The system shall allow student and homeroom teacher to send request to re-check or change their record.
* **View list of pending change requests** (view, approve, reject)
  + Use cases: UC-44
  + Description: The system shall allow homeroom teacher to view list of pending change requests and approve or reject the request.
* **View history of record’s changes**
  + Use cases: UC-45, UC-56
  + Description: The system shall allow user to view history of record’s changes.
* **Subject teacher changes their subject score**
  + Use cases: UC-42
  + Description: The system shall allow subject teacher to change their subject score on student’s record.

#### 3.3.2.3 **Student Key Features**

* **Student create verified key**
  + Use cases: UC-51
  + Description: The system shall allow student to create verified key for their parents to use it to view their academic profile and records.
* **Third party’s member view student’s academic profile and records**
  + Use cases: UC-57
  + Description: The system shall allow third party’s member to view student’s academic profile and records by using verified key given by the student.
* **Third party’s member view statistics of a grade in a year**
  + Use cases: UC-58
  + Description: The system shall allow third party’s member to view statistics of a grade in a year by using verified statistic key given by admin or staff.

## 3.4 Non-Functional Requirements

### 3.4.1 External Interfaces

#### 3.4.1.1 **User Interfaces**

* **UI-1:** The system shall provide a user interface for admin manage all accounts.
* **UI-2:** The system shall provide a user interface for staff to manage classes.
* **UI-3:** The system shall provide a user interface for teacher to view classes and manage students, student’s records.
* **UI-4:** The system shall provide a user interface for teacher to view list of requests to change student’s records.
* **UI-5:** The system shall provide a user interface for student to view classes and view their records.
* **UI-6:** The system shall provide a user interface for parent to view their children’s records.
* **UI-7:** The system shall provide a user interface for student to generate private key for their parents to view their records.
* **UI-8:** The system shall permit complete access to the system via a web browser.
* **UI-9:** The web-application shall permit complete navigation.
* **UI-10:** The web-application shall permit complete all functions.

#### 3.4.1.2 **Hardware Interfaces**

* **HI-1:** The web-app shall be able to run on any device that can run a web browser.
* **HI-2:** Graphic card is required to upload student’s academic record using image file.

#### 3.4.1.3 **Software Interfaces**

* **SI-1:** Hyperledger Fabric network.
  + **SI-1.1:** The system shall initialize decentralized network using Mini-fabric smoothly.
  + **SI-1.2:** The network shall install chaincode in all peers smoothly.
* **SI-2:** EduBlock client
* The request server shall communicate with user interface through API to perform following operations:
  + **SI-2.1:** The system shall allow user to login.
  + **SI-2.2:** The system shall allow user to view their profile.
  + **SI-2.3:** The system shall allow user to reset their password.
  + **SI-2.4:** The system shall allow Admin to perform CRUD operations on account.
  + **SI-2.5:** The system shall allow Staff to perform CRU operations on classes.
  + **SI-2.6:** Teacher to send request to change student’s academic record.
  + **SI-2.7:** Teacher to approve or reject request to change student’s academic record.
  + **SI-2.8:** Student to send request to re-check student’s academic record.
  + **SI-2.9:** Student to upload student’s academic record.

### 3.4.2 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.
  + The application is designed to be used by admin, staff, teacher, student, and third-party’s member.
* **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.

# 4. Software Design Description

## 4.1 Overall Description

### 4.1.1 Assumptions

* The target platform is a Docker-compatible operating system (Preferably Linux).
* The target web browser is Google Chrome.
* All products are run and operated in the same machine (Monolithic architecture).
* Only administrators can access the running system. Other users can only access the system through the frontend server.
* The blockchain is a private blockchain.
* The blockchain is an optional feature that can be turned on or off.
* The database is a H2 database.
* The mode of the database can be in-memory (for testing), file or remote (for production).

### 4.1.2 Design Constraints

* The backend system is a REST API server.
* The backend system is a Java application.
* The blockchain is a Hyperledger Fabric blockchain.
* There should be an option to turn on or off the blockchain. If the blockchain is turned off, the backend system should still work as a normal REST API server with a local database.

### 4.1.3 Technology Suggestion

* The endpoints of the backend system can be exposed so that a node browser can be developed to search & access the endpoints and get the necessary information.

## 4.2 System Architecture Design

### 4.2.1 Overall Architecture

|  |
| --- |
| Figure 4.1: Overall Architecture |

### 4.2.2 System Architecture

|  |
| --- |
| Figure 4.2: System Architecture |

### 4.2.3 Package Diagram

|  |
| --- |
| Figure 4.3: Package Diagram of Request Server |

Table 4.1: Package Detail of the 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 |

## 4.3 System Detailed Design

### 4.3.1 Class Specification

|  |
| --- |
| Figure 4.4: Class Diagram of the Request Server |

#### 4.3.1.1 Account

Table 4.2: Fields of Account

| Field Name | Type | Description |
| --- | --- | --- |
| id | long | The account id |
| username | String | The username |
| hashedPassword | String | The hashed password |
| salt | String | The salt of the password |
| role | String | The role of the account |
| createdAt | Date | The date when the account was created |
| classrooms | List ClassTeacher | The list of references to the classrooms that the account participates if its role is Teacher |
| recordEntries | List RecordEntry | The list of record entries related to the subjects that the account is teaching if its role is Teacher |
| requestedRecordEntries | List RecordEntry | The list of verified record entries that the account requested to changes |
| approvedRecordEntries | List RecordEntry | The list of verified record entries that the account accepted |
| pendingRecordEntries | List PendingRecordEntry | The list of pending record entries related to the subjects that the account is teaching if its role is Teacher |
| requestedPendingRecordEntries | List PendingRecordEntry | The list of pending record entries that the account requested to changes |
| homeClassrooms | List Classroom | The list of classrooms the the account is a homeroom teacher at |

#### 4.3.1.2 Profile

Table 4.3: Fields of Profile

| Field Name | Type | Description |
| --- | --- | --- |
| id | long | The account id |
| account | Account | The reference to the Account object |
| firstName | String | The first name |
| lastName | String | The last name |
| male | boolean | Is the person male? false if she is a female |
| avatar | String | The link to the avatar image |
| birthDate | Date | The date of the birthday |
| address | String | The address |
| phone | String | The phone number |
| email | String | The email |
| updated | boolean | The flag indicates that the profile requires synchronization with the Chain Node |

