Student Management System

By Ihor Melashchenko, Marvin Adorian Zanchi Dos Santos, and Emmanuel Abolade

Description

Project Description

The Student Registration and Management System (SRMS) is a straightforward and efficient platform designed to facilitate the registration of students for courses and manage student and course information. It provides essential functionalities for staff members to add, view, update, and delete student and course details. Access to the system is restricted to authorized users, who must log in to perform operations and can securely log out when their tasks are complete.

Key features

1. User Authentication:
   Secure login mechanism requiring valid credentials (username and password) for access. User authentication ensures that only authorized staff members can access the system.

2. Student Registration:
   Staff members can add new student details, including name, ID number, contact information, and course preferences. The system validates student information to ensure accuracy and completeness during registration.

3. Course Management:
   Staff members can perform CRUD (Create, Read, Update, Delete) operations on course records. They can add new courses, view existing course details, update course information, and delete courses as needed.

4. Student Information Management:
   CRUD operations are available for managing student records. Staff members can view student details, update information such as contact numbers or addresses, and delete student records if necessary.

5. Logging In and Out: Staff members must log in with their credentials to access the system and perform operations. A secure logout feature allows users to end their session and securely exit the system when their tasks are complete.

Benefits

- Simplicity: The system offers a straightforward interface with intuitive functionalities, making it easy for staff members to navigate and perform tasks efficiently.
- Efficiency: By streamlining the student registration process and providing easy access to course and student information, the SRMS saves time and reduces administrative overhead.
- Data Integrity: Validation mechanisms ensure that only accurate and complete information is entered into the system, maintaining data integrity and reliability.
- Security: User authentication and secure login mechanisms protect sensitive student and course data from unauthorized access.
- Convenience: Staff members can access the system from any location with internet access, enabling them to manage student and course information conveniently..

Summary

The Student Registration and Management System (SRMS) provides a simple yet effective solution for educational institutions to manage student registrations and course information efficiently. With its user-friendly interface and essential functionalities, it serves as a valuable tool for enhancing administrative processes and ensuring data accuracy and security.

Requirements

System Requirements Document: Student Registration and Management System (SRMS)

The Student Registration and Management System (SRMS) is a Java-based application with Swing GUI designed to facilitate student registration for courses and manage student and course information within an educational institution. This document outlines the functional and non-functional requirements of the SRMS, focusing on its implementation using Java and Swing.

Functional Requirements:

1. User Authentication

- The system shall provide a secure login mechanism implemented in Java to authenticate staff members accessing the SRMS.

- Users shall be required to enter a username and password using Swing GUI components for authentication.

- The system shall validate user credentials against a predefined list of authorized users stored in a secure database.

2. Student Registration

- Staff members shall be able to add new student details using Swing GUI forms, including name, ID number, contact information such as email and home address, date of birth and course preferences.

- The system shall validate student information input through Swing GUI components to ensure accuracy and completeness during registration.

- Staff members shall have the ability to view and update student registration details through intuitive Swing GUI interfaces.

3. Course Management

- Staff members shall be able to perform CRUD operations on course records using Swing GUI interfaces, including adding, viewing, updating, and deleting courses.

- The system will validate course information entered through Swing GUI forms to prevent duplicate or incomplete entries.

- Staff members will have access to course details such as course code, title, level, category, duration of course, description, etc presented in Swing GUI components.

4. Student Information Management

- Staff members shall be able to perform CRUD operations on student records using Swing GUI interfaces, including viewing, updating, and deleting student information.

- The system will maintain a record of student details presented in Swing GUI forms, including personal information, contact details, and course enrolment status.

5. Logging In and Out

- The system shall provide a secure logout feature implemented in Java Swing to allow users to end their session and securely exit the SRMS.

- Users shall be automatically logged out after a specified period of inactivity to ensure security using Swing GUI components for session management.

Non-Functional Requirements

1. Performance

- The system shall be implemented in Java with efficient algorithms and data structures to ensure optimal performance.

- Response times for user interactions with Swing GUI components shall be kept within acceptable limits to provide a seamless user experience.

2. Security

- User authentication shall be implemented using secure encryption algorithms (hashing) in Java to protect user credentials during login.

- Access to sensitive student and course data shall be restricted to authorized users only, enforced through Java-based security mechanisms.

- The system shall maintain logs of user activities for auditing and security purposes, with log data stored securely in the system database.

3. Usability

- The Swing GUI interface will be designed to be intuitive and user-friendly, with clear navigation and layout for ease of use.

- Error messages will be displayed in user-friendly dialogs to assist users in resolving issues promptly and effectively.