#### 4.3.1.3 Student

Table 4.4: Fields of Student

| Field Name | Type | Description |
| --- | --- | --- |
| id | long | The account id |
| account | Account | The reference to the Account object |
| ethnic | String | The ethnic of the student |
| fatherName | String | The name of the father of the student |
| fatherJob | String | The job of the father of the student |
| motherName | String | The name of the mother of the student |
| motherJob | String | The job of the mother of the student |
| guardianName | String | The name of the guardian of the student |
| guardianJob | String | The job of the guardian of the student |
| homeTown | String | The home town of the student |
| classrooms | List ClassStudent | The list of references to the classrooms that the student participates |
| records | List Record | The list of records related to the classrooms that the student participates |
| updaterKey | List UpdaterKey | The list of updater keys of the student. Used to allow outsiders to get information of the student. |

#### 4.3.1.4 Classroom

Table 4.5: Fields of Classroom

| Field Name | Type | Description |
| --- | --- | --- |
| id | long | The classroom id |
| name | String | The name of the classroom |
| grade | int | The grade of the classroom |
| year | int | The year of the classroom |
| homeroomTeacher | Account | The reference to the homeroom teacher of the classroom |
| students | List ClassStudent | The list of references to the students that participate in the classroom |
| teachers | List ClassTeacher | The list of references to the teachers that participate in the classroom |
| records | List Record | The list of records related to the classroom |

#### 4.3.1.5 ClassStudent

Table 4.6: Fields of ClassStudent

| Field Name | Type | Description |
| --- | --- | --- |
| id | long | The id of the reference |
| classroom | Classroom | The reference to the classroom |
| student | Student | The reference to the student |

#### 4.3.1.6 ClassTeacher

Table 4.7: Fields of ClassTeacher

| Field Name | Type | Description |
| --- | --- | --- |
| id | long | The id of the reference |
| classroom | Classroom | The reference to the classroom |
| teacher | Account | The reference to the teacher |
| subjectId | long | The id of the subject that the teacher teaches |

#### 4.3.1.7 Record

Table 4.8: Fields of Record

| Field Name | Type | Description |
| --- | --- | --- |
| id | long | The record id |
| classroom | Classroom | The reference to the classroom |
| student | Student | The reference to the student |
| recordEntry | List RecordEntry | The list of verified record entries related to the record |
| pendingRecordEntry | List PendingRecordEntry | The list of pending record entries related to the record |

#### 4.3.1.8 RecordEntry

Table 4.9: Fields of RecordEntry

| Field Name | Type | Description |
| --- | --- | --- |
| id | long | The record entry id |
| subjectId | long | The id of the subject that the record entry is related to |
| firstHalfScore | int | The score of the first semester of the subject |
| secondHalfScore | int | The score of the second semester of the subject |
| finalScore | int | The final score of the subject |
| requestDate | Date | The date when the record entry was requested |
| approvalDate | Date | The date when the record entry was approved |
| updateComplete | boolean | The flag indicates that the record entry was updated to the Chain Node |
| teacher | Account | The reference to the teacher that teaches the subject |
| requester | Account | The reference to the account that requested the record entry |
| approver | Account | The reference to the account that approved the record entry |
| record | Record | The reference to the record that the record entry is related to |

#### 4.3.1.9 PendingRecordEntry

Table 4.10: Fields of PendingRecordEntry

| Field Name | Type | Description |
| --- | --- | --- |
| id | long | The pending record entry id |
| subjectId | long | The id of the subject that the pending record entry is related to |
| firstHalfScore | int | The score of the first semester of the subject |
| secondHalfScore | int | The score of the second semester of the subject |
| finalScore | int | The final score of the subject |
| requestDate | Date | The date when the pending record entry was requested |
| teacher | Account | The reference to the teacher that teaches the subject |
| requester | Account | The reference to the account that requested the pending record entry |
| record | Record | The reference to the record that the pending record entry is related to |

#### 4.3.1.10 UpdaterKey

Table 4.11: Fields of UpdaterKey

| Field Name | Type | Description |
| --- | --- | --- |
| id | String | The unique key |
| student | Student | The reference to the student that the key is related to |

#### 4.3.1.11 StatisticKey

Table 4.12: Fields of StatisticKey

| Field Name | Type | Description |
| --- | --- | --- |
| id | String | The unique key |
| year | int | The year that the key is referred to |
| grade | int | The grade that the key is referred to |