4. Reliability

- The SRMS shall be implemented with error handling mechanisms in Java and SQL to minimize downtime and ensure high availability.

- Regular backups of the system database shall be performed to prevent data loss in the event of system failure or corruption.

5. Compatibility:

- The SRMS shall be compatible with Java Runtime Environment (JRE) versions supported by Swing GUI for seamless deployment across different platforms. There will provision for the executable jar file.

- The system will be tested for compatibility with commonly used web browsers to ensure consistent performance of Swing GUI components.

4. Implementation Constraints:

- The SRMS shall be developed using Java programming language and Swing GUI framework in adherence to the institution's technology stack requirements.

- Integration with existing student information systems or databases shall be implemented using Java Database Connectivity (JDBC) for data synchronization and interoperability.

5. Legal and Regulatory Requirements:

- The SRMS shall comply with relevant data protection regulations, including but not limited to GDPR (General Data Protection Regulation) and FERPA (Family Educational Rights and Privacy Act).

- Data privacy and security measures shall be implemented in Java to safeguard sensitive student and course information as per legal requirements.

6. Acceptance Criteria:

- The SRMS shall undergo thorough testing to ensure that all functional and non-functional requirements are met.

- User acceptance testing (UAT) shall be conducted with representatives from the institution to validate the usability, functionality, and performance of the Java Swing-based SRMS.