### 4.3.2 Sequence Diagram

#### 4.3.2.1 Create Account

|  |
| --- |
| Figure 4.5: Sequence Diagram to Create Account |

#### 4.3.2.2 Get Account

|  |
| --- |
| Figure 4.6: Sequence Diagram to Get Account |

#### 4.3.2.3 Get Account List

|  |
| --- |
| Figure 4.7: Sequence Diagram to Get Account List |

#### 4.3.2.4 Login

|  |
| --- |
| Figure 4.8: Sequence Diagram to Login |

#### 4.3.2.5 Update Account Password

|  |
| --- |
| Figure 4.9: Sequence Diagram to Update Account Password |

#### 4.3.2.6 Update Account Profile

|  |
| --- |
| Figure 4.10: Sequence Diagram to Update Account Profile |

#### 4.3.2.7 Create Classroom

|  |
| --- |
| Figure 4.11: Sequence Diagram to Create Classroom |

#### 4.3.2.8 Add Students To Classroom

|  |
| --- |
| Figure 4.12: Sequence Diagram to Add Students To Classroom |

#### 4.3.2.9 Remove Students From Classroom

|  |
| --- |
| Figure 4.13: Sequence Diagram to Remove Students From Classroom |

#### 4.3.2.10 Add Teachers To Classroom

|  |
| --- |
| Figure 4.14: Sequence Diagram to Add Teachers To Classroom |

#### 4.3.2.11 Remove Teachers From Classroom

|  |
| --- |
| Figure 4.15: Sequence Diagram to Remove Teachers From Classroom |

#### 4.3.2.12 Update Classroom

|  |
| --- |
| Figure 4.16: Sequence Diagram to Update Classroom |

#### 4.3.2.13 Get Classroom

|  |
| --- |
| Figure 4.17: Sequence Diagram to Get Classroom |

#### 4.3.2.14 Get Classroom List

|  |
| --- |
| Figure 4.18: Sequence Diagram to Get Classroom List |

#### 4.3.2.15 Get Students In Classroom

|  |
| --- |
| Figure 4.19: Sequence Diagram to Get Students In Classroom |

#### 4.3.2.16 Get Teachers In Classroom

|  |
| --- |
| Figure 4.20: Sequence Diagram to Get Teachers In Classroom |

#### 4.3.2.17 Get Student Record



#### 4.3.2.18 Get Student Record List



#### 4.3.2.19 Update Student Record



#### 4.3.2.20 Create Request To Update Student Record



#### 4.3.2.21 Get Pending Record Requests



#### 4.3.2.22 Approve Pending Record Request



#### 4.3.2.23 Upload Record To Chain Node



#### 4.3.2.24 Create Statistic Key



#### 4.3.2.25 Get Statistic Key List



#### 4.3.2.26 Delete Statistic Key



#### 4.3.2.27 Get Statistic Data



#### 4.3.2.28 Create Student Key



#### 4.3.2.29 Get Student Key List



#### 4.3.2.30 Delete Student Key



#### 4.3.2.31 Get Student Data From Key



#### 4.3.2.32 Upload Legacy Student Record



## 4.4 Data & Database Design

### 4.4.1 Database Design

|  |
| --- |
| Figure 4.21: Database Design of the Request Server |

#### 4.4.1.1 Account

Table 4.13: Attributes of the Account table

| 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 |  |  |

#### 4.4.1.2 Profile

Table 4.14: Attributes of the Profile table

| 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 |

#### 4.4.1.3 Student

Table 4.15: Attributes of the Student table

| 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 |  |  |

#### 4.4.1.4 Classroom

Table 4.16: Attributes of the Classroom table

| 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 |  |  |

#### 4.4.1.5 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 |  |

#### 4.4.1.6 Class Teacher

Table 4.17: Attributes of the Class Teacher table

| 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 |

#### 4.4.1.7 Record

Table 4.18: Attributes of the Record table

| 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 |  |

#### 4.4.1.8 Record Entry

Table 4.19: Attributes of the Record Entry table

| 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 |

#### 4.4.1.9 Pending Record Entry

Table 4.20: Attributes of the Pending Record Entry table

| 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 |

#### 4.4.1.10 Updater Key

Table 4.21: Attributes of the Updater Key table

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

#### 4.4.1.11 Statistic Key

Table 4.22: Attributes of the Statistic Key table

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

### 4.4.2 Data File Design

Table 4.23: Data File Structure

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

# 5. Software Testing Documentation

## 5.1 Overall Description

### 5.1.1 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.

|  |
| --- |
| Figure 5.1: V-model (Herman Bruyninckx, CC BY-SA 3.0, via Wikimedia Commons) |

### 5.1.2 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.

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.

### 5.1.3 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.

## 5.2 Test Plan

### 5.2.1 Test Stages

[Report\_Test-Stages.xlsx](excel/Report_Test-Stages.xlsx)

### 5.2.2 Resources

#### 5.2.2.1 Human Resources

Table 5.1: Human Resources

| Worker/ Doer | Role | Specifice Responsibilities/Comments |
| --- | --- | --- |
| TienHQ | BE-Tester | Test if the Request Server is working properly or not |
| TuLX | FE-Tester | Test if the ChainCode is working properly or not |
| KhoaND | Reporter | Handling reports related work |
| UyCHA | BE-Tester | Test if the UI is working properly or not |
| KhoiNM | FE-Tester | Test if the UI is working properly or not |

#### 5.2.2.2 Environment

None

### 5.2.3 Test Milestones

Table 5.2: Test Milestones

| Milestone Task | Effort (md) | Start Date | End Date |
| --- | --- | --- | --- |
| ChainCode | 6 | Nov 9, 2022 | Nov 14, 2022 |
| Request Server | 20 | Nov 15, 2022 | Dec 04, 2022 |

### 5.2.4 Deliverables

Table 5.3: Deliverables

| No | Deliverables | Due Date |
| --- | --- | --- |
| 1 | Test Design | Nov 11, 2022 |
| 2 | ChainCode Test script | Nov 14, 2022 |
| 3 | Request Server Test script | Dec 15, 2022 |
| 4 | Test results | Dec 16, 2022 |

## 5.3 Test Cases

* Unit Test Cases: [Report\_Unit-Test-Case.xlsx](excel/Report_Unit-Test-Case.xlsx)
* Other Test Cases: [Report\_Test-Case-Document.xlsx](excel/Report_Test-Case-Document.xlsx)

## 5.4 Test Reports

Test Report has been fully integrated in Unit\_Test-Case and Test-Case-Document.

# 6. Release Package & User Guides

## 6.1 Deliverable Package

### 6.1.1 Source Codes & Documents

Table 6.1: Source code & Documents

| Items | Sub-Items | Type | Version |
| --- | --- | --- | --- |
| EduBlock | EduBlock Client | Code | 1.0 |
| Request Server | EduBlock Backend | Code | 3.0 |
| Blockchain Network | Blockchain Chaincode | Code | 1.0 |
| OCR | Record Table Processor | Code | 1.0 |
| Requirement | SRS.docx | Document | 1.0 |
| Deployment | UserGuide.docx | Document | 1.0 |
| Final Report | FinalReport.docx | Document | 1.0 |

### 6.1.2 Known Issues, Limitations & Restrictions

#### 6.1.2.1 Limitations

* Verified records coundn’t be synchronized between nodes.
* Verified keys for third-party can only be used in the node it was created.
* Currently network have ability to provide verified data with blockchain technology, and a node can be used as restore point if database of other nodes missing.

## 6.2 Installation Guides

### 6.2.1 System Requirements

OS: any

CPU: at least 4 cores

RAM: at least 4Gb

NETWORK: required

SOFTWARE: Docker

### 6.2.2 Setup Files

* docker-compose.yml
* .env

### 6.2.3 Installation Instruction

* Update value in the .env as required
* Run command docker-compose up to deploy program
* There are many solution to deploy a network. In this scenario, we deploy using virtual network. Other solution can be found [here](https://hyperledger-fabric.readthedocs.io/en/release-2.5/deployment_guide_overview.html)

## 6.3 User Manual

### 6.3.1 Terms & Definitions

Table 6.2: Terms & Definitions

| No. | Term | Definition |
| --- | --- | --- |
| 01 | FT | Feature |
| 02 | GUEST | Guest, Third-party |
| 03 | ADMIN | Admin |
| 04 | STAFF | Staff |
| 05 | TEACHER | Teacher |
| 06 | STUDENT | Student |
| 07 | AUTH | Any authenticated role |
| 08 | MAN | Admin or Staff |
| 09 | USER | Teacher or Student |
| 10 | PERSONAL | Personal |