7. Glossary:

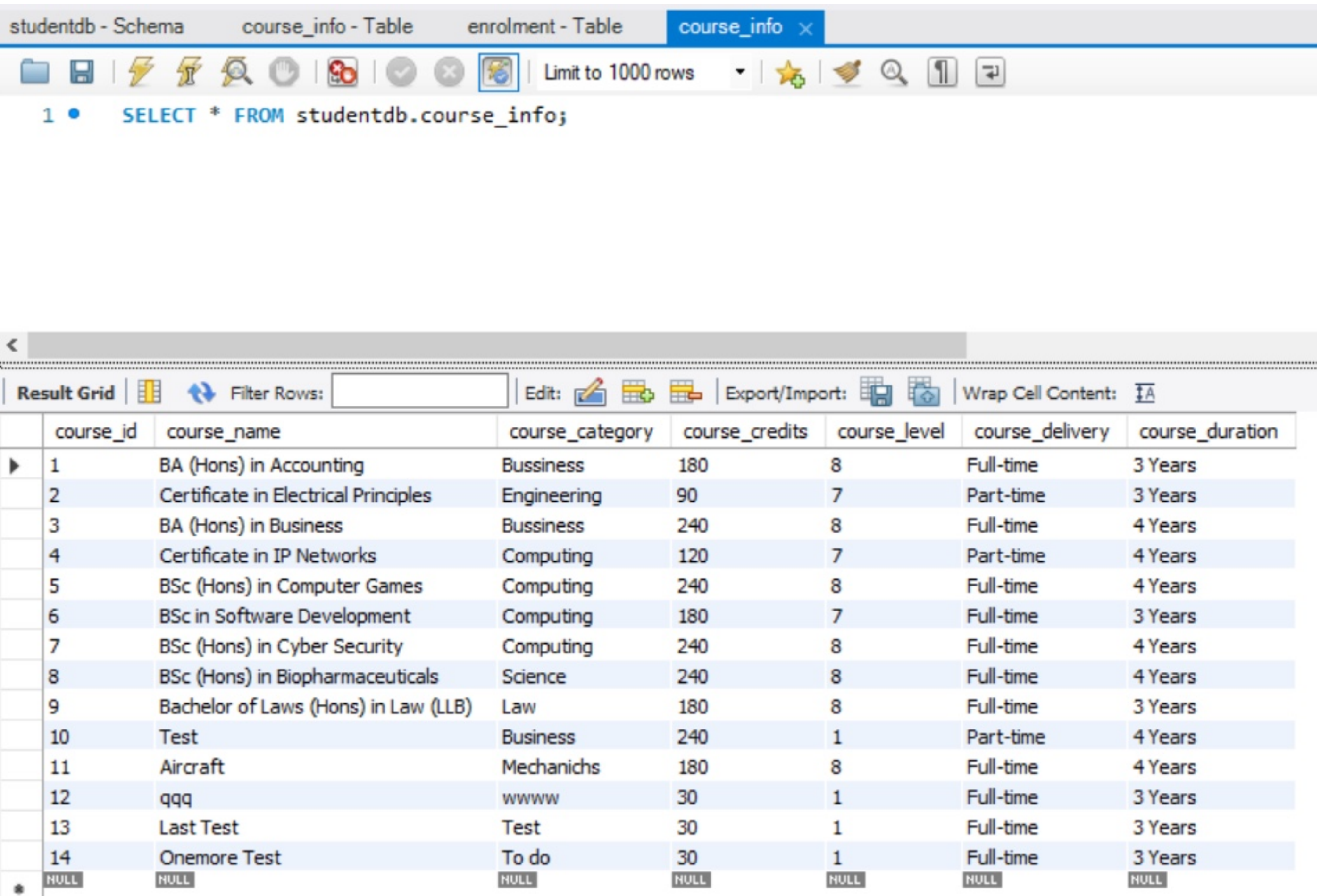
- CRUD: Create, Read, Update, Delete

- GDPR: General Data Protection Regulation

- FERPA: Family Educational Rights and Privacy Act

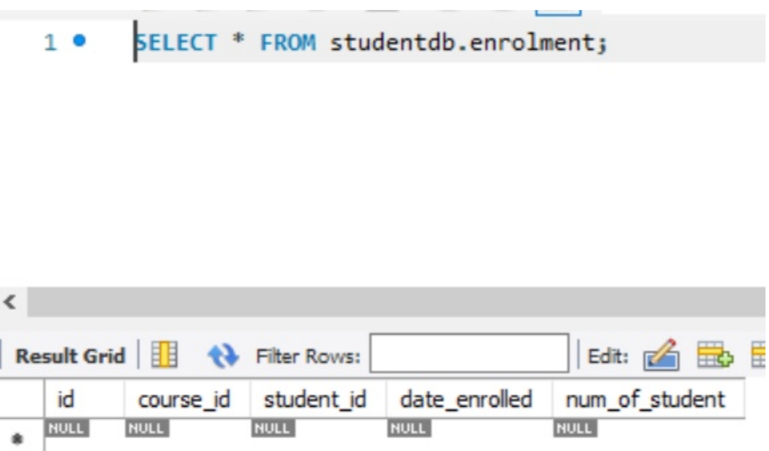
This System Requirements Document provides a comprehensive outline for the development and implementation of the Student Registration and Management System (SRMS) using Java and Swing GUI, guiding the design, testing, and deployment processes to ensure the successful realization of the system.

Database Tables
(Structure & Data)



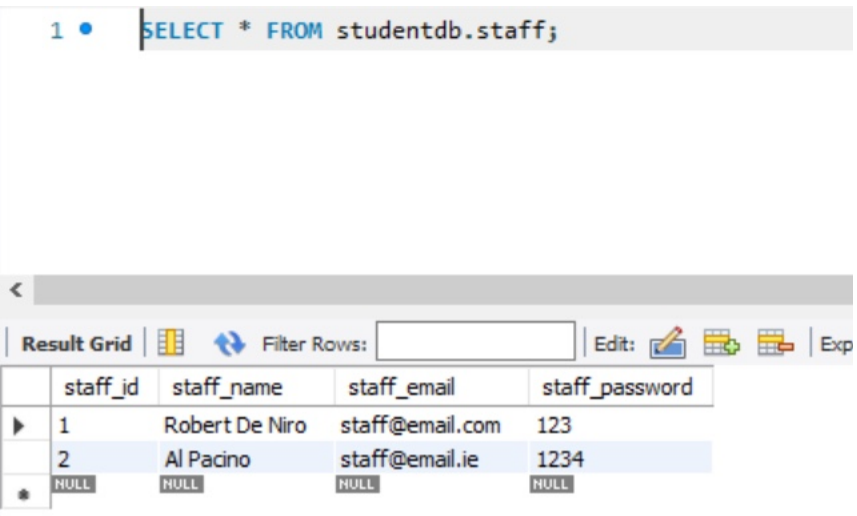
```
SELECT * FROM studentdb.course_info;
```

course_id	course_name	course_category	course_credits	course_level	course_delivery	course_duration
1	BA (Hons) in Accounting	Bussiness	180	8	Full-time	3 Years
2	Certificate in Electrical Principles	Engineering	90	7	Part-time	3 Years
3	BA (Hons) in Business	Bussiness	240	8	Full-time	4 Years
4	Certificate in IP Networks	Computing	120	7	Part-time	4 Years
5	BSc (Hons) in Computer Games	Computing	240	8	Full-time	4 Years
6	BSc in Software Development	Computing	180	7	Full-time	3 Years
7	BSc (Hons) in Cyber Security	Computing	240	8	Full-time	4 Years
8	BSc (Hons) in Biopharmaceuticals	Science	240	8	Full-time	4 Years
9	Bachelor of Laws (Hons) in Law (LLB)	Law	180	8	Full-time	3 Years
10	Test	Business	240	1	Part-time	4 Years
11	Aircraft	Mechanichs	180	8	Full-time	4 Years
12	qqq	wwww	30	1	Full-time	3 Years
13	Last Test	Test	30	1	Full-time	3 Years
14	Onemore Test	To do	30	1	Full-time	3 Years
NULL	NULL	NULL	NULL	NULL	NULL	NULL