### 6.3.2 System Requirements

OS: any

CPU: any

RAM: at least 1Gb

NETWORK: required

### 6.3.3 Application Usage

#### 6.3.3.1 Overview

Table 6.3: Feature Matrix

| No. | Feature | ADMIN | STAFF | TEACHER | STUDENT | GUEST | Note |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 01 | Login | x | x | x | x |  |  |
| 02 | View list of all accounts | x | x |  |  |  |  |
| 03 | View details of an account | x | x |  |  |  |  |
| 04 | Create new accounts | x |  |  |  |  |  |
| 05 | Search account | x | x |  |  |  |  |
| 06 | Update own profile | x | x | x | x |  |  |
| 07 | Change others account password | x |  |  |  |  |  |
| 08 | Get classification report of year and grade | x | x |  |  |  |  |
| 09 | Get report of a classroom | x | x | x |  |  |  |
| 10 | View list of all classrooms | x | x |  |  |  |  |
| 11 | View details of a classroom | x | x | x | x |  |  |
| 12 | Create new classroom |  | x |  |  |  |  |
| 13 | Update details of a classroom |  | x |  |  |  |  |
| 14 | View list of students of a classroom | x | x | x |  |  |  |
| 15 | Assign multiple students to a classroom |  | x |  |  |  |  |
| 16 | View details of a student | x | x | x |  |  |  |
| 17 | Update details of a student |  | x |  |  |  |  |
| 18 | Remove a student from classroom |  | x |  |  |  |  |
| 19 | View list of teachers of a classroom | x | x | x | x |  |  |
| 20 | Assign multiple teachers to a classroom |  | x |  |  |  |  |
| 21 | Remove a teacher from classroom |  | x |  |  |  |  |
| 22 | Print record table of a student | x | x | x | x |  |  |
| 23 | View own profile | x | x | x | x |  |  |
| 24 | Update own password | x | x | x | x |  |  |
| 25 | View teaching classrooms |  |  | x |  |  |  |
| 26 | Request update record of student |  |  | x | x |  |  |
| 27 | Request update record of student by image |  |  | x | x |  |  |
| 28 | Update a record of student |  |  | x |  |  |  |
| 29 | View list of pending record’s update requests |  |  | x |  |  |  |
| 30 | Approve or reject record update request |  |  | x |  |  |  |
| 31 | View update history of a record of student | x | x | x | x |  |  |
| 32 | View list of classroom taking part in |  |  |  | x |  |  |
| 33 | View own records |  |  |  | x |  |  |
| 34 | Create key to view records of a student |  |  |  | x |  |  |
| 35 | View list of keys to view records of a student |  |  |  | x |  |  |
| 36 | Remove a key to view records of a student |  |  |  | x |  |  |
| 37 | Create key to view records of students | x | x |  |  |  |  |
| 38 | View list of keys to view records of students | x | x |  |  |  |  |
| 39 | Remove a key to view records of students | x | x |  |  |  |  |
| 40 | View verified records of a student | x | x | x | x | x |  |
| 41 | View verified records of students | x | x | x | x | x |  |

#### 6.3.3.2 Login

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click Login in the homepage | Navigate to login page | [Figure 6.1](#fig-ug-01-step1) |
| 02 | Fill the form with provided credential |  | [Figure 6.2](#fig-ug-01-step2) |
| 03 | Click Login to submit the credential | Navigate to dashboard if credential is valid | [Figure 6.3](#fig-ug-01-step3) |

|  |
| --- |
| Figure 6.1: Home view |

|  |
| --- |
| Figure 6.2: Login view |

|  |
| --- |
| Figure 6.3: Login |

|  |
| --- |
| Figure 6.4: Login success |

#### 6.3.3.3 View list of all accounts

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Account in the left sidebar | Navigate to the account list page | [Figure 6.5](#fig-ug-02-step1) |

|  |
| --- |
| Figure 6.5: Account Navigation item |

|  |
| --- |
| Figure 6.6: Account list view |

#### 6.3.3.4 View details of an account

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Account in the left sidebar | Navigate to the account list page | [Figure 6.7](#fig-ug-03-step1) |
| 02 | Find the target account |  | [Figure 6.8](#fig-ug-03-step2) |
| 03 | Click on Details in the Actions column of the table | Navigate to the account’s profile page | [Figure 6.9](#fig-ug-03-step3) |

|  |
| --- |
| Figure 6.7: Account list view |

|  |
| --- |
| Figure 6.8: Find an account |

|  |
| --- |
| Figure 6.9: Button to view account profile |

|  |
| --- |
| Figure 6.10: Account profile view |

#### 6.3.3.5 Create new accounts

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 02 to navigate to the account list page |  |  |
| 01 | Click on Create to open the accounts creation form | A modal with form show up | [Figure 6.11](#fig-ug-04-step1) |
| 02 | Fill the form with desired values |  | [Figure 6.12](#fig-ug-04-step2) |
| 03 | Click on Create Accounts | New accounts added at the end of list | [Figure 6.13](#fig-ug-04-step3) |

|  |
| --- |
| Figure 6.11: Account list view |

|  |
| --- |
| Figure 6.12: Form to create account |

|  |
| --- |
| Figure 6.13: Confirm creation of provided account |

|  |
| --- |
| Figure 6.14: Account created |

#### 6.3.3.6 Search an account

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 02 to navigate to the account list page |  |  |
| 01 | Click on Search to show search input | Search input visible | [Figure 6.15](#fig-ug-05-step1) |
| 02 | Fill the input with desired values | List of accounts auto refreshed with search value | [Figure 6.16](#fig-ug-05-step2) |

|  |
| --- |
| Figure 6.15: Account list view |

|  |
| --- |
| Figure 6.16: Account search input |

|  |
| --- |
| Figure 6.17: Search result |

#### 6.3.3.7 Update own profile

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Account name in the left sidebar | Navigate to the profile page | [Figure 6.18](#fig-ug-06-step1) |
| 02 | Click on Update profile to open profile update form | A modal with form show up | [Figure 6.19](#fig-ug-06-step2) |
| 03 | Fill the form with desired values |  | [Figure 6.20](#fig-ug-06-step3) |
| 04 | Click on Submit | Update if form values is valid | [Figure 6.21](#fig-ug-06-step4) |

|  |
| --- |
| Figure 6.18: Profile navigation location |

|  |
| --- |
| Figure 6.19: Account profile view |

|  |
| --- |
| Figure 6.20: Profile update form |

|  |
| --- |
| Figure 6.21: Confirm profile information |

|  |
| --- |
| Figure 6.22: Profile updated |

#### 6.3.3.8 Change other’s password

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 02 to navigate to the account list page |  |  |
| 01 | Find the target account |  | [Figure 6.23](#fig-ug-07-step1) |
| 02 | Click on Update password in the Actions column | A modal with form show up | [Figure 6.24](#fig-ug-07-step2) |
| 03 | Fill the form with desired values |  | [Figure 6.25](#fig-ug-07-step3) |
| 04 | Click on Update | Password for target account updated | [Figure 6.26](#fig-ug-07-step4) |

|  |
| --- |
| Figure 6.23: Select an account from list |

|  |
| --- |
| Figure 6.24: Choose update target account information |

|  |
| --- |
| Figure 6.25: Password update form |

|  |
| --- |
| Figure 6.26: Confirm update with new password |

#### 6.3.3.9 Get classification report of grade and year

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Dashboard in the left sidebar | Navigate to the dashboard page | [Figure 6.27](#fig-ug-08-step1) |
| 02 | Select grade and year |  | [Figure 6.28](#fig-ug-08-step2) |
| 03 | Click on Get grade report or Get classification report | Report file can be download if exist | [Figure 6.29](#fig-ug-08-step3) |

|  |
| --- |
| Figure 6.27: Dashboard navigation item |

|  |
| --- |
| Figure 6.28: Select range for report |

|  |
| --- |
| Figure 6.29: Download report for selected range |

|  |
| --- |
| Figure 6.30: Report ready to download |

#### 6.3.3.10 Get report of a classroom

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Classroom in the left sidebar | Navigate to the classroom list page | [Figure 6.31](#fig-ug-09-step1) |
| 02 | Find the target classroom |  | [Figure 6.32](#fig-ug-09-step2) |
| 03 | Click on Details in the Actions column | Navigate to the classroom details page | [Figure 6.33](#fig-ug-09-step3) |
| 04 | Click on Get semester report or Get subject report | Report file can be download | [Figure 6.34](#fig-ug-09-step4) |

|  |
| --- |
| Figure 6.31: Classroom list navigation item |

|  |
| --- |
| Figure 6.32: Select a classroom |

|  |
| --- |
| Figure 6.33: View details of selected classroom |

|  |
| --- |
| Figure 6.34: Get report of selected classroom |

|  |
| --- |
| Figure 6.35: Report ready to download |

#### 6.3.3.11 View list of all classrooms

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Classroom in the left sidebar | Navigate to the classroom list page | [Figure 6.36](#fig-ug-10-step1) |

|  |
| --- |
| Figure 6.36: Classroom list navigation item |

|  |
| --- |
| Figure 6.37: Classroom list view |

#### 6.3.3.12 View details of a classroom

##### 6.3.3.12.1 MAN

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 10 to navigate to the classroom list page |  |  |
| 01 | Find the target classroom |  | [Figure 6.38](#fig-ug-11-step1-rman) |
| 02 | Click on Details in the Actions column | Navigate to the classroom details page | [Figure 6.39](#fig-ug-11-step2-rman) |

|  |
| --- |
| Figure 6.38: Select a classroom from classroom list |

|  |
| --- |
| Figure 6.39: Navigate to selected classroom details |

|  |
| --- |
| Figure 6.40: Details of selected classroom |

##### 6.3.3.12.2 USER

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Dashboard in the left sidebar | Navigate to the dashboard page | [Figure 6.41](#fig-ug-11-step1-rusr) |
| 02 | Find the target classroom |  | [Figure 6.42](#fig-ug-11-step2-rusr) |
| 03 | Click on Details in the Actions column | Navigate to the classroom details page | [Figure 6.43](#fig-ug-11-step3-rusr) |

|  |
| --- |
| Figure 6.41: Dashboard navigation item |

|  |
| --- |
| Figure 6.42: Select classroom from classroom list |

|  |
| --- |
| Figure 6.43: Navigate to classroom details |

|  |
| --- |
| Figure 6.44: Details of selected classroom |

#### 6.3.3.13 Create new classroom

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 10 to navigate to classroom list page |  |  |
| 01 | Click on Create | A modal with form show up | [Figure 6.45](#fig-ug-12-step1) |
| 02 | Fill the form with desired value |  | [Figure 6.46](#fig-ug-12-step2) |
| 03 | Click on Create classroom | New classroom created | [Figure 6.47](#fig-ug-12-step3) |

|  |
| --- |
| Figure 6.45: Classroom list view |

|  |
| --- |
| Figure 6.46: New classroom creation form |

|  |
| --- |
| Figure 6.47: Confirm new classroom creation |

#### 6.3.3.14 Update details of a classroom

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 11 to navigate to target classroom details |  |  |
| 01 | Click on Update details | A modal with form show up | [Figure 6.48](#fig-ug-13-step1) |
| 02 | Fill the form with desired value |  | [Figure 6.49](#fig-ug-13-step2) |
| 03 | Click on Update | Classroom details updated | [Figure 6.50](#fig-ug-13-step3) |

|  |
| --- |
| Figure 6.48: Classroom details view |

|  |
| --- |
| Figure 6.49: Form to update classroom details |

|  |
| --- |
| Figure 6.50: Confirm update new information for current classroom |

#### 6.3.3.15 View list of students of a classroom

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 11 to navigate to target classroom details |  |  |
| 01 | Click on Students |  | [Figure 6.51](#fig-ug-14-step1) |

|  |
| --- |
| Figure 6.51: Classroom details view |

|  |
| --- |
| Figure 6.52: Classroom student view |

#### 6.3.3.16 Assign multiple students to classroom

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 14 to navigate to student list of target classroom |  |  |
| 01 | Click on Add students | A modal with form show up | [Figure 6.53](#fig-ug-15-step1) |
| 02 | Fill the form with desired value |  | [Figure 6.54](#fig-ug-15-step2) |
| 03 | Click on Confirm | New student appear in target classroom | [Figure 6.55](#fig-ug-15-step3) |

|  |
| --- |
| Figure 6.53: Classroom student list |

|  |
| --- |
| Figure 6.54: Form to add new students to classroom |

|  |
| --- |
| Figure 6.55: Confirm add selected students to classroom |

#### 6.3.3.17 View details of a student

##### 6.3.3.17.1 MAN

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 02 to navigate to list of accounts |  |  |
| 01 | Find the target student |  | [Figure 6.56](#fig-ug-16-step1-rman) |
| 02 | Click on Details on the Actions column | Navigate to the target student profile | [Figure 6.57](#fig-ug-16-step2-rman) |