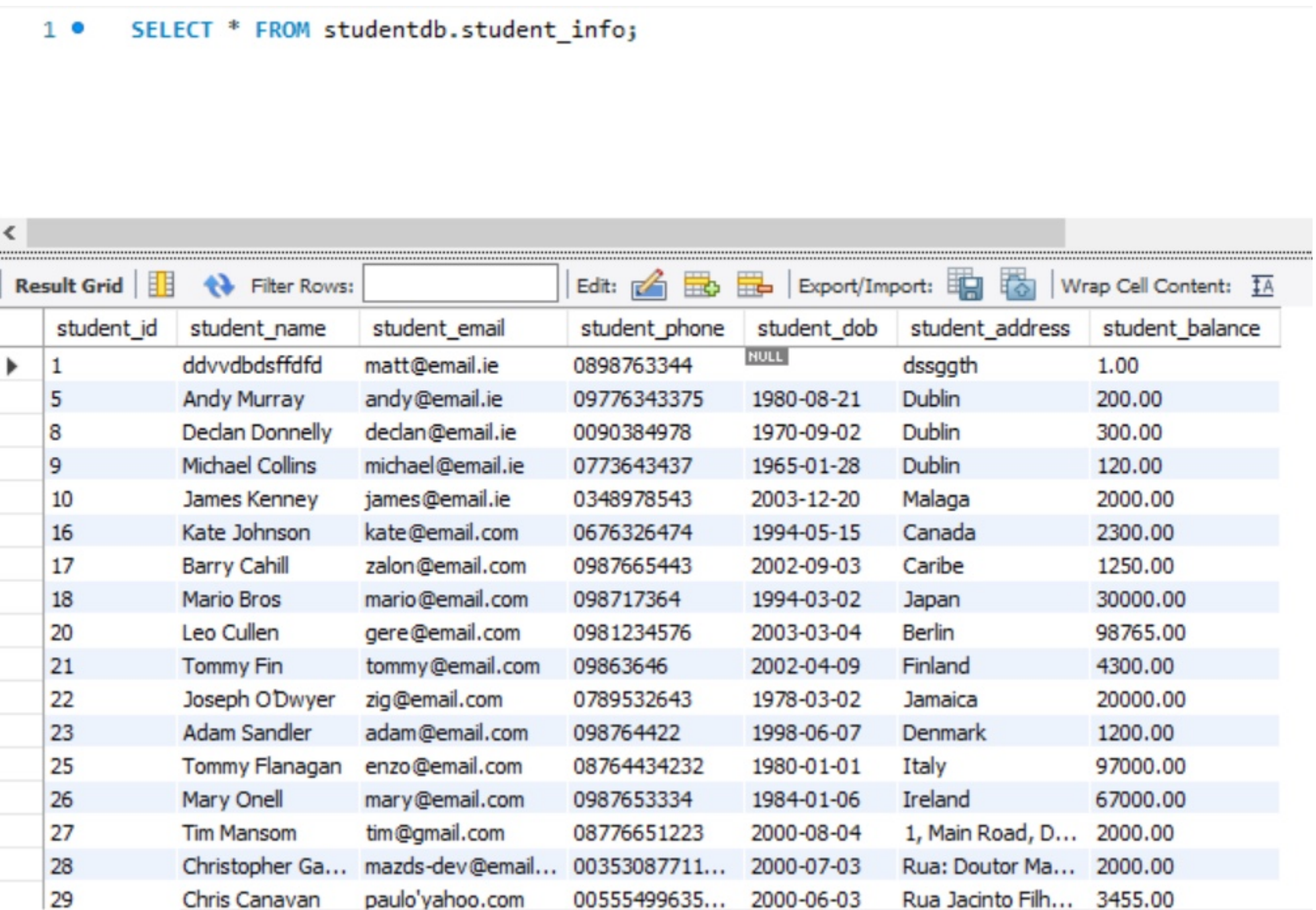
```
SELECT * FROM studentdb.enrolment;
```

id	course_id	student_id	date_enrolled	num_of_student
NULL	NULL	NULL	NULL	NULL



```
SELECT * FROM studentdb.staff;
```