|  |
| --- |
| Figure 6.56: Select a student account |

|  |
| --- |
| Figure 6.57: Navigate to selected student profile |

|  |
| --- |
| Figure 6.58: Student profile view |

##### 6.3.3.17.2 TEACHER

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 14 to navigate to student list of a classroom |  |  |
| 01 | Find the target student |  | [Figure 6.59](#fig-ug-16-step1-rtch) |
| 02 | Click on Details on the Actions column | Navigate to the target student profile | [Figure 6.60](#fig-ug-16-step2-rtch) |

|  |
| --- |
| Figure 6.59: Select a student in classroom |

|  |
| --- |
| Figure 6.60: Navigate to selected student profile |

|  |
| --- |
| Figure 6.61: Student profile view |

#### 6.3.3.18 Update details of a student

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 16 to navigate to student profile |  |  |
| 01 | Click on Update | A modal with form show up | [Figure 6.62](#fig-ug-17-step1) |
| 02 | Fill the form with desired values |  | [Figure 6.63](#fig-ug-17-step2) |
| 03 | Click on Confirm | Student information updated if provided form valid | [Figure 6.64](#fig-ug-17-step3) |

|  |
| --- |
| Figure 6.62: Student profile view |

|  |
| --- |
| Figure 6.63: Form to update student information |

|  |
| --- |
| Figure 6.64: Confirm update new information |

#### 6.3.3.19 Remove a student from classroom

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 14 to navigate to student list of a classroom |  |  |
| 01 | Find the target student |  | [Figure 6.65](#fig-ug-18-step1) |
| 02 | Click on Remove on the Actions column | Target student removed from classroom | [Figure 6.66](#fig-ug-18-step2) |

|  |
| --- |
| Figure 6.65: Select a student in classroom |

|  |
| --- |
| Figure 6.66: Remove selected student from classroom |

#### 6.3.3.20 View list of teachers of a classroom

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 11 to navigate to classroom details |  |  |
| 01 | Click on Teachers |  | [Figure 6.67](#fig-ug-19-step1) |

|  |
| --- |
| Figure 6.67: Classroom details view |

|  |
| --- |
| Figure 6.68: Classroom teacher list view |

#### 6.3.3.21 Assign multiple teachers to a classroom

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 19 to navigate to teacher list of a classroom |  |  |
| 01 | Click on Add teachers | A modal with form show up | [Figure 6.69](#fig-ug-20-step1) |
| 02 | Fill the form with desired values |  | [Figure 6.70](#fig-ug-20-step2) |
| 03 | Click on Confirm | New teachers added to target classroom | [Figure 6.71](#fig-ug-20-step3) |

|  |
| --- |
| Figure 6.69: Classroom teacher list view |

|  |
| --- |
| Figure 6.70: Form to add new teachers to classroom |

|  |
| --- |
| Figure 6.71: Confirm add selected teacher to classroom |

#### 6.3.3.22 Remove a teacher from classroom

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 19 to navigate to teacher list of a classroom |  |  |
| 01 | Find the target teacher |  | [Figure 6.72](#fig-ug-21-step1) |
| 02 | Click on Remove | Target teacher removed from classroom | [Figure 6.73](#fig-ug-21-step2) |

|  |
| --- |
| Figure 6.72: Select a teacher in classroom |

|  |
| --- |
| Figure 6.73: Remove selected teacher in classroom |

#### 6.3.3.23 Print record table of a student

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 16 to navigate to student profile |  |  |
| 01 | Select a classroom of the target student |  | [Figure 6.74](#fig-ug-22-step1) |
| 02 | Click on Print record | Print dialog show up | [Figure 6.75](#fig-ug-22-step2) |

|  |
| --- |
| Figure 6.74: Student profile view |

|  |
| --- |
| Figure 6.75: Student record view of a classroom |

|  |
| --- |
| Figure 6.76: Record table ready to be print |

#### 6.3.3.24 View own profile

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Account name in the left sidebar | Navigate to the profile page | [Figure 6.77](#fig-ug-23-step1) |

|  |
| --- |
| Figure 6.77: Account profile navigation item |

|  |
| --- |
| Figure 6.78: Account profile page |

#### 6.3.3.25 Update own password

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 23 to navigate to profile page |  |  |
| 01 | Click on Change password | A modal with form appear | [Figure 6.79](#fig-ug-24-step1) |
| 02 | Fill the form with desired values |  | [Figure 6.80](#fig-ug-24-step2) |
| 03 | Click on Submit | Password updated | [Figure 6.81](#fig-ug-24-step3) |

|  |
| --- |
| Figure 6.79: Account profile view |

|  |
| --- |
| Figure 6.80: Form to update password |

|  |
| --- |
| Figure 6.81: Confirm update to new password |

#### 6.3.3.26 View teaching classroom

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Dashboard | Show list of teaching classroom | [Figure 6.82](#fig-ug-25-step1) |

|  |
| --- |
| Figure 6.82: Classroom list view of teacher |

#### 6.3.3.27 Request record update

##### 6.3.3.27.1 TEACHER

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 25 to navigate to teaching classroom list |  |  |
| 01 | Click Details in the Actions column | Navigate to classroom details | [Figure 6.83](#fig-ug-26-step1-rtch) |
| 02 | Click Students | Navigate to classroom student list | [Figure 6.84](#fig-ug-26-step2-rtch) |
| 03 | Find the target student |  | [Figure 6.85](#fig-ug-26-step3-rtch) |
| 04 | Click on Details in the Actions column | Navigate to student record of current classroom | [Figure 6.86](#fig-ug-26-step4-rtch) |
| 05 | Find the target record |  | [Figure 6.87](#fig-ug-26-step5-rtch) |
| 06 | Click on Request in the Actions column | A modal with form show up | [Figure 6.88](#fig-ug-26-step6-rtch) |
| 07 | Fill the form with desired values |  | [Figure 6.89](#fig-ug-26-step7-rtch) |
| 08 | Click on Request | Request sent | [Figure 6.90](#fig-ug-26-step8-rtch) |

|  |
| --- |
| Figure 6.83: Navigate to selected classroom details |

|  |
| --- |
| Figure 6.84: Navigate to student list of selected classroom |

|  |
| --- |
| Figure 6.85: Select a student from list of students |

|  |
| --- |
| Figure 6.86: Navigate to selected student profile |

|  |
| --- |
| Figure 6.87: Select a record from record table |

|  |
| --- |
| Figure 6.88: Request update selected record |

|  |
| --- |
| Figure 6.89: Form to request record update |

|  |
| --- |
| Figure 6.90: Confirm request details to be send |

##### 6.3.3.27.2 STUDENT

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 23 to navigate to profile page |  |  |
| 01 | Find the target classroom | Record data switch to target classroom | [Figure 6.91](#fig-ug-26-step1-rstd) |
| 02 | Find the target record |  | [Figure 6.92](#fig-ug-26-step2-rstd) |
| 03 | Click on Request in the Actions column | A modal with form show up | [Figure 6.93](#fig-ug-26-step3-rstd) |
| 04 | Fill the form with desired values |  | [Figure 6.94](#fig-ug-26-step4-rstd) |
| 05 | Click on Request | Request sent | [Figure 6.95](#fig-ug-26-step5-rstd) |

|  |
| --- |
| Figure 6.91: Select a classroom in personal profile |

|  |
| --- |
| Figure 6.92: Select a record in the record table |

|  |
| --- |
| Figure 6.93: Request update selected record |

|  |
| --- |
| Figure 6.94: Form to request record update |

|  |
| --- |
| Figure 6.95: Confirm request details to send |

#### 6.3.3.28 Request record update by legacy record image

##### 6.3.3.28.1 TEACHER

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 25 to navigate to teaching classroom list |  |  |
| 01 | Click Details in the Actions column | Navigate to classroom details | [Figure 6.96](#fig-ug-27-step1-rtch) |
| 02 | Click Students | Navigate to classroom student list | [Figure 6.97](#fig-ug-27-step2-rtch) |
| 03 | Find the target student |  | [Figure 6.98](#fig-ug-27-step3-rtch) |
| 04 | Click on Details in the Actions column | Navigate to student record of current classroom | [Figure 6.99](#fig-ug-27-step4-rtch) |
| 05 | Click on Upload next to Record Heading | A modal with form show up | [Figure 6.100](#fig-ug-27-step5-rtch) |
| 06 | Fill the form with desired values |  | [Figure 6.101](#fig-ug-27-step6-rtch) |
| 07 | Click on Submit | Request sent | [Figure 6.102](#fig-ug-27-step7-rtch) |

|  |
| --- |
| Figure 6.96: Navigate to selected classroom |

|  |
| --- |
| Figure 6.97: Navigate to student list of the classroom |

|  |
| --- |
| Figure 6.98: Select a student in student list |

|  |
| --- |
| Figure 6.99: Navigate to selected student profile |

|  |
| --- |
| Figure 6.100: Upload legacy record table |

|  |
| --- |
| Figure 6.101: Form to upload image of record table |

|  |
| --- |
| Figure 6.102: Confirm values to be updated |

##### 6.3.3.28.2 STUDENT

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 23 to navigate to profile page |  |  |
| 01 | Find the target classroom | Record data switch to target classroom | [Figure 6.103](#fig-ug-27-step1-rstd) |
| 02 | Click on Upload next to Record Heading | A modal with form show up | [Figure 6.104](#fig-ug-27-step2-rstd) |
| 03 | Fill the form with desired values |  | [Figure 6.105](#fig-ug-27-step3-rstd) |
| 04 | Click on Submit | Request sent | [Figure 6.106](#fig-ug-27-step4-rstd) |

|  |
| --- |
| Figure 6.103: Select a classroom in student profile view |

|  |
| --- |
| Figure 6.104: Upload legacy record table |

|  |
| --- |
| Figure 6.105: Form to upload image of record table |

|  |
| --- |
| Figure 6.106: Confirm values to be updated |

#### 6.3.3.29 Update a record of student

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 25 to navigate to teaching classroom list |  |  |
| 01 | Click Details in the Actions column | Navigate to classroom details | [Figure 6.107](#fig-ug-28-step1) |
| 02 | Click Students | Navigate to classroom student list | [Figure 6.108](#fig-ug-28-step2) |
| 03 | Find the target student |  | [Figure 6.109](#fig-ug-28-step3) |
| 04 | Click on Details in the Actions column | Navigate to student record of current classroom | [Figure 6.110](#fig-ug-28-step4) |
| 05 | Find the target record |  | [Figure 6.111](#fig-ug-28-step5) |
| 06 | Click on Request in the Actions column | A modal with form show up | [Figure 6.112](#fig-ug-28-step6) |
| 07 | Fill the form with desired values |  | [Figure 6.113](#fig-ug-28-step7) |
| 08 | Click on Request | Record updated | [Figure 6.114](#fig-ug-28-step8) |

|  |
| --- |
| Figure 6.107: Navigate to selected classroom |

|  |
| --- |
| Figure 6.108: Navigate to student list of classroom |

|  |
| --- |
| Figure 6.109: Select a student in student list |

|  |
| --- |
| Figure 6.110: Navigate to selected student profile |

|  |
| --- |
| Figure 6.111: Select a record from record table |

|  |
| --- |
| Figure 6.112: Request update seleted record |

|  |
| --- |
| Figure 6.113: Form to request update record |

|  |
| --- |
| Figure 6.114: Send request to teacher |

#### 6.3.3.30 View list of pending record update requests

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Pending request in the Actions column | Navigate to request list | [Figure 6.115](#fig-ug-29-step1) |

|  |
| --- |
| Figure 6.115: List of pending record update request |

#### 6.3.3.31 Approve or reject record update request

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 29 to navigate to request list page |  |  |
| 01 | Find the target request |  | [Figure 6.116](#fig-ug-30-step1) |
| 02 | Click Approve or Reject the request | Request status changed | [Figure 6.117](#fig-ug-30-step2) |

|  |
| --- |
| Figure 6.116: Select a pending request |

|  |
| --- |
| Figure 6.117: Accept or reject the request |

#### 6.3.3.32 View update history of a record of student

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 16 to navigate to student profile |  |  |
| 01 | Find the target classroom | Record data switch to target classroom | [Figure 6.118](#fig-ug-31-step1) |
| 02 | Find the target record |  | [Figure 6.119](#fig-ug-31-step2) |
| 03 | Hover History in the Actions column | A temporary card show update history | [Figure 6.120](#fig-ug-31-step3) |

|  |
| --- |
| Figure 6.118: Select a classroom in the student profile |

|  |
| --- |
| Figure 6.119: Select a record in the record table |

|  |
| --- |
| Figure 6.120: History view of selected record |

#### 6.3.3.33 View list of classroom taking part in

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Dashboard | Navigate to classroom list | [Figure 6.121](#fig-ug-32-step1) |

|  |
| --- |
| Figure 6.121: Classroom list view of student |

#### 6.3.3.34 View own records

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 23 to navigate to own record page |  |  |