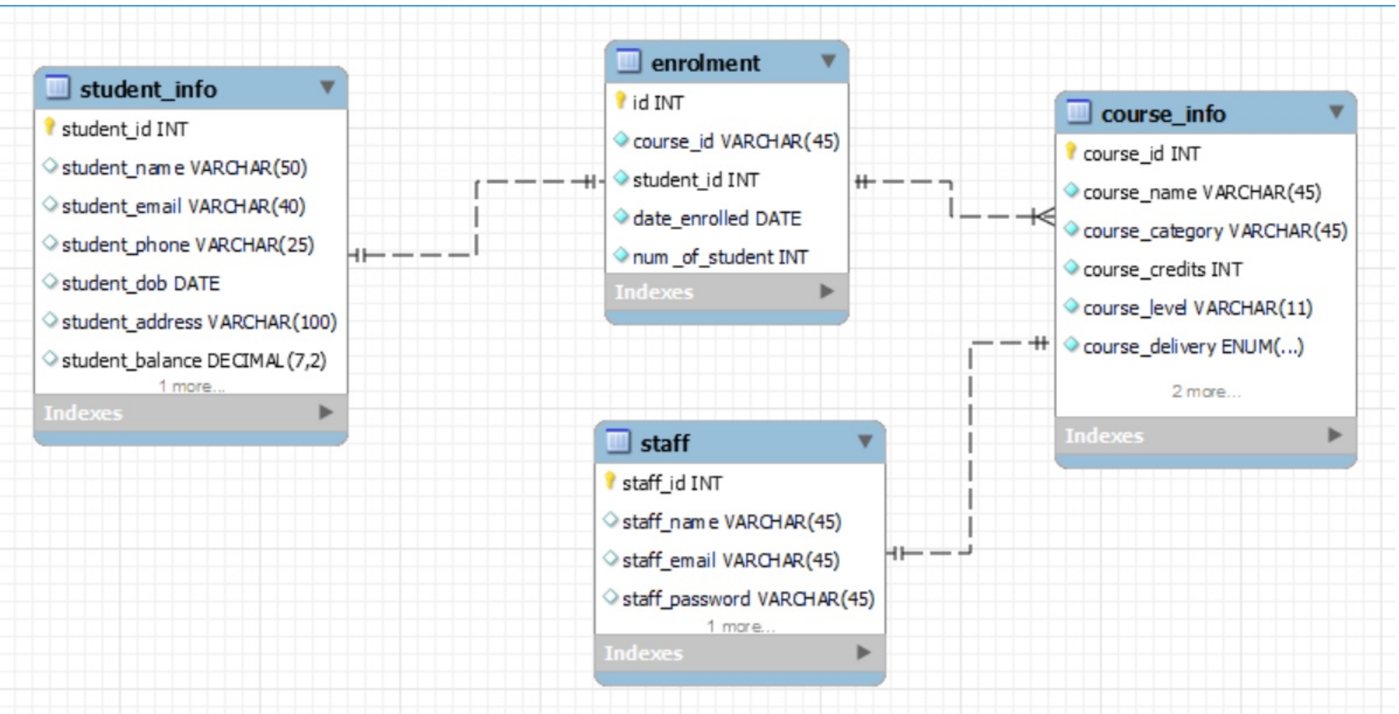
```
staff_id  staff_name       staff_email       staff_password
1         Robert De Niro   staff@email.com   123
2         Al Pacino        staff@email.ie    1234
NULL      NULL             NULL              NULL
```



```sql
SELECT * FROM studentdb.student_info;
```

student_id	student_name	student_email	student_phone	student_dob	student_address	student_balance
1	ddvvdbdsffdfd	matt@email.ie	0898763344	NULL	dssggth	1.00
5	Andy Murray	andy@email.ie	09776343375	1980-08-21	Dublin	200.00
8	Declan Donnelly	declan@email.ie	0090384978	1970-09-02	Dublin	300.00
9	Michael Collins	michael@email.ie	0773643437	1965-01-28	Dublin	120.00
10	James Kenney	james@email.ie	0348978543	2003-12-20	Malaga	2000.00
16	Kate Johnson	kate@email.com	0676326474	1994-05-15	Canada	2300.00
17	Barry Cahill	zalon@email.com	0987665443	2002-09-03	Caribe	1250.00
18	Mario Bros	mario@email.com	098717364	1994-03-02	Japan	30000.00
20	Leo Cullen	gere@email.com	0981234576	2003-03-04	Berlin	98765.00
21	Tommy Fin	tommy@email.com	09863646	2002-04-09	Finland	4300.00
22	Joseph O'Dwyer	zig@email.com	0789532643	1978-03-02	Jamaica	20000.00
23	Adam Sandler	adam@email.com	098764422	1998-06-07	Denmark	1200.00
25	Tommy Flanagan	enzo@email.com	08764434232	1980-01-01	Italy	97000.00
26	Mary Onell	mary@email.com	0987653334	1984-01-06	Ireland	67000.00
27	Tim Mansom	tim@gmail.com	08776651223	2000-08-04	1, Main Road, D...	2000.00
28	Christopher Ga...	mazds-dev@email...	00353087711...	2000-07-03	Rua: Doutor Ma...	2000.00
29	Chris Canavan	paulo'yahoo.com	00555499635...	2000-06-03	Rua Jacinto Filh...	3455.00