#### 6.3.3.35 Create key to view records of a student

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Manage verified keys | Navigate to verified key list page | [Figure 6.122](#fig-ug-34-step1) |
| 02 | Click on Create new key | new key created | [Figure 6.123](#fig-ug-34-step2) |

|  |
| --- |
| Figure 6.122: Verified key list navigation item |

|  |
| --- |
| Figure 6.123: Create new verified key for third-party |

#### 6.3.3.36 View list of keys to view records of a student

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Manage verified keys | Navigate to verified key list page | [Figure 6.124](#fig-ug-35-step1) |

|  |
| --- |
| Figure 6.124: Verified key list navigation item |

#### 6.3.3.37 Remove a key to view records of a student

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 35 to navigate to the list of verified key page |  |  |
| 01 | Find the target key |  | [Figure 6.125](#fig-ug-36-step1) |
| 01 | Click on Remove | Target key removed | [Figure 6.126](#fig-ug-36-step2) |

|  |
| --- |
| Figure 6.125: Select a verified key in list |

|  |
| --- |
| Figure 6.126: Remove selected key |

#### 6.3.3.38 Create key to view records of students

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Manage stats key list | Navigate to verified key list page | [Figure 6.127](#fig-ug-37-step1) |
| 02 | Select year and grade for the key |  | [Figure 6.128](#fig-ug-37-step2) |
| 02 | Click on Create new key | new key created | [Figure 6.129](#fig-ug-37-step3) |

|  |
| --- |
| Figure 6.127: Access key list view for third-party |

|  |
| --- |
| Figure 6.128: Input grade and year for new key |

|  |
| --- |
| Figure 6.129: Confirm create new key with provided grade and year |

#### 6.3.3.39 View list of keys to view records of students

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Click on Manage stats key list | Navigate to verified key list page | [Figure 6.130](#fig-ug-38-step1) |

|  |
| --- |
| Figure 6.130: Access key list view for third-party |

#### 6.3.3.40 Remove a key to view records of students

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 00 | Follow Feature 38 to navigate to the list of verified key page |  |  |
| 01 | Find the target key |  | [Figure 6.131](#fig-ug-39-step1) |
| 01 | Click on Remove | Target key removed | [Figure 6.132](#fig-ug-39-step2) |

|  |
| --- |
| Figure 6.131: Select an access key in the key list |

|  |
| --- |
| Figure 6.132: Remove selected access key |

#### 6.3.3.41 View verified records of a student

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Press Ctrl + K | A command pallette show up | [Figure 6.133](#fig-ug-40-step1) |
| 02 | Select View verified data | Navigate to verified data page | [Figure 6.134](#fig-ug-40-step2) |
| 02 | Input key | Show verified data of the provided key | [Figure 6.135](#fig-ug-40-step3) |

|  |
| --- |
| Figure 6.133: Open command palette |

|  |
| --- |
| Figure 6.134: Select view records of a student |

|  |
| --- |
| Figure 6.135: Provide the verified key |

|  |
| --- |
| Figure 6.136: Show the verified key information |

#### 6.3.3.42 View verified records of students

| Step | Description | Effect | Figure |
| --- | --- | --- | --- |
| 01 | Press Ctrl + K | A command pallette show up | [Figure 6.137](#fig-ug-41-step1) |
| 02 | Select View verified data list | Navigate to verified data page | [Figure 6.138](#fig-ug-41-step2) |
| 02 | Input key | Show verified data of the provided key | [Figure 6.139](#fig-ug-41-step3) |

|  |
| --- |
| Figure 6.137: Open command palette |

|  |
| --- |
| Figure 6.138: Select view records of multiple student |

|  |
| --- |
| Figure 6.139: Provide the access key |

|  |
| --- |
| Figure 6.140: Show the verified key data |

### 6.3.4 Troubleshooting

# Appendix

## Acceptance Note

|  |
| --- |
| Figure 6.141: Acceptance Note |

## References

[1] M. Walport, Distributed ledger technology: Beyond Blockchain, UK Government Office for Science (2016).

[2] Duong-Trung, N., Son, H. X., Le, H. T., & Phan, T. T. (2020). Smart Care. Proceedings of the 2020 4th International Conference on Cryptography, Security and Privacy. https://doi.org/10.1145/3377644.3377667.

[3] Liang, Xueping & Shetty, Sachin & Tosh, Deepak & Kamhoua, Charles & Kwiat, Kevin & Njilla, Laurent. (2017). ProvChain: A Blockchain-Based Data Provenance Architecture in Cloud Environment with Enhanced Privacy and Availability. 10.1109/CCGRID.2017.8.

[4] Vanita Jain, Akanshu Raj & Abhishek Tanwar & Mridul Khurana. “eVote – A Decentralized Voting Platform”. Digital Twin Technology, CRC Press, 2021.

[5] Mettler, M. (2016). Blockchain technology in healthcare: The revolution starts here. 2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom). doi:10.1109/healthcom.2016.7749510.

[6] Samaniego, M., & Deters, R. (2016). Blockchain as a Service for IoT. 2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). doi:10.1109/ithings-greencom-cpscom-smartdata.2016.102.

[7] Chowdhury, M. , Suchana, K. , Alam, S. and Khan, M. (2021) Blockchain Application in Banking System. Journal of Software Engineering and Applications, 14, 298-311. doi: 10.4236/jsea.2021.147018.

[8] Ghaffar, A., & Hussain, M. (2019). BCEAP - A Blockchain Embedded Academic Paradigm to Augment Legacy Education through Application. Proceedings of the 3rd International Conference on Future Networks and Distributed Systems - ICFNDS ’19. doi:10.1145/3341325.3342036.

[9] Vidal, F. R., Gouveia, F., & Soares, C. (2020). Revocation Mechanisms for Academic Certificates Stored on a Blockchain. 2020 15th Iberian Conference on Information Systems and Technologies (CISTI). doi:10.23919/cisti49556.2020.9141088.

[10] M. Sharples and J. Domingue, “The Blockchain and Kudos: A Distributed System for Educational Record, Reputation and Reward,” in 11th European Conference on Technology Enhanced Learning, Lyon, France, 2016.