ER Diagram


student_info
student_id INT
student_name VARCHAR(50)
student_email VARCHAR(40)
student_phone VARCHAR(25)
student_dob DATE
student_address VARCHAR(100)
student_balance DECIMAL(7,2)
1 more...
Indexes
enrolment
id INT
course_id VARCHAR(45)
student_id INT
date_enrolled DATE
num_of_student INT
Indexes
course_info
course_id INT
course_name VARCHAR(45)
course_category VARCHAR(45)
course_credits INT
course_level VARCHAR(11)
course_delivery ENUM(...)
2 more...
Indexes
staff
staff_id INT
staff_name VARCHAR(45)
staff_email VARCHAR(45)
staff_password VARCHAR(45)
1 more...
Indexes

Interesting Source Code Snippets

Code snippets:

Interesting code snippets in the program include:

1. Method to get students by id

```
   public List<String> getAllStudentIds() throws SQLException {
       List<String> studentIds = new ArrayList<>(); // Create a list to store student IDs

       // Use a try-with-resources block to ensure resources are properly managed
       try (Connection connection = databaseUtil.getConnection();
            Statement statement = connection.createStatement();
            ResultSet resultSet =
statement.executeQuery(QueryUtil.selectAllStudentIDsQuery())) { // Query to fetch all
student IDs

           // Loop through the result set and add each student ID to the list
           while (resultSet.next()) {
               String studentId = resultSet.getString("student_id"); // Fetch the student ID from
the result set
               studentIds.add(studentId); // Add it to the list
           }
       }

       return studentIds; // Return the list of student IDs
   }
```

2. Method to get all student using arraylist

```
// Method to get all students from the database
   public List<Student> getAllStudent() throws SQLException {
       List<Student> students = new ArrayList<>(); // Create a list to store students

       // Use a try-with-resources block to ensure resources are properly managed
       try (Connection connection = databaseUtil.getConnection();
            Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery(QueryUtil.selectAllStudentQuery())) {

           // Loop through the result set and add students to the list
           while (resultSet.next()) {
               // Create a Student object from the current result set row
```

```
                Student student = new Student(
                        resultSet.getInt("student_Id"), // Student ID
                        resultSet.getString("student_name"), // Student's name
                        resultSet.getString("student_email"), // Student's email
                        resultSet.getString("student_phone"), // Student's phone
                        resultSet.getString("student_dob"), // Student's date of birthday
                        resultSet.getString("student_address"), // Student's address
                        resultSet.getDouble("student_balance") // Student's balance
                );
                students.add(student); // Add to the list
            }
        }

        return students; // Return the list of students
    } // End getAllStudent
```

3. Secure hash function

```
// Secure hashing function for passwords (SHA-256)
    private String hashPassword(String password) throws Exception {
        MessageDigest digest = MessageDigest.getInstance("SHA-256"); // Get the SHA-256 
hashing instance
        byte[] hash = digest.digest(password.getBytes("UTF-8")); // Hash the password
        StringBuilder hexString = new StringBuilder(); // Create a string to store the hash in hex

        // Convert the byte array to a hex string
        for (byte b : hash) {
            hexString.append(String.format("%02x", b)); // Format as two-digit hex
        }

        return hexString.toString(); // Return the hashed password as a hex string
    }
```

4. Method to validate credentials

```
public boolean validateStaffCredentials(String email, String password) throws 
SQLException, Exception {
        try (Connection connection = databaseUtil.getConnection();
             PreparedStatement stmt = connection.prepareStatement("SELECT * FROM staff 
WHERE staff_email = ? AND staff_password = ?")) {

            // Set the email and hashed password in the prepared statement
            stmt.setString(1, email);
            // Comment out the hashing line to use plain text passwords for testing
            // stmt.setString(2, hashPassword(password));
            stmt.setString(2, password); // Use plain text password for testing
            try (ResultSet rs = stmt.executeQuery()) {
                // Return true if there's at least one result (meaning valid credentials)
                return rs.next();
            }
        }
    } // End validateStaffCredentials- Users shall be automatically logged out after a specified
period of inactivity to ensure security using Swing GUI components for session management.
```

Tests

TEST CASES

TEST TITLE	PRIORITY	TEST CASE ID	TEST NUMBER	TEST DATE
LOGIN VALIDATION	1	SRMS001	1	20/04/24

TEST DESCRIPTION	TEST DESIGNED BY	TEST EXECUTED BY	EXECUTION DATE
Verify that user should be able to log in with valid email and password	Emmanuel. A.	Marvin. S.	21/04/24

TEST DESCRIPTION	TEST DEPENDENCIES	TEST CONDITIONS	TEST CONTROL
Security and authorization to access the system	The saved email and password from the database are required for login. Both email and password must be valid. Database connection with the application should be successful.	1. No entry at all 2. No email entry 3. No password entry 4. Invalid email entry 5. Invalid password entry 6. Invalid email and password 7. Valid email and password	Access should be denied except if a valid email and password are entered.

STEP ID	STEP DESCRIPTION	TEST DATE	EXPECTED RESULTS	ACTUAL RESULTS	PASS / FAIL	ADDITIONAL NOTES
1.1	Press login button	21/04/24	No access	No access	Pass	
1.2	Enter no email and a valid password	21/04/24	No access	No access	Pass	
1.3	Enter a valid email and no password	21/04/24	No access	No access	Pass	
1.4	Enter invalid email and valid password	21/04/24	No access	No access	Pass	
1.5	Enter a valid email and invalid password	21/04/24	No access	No access	Pass	
1.6	Enter an invalid email and invalid password	21/04/24	No access	No access	Pass	
1.7	Enter valid email and password	21/04/24	Access successful	Access successful	Pass	Visible password

TEST TITLE	PRIORITY	TEST CASE ID	TEST NUMBER	TEST DATE
ADD STUDENT DETAILS	2	SRMS002	2	21/04/24

TEST DESCRIPTION	TEST DESIGNED BY	TEST EXECUTED BY	EXECUTION DATE
Verify that the system successfully and accurately adds student details into the data base	Ihor. A.	Marvin. S.	22/04/24

TEST DESCRIPTION	TEST DEPENDENCIES	TEST CONDITIONS	TEST CONTROL
Insert student details to database which must take in the name, email, phone number, date of birth, address, and course.	The database server should be up and running. The application establishes connection to the database. The necessary tables and columns store student details.	Testing Valid inputs. Testing Invalid inputs. Testing no inputs Testing duplicate data. Testing multiple user data entry	Ensure there were no data in the database table and conditions for adding student are adequately tested.

STEP ID	STEP DESCRIPTION	TEST DATE	EXPECTED RESULTS	ACTUAL RESULTS	PASS / FAIL	ADDITIONAL NOTES
2.1	Add valid inputs	22/04/24	Successful addition of student details	Successful addition of student details	Pass	
2.2	Add invalid inputs	22/04/24	Request the entry of valid inputs	-	Failed	No validation done yet
2.3	Add no inputs	22/04/24	Request the entry of valid inputs	-	Failed	No validation done yet
2.4	Allow a user to enter duplicate record	22/04/24	Warn user of duplicate record	-	Failed	No validation done yet
2.5	Allow different user to enter same record	22/04/24	Warn user of duplicate record	-	Failed	No validation done yet

TEST TITLE	PRIORITY	TEST CASE ID	TEST NUMBER	TEST DATE
UPATE STUDENT DETAILS	3	SRMS003	3	21/04/24

TEST DESCRIPTION	TEST DESIGNED BY	TEST EXECUTED BY	EXECUTION DATE
Verify that the system successfully and accurately update student details in the data base	Marvin. S.	Ihor. A.	23/04/24

TEST DESCRIPTION	TEST DEPENDENCIES	TEST CONDITIONS	TEST CONTROL
Update student details in the database which may include the name, email, phone number, date of birth, address, and course.	The database server should be up and running. The application establishes connection to the database. The necessary tables and columns update student details.	Testing Valid inputs. Testing Invalid inputs. Testing duplicate data. Testing multiple user data entry Testing a deleted entry	Ensure there were changes made in the database table and conditions for updating student details were adequately tested.

STEP ID	STEP DESCRIPTION	TEST DATE	EXPECTED RESULTS	ACTUAL RESULTS	PASS / FAIL	ADDITIONAL NOTES
3.1	Update with valid inputs	22/04/24	Successful update of student details	-	Failed	No successful update yet
3.2	Update with invalid inputs	22/04/24	Request the entry of valid inputs	-	Failed	No validation done yet
3.3	Allow a user to update duplicate record	22/04/24	Warn user of duplicate record	-	Failed	No validation done yet
3.4	Allow different user to update same record	22/04/24	Warn user of duplicate record	-	Failed	No validation done yet
3.5	Update a deleted record	22/04/24	Warn user that record does not exist.	Successful update	Failed	No error handling done yet

TEST TITLE	PRIORITY	TEST CASE ID	TEST NUMBER	TEST DATE
REMOVE STUDENT DETAILS	4	SRMS004	4	23/04/24

TEST DESCRIPTION	TEST DESIGNED BY	TEST EXECUTED BY	EXECUTION DATE
Verify that the system successfully and accurately removes student details from the data base	Marvin. S.	Emmanuel. A.	23/04/24

TEST DESCRIPTION	TEST DEPENDENCIES	TEST CONDITIONS	TEST CONTROL
Delete student details from the database which must include all records such as name, email, phone number, date of birth, address, and course.	The database server should be up and running. The application establishes connection to the database. The necessary tables and columns remove student details.	Testing Valid inputs. Testing Invalid inputs. Testing deleted record.	Ensure there were records of student details in the database table and conditions for removing a student are adequately tested.

STEP ID	STEP DESCRIPTION	TEST DATE	EXPECTED RESULTS	ACTUAL RESULTS	PASS / FAIL	ADDITIONAL NOTES
4.1	Remove student details using valid inputs	24/04/24	Successful removal of student details	Successful removal of student details	Pass	
4.2	Remove student details using invalid inputs	24/04/24	Request the entry of valid inputs	-	Failed	No validation done yet
4.3	Remove an already deleted record	24/04/24	Warn user that record does not exist.	-	Failed	No validation done yet

TEST TITLE	PRIORITY	TEST CASE ID	TEST NUMBER	TEST DATE
ADD COURSE DETAILS	5	SRMS005	5	24/04/24

TEST DESCRIPTION	TEST DESIGNED BY	TEST EXECUTED BY	EXECUTION DATE
Verify that the system successfully and accurately adds course details into the data base	Ihor. A.	Emmanuel. A.	25/04/24

TEST DESCRIPTION	TEST DEPENDENCIES	TEST CONDITIONS	TEST CONTROL
Insert course details to database which must take in the name, category, credit, level, course delivery and duration.	The database server should be up and running. The application establishes connection to the database. The necessary tables and columns store course details.	Testing Valid inputs. Testing Invalid inputs. Testing no inputs. Testing duplicate data. Testing multiple user data entry	Ensure there were no data in the database table and conditions for adding courses are adequately tested.

STEP ID	STEP DESCRIPTION	TEST DATE	EXPECTED RESULTS	ACTUAL RESULTS	PASS / FAIL	ADDITIONAL NOTES
5.1	Add valid inputs	24/04/24	Successful addition of student details	Successful addition of student details	Passed	-
5.2	Add invalid inputs	24/04/24	Request the entry of valid inputs	Request the entry of valid inputs	Passed	-
5.3	Add no inputs	24/04/24	Request user to enter valid inputs	-	Failed	No validation done yet
5.4	Allow a user to enter duplicate record	24/04/24	Warn user of duplicate record	-	Failed	No validation done yet
5.5	Allow different user to enter same record	24/04/24	Warn user of duplicate record	-	Failed	No validation done yet

TEST TITLE	PRIORITY	TEST CASE ID	TEST NUMBER	TEST DATE
UPATE COURSE DETAILS	6	SRMS006	6	23/04/24

TEST DESCRIPTION	TEST DESIGNED BY	TEST EXECUTED BY	EXECUTION DATE
Verify that the system successfully and accurately update student details in the data base	Marvin. S.	Ihor. A.	24/04/24

TEST DESCRIPTION	TEST DEPENDENCIES	TEST CONDITIONS	TEST CONTROL
Update course details in the database which may include the name, category, credits, level, course delivery, and course duration.	The database server should be up and running. The application establishes connection to the database. The necessary tables and columns update course details.	Testing Valid inputs. Testing Invalid inputs. Testing duplicate data. Testing multiple user data entry Testing a deleted entry	Ensure there were changes made in the database table and conditions for updating course details were adequately tested.

STEP ID	STEP DESCRIPTION	TEST DATE	EXPECTED RESULTS	ACTUAL RESULTS	PASS / FAIL	ADDITIONAL NOTES
6.1	Update with valid inputs	24/04/24	Successful update of course details	Successful update of course details	Passed	-
6.2	Update with invalid inputs	24/04/24	Request the entry of valid inputs	-	Failed	No validation done yet
6.3	Allow a user to update duplicate record	24/04/24	Warn user of duplicate record	-	Failed	No validation done yet
6.4	Allow different user to update same record	24/04/24	Warn user of duplicate record	-	Failed	No validation done yet
6.5	Update a deleted record	24/04/24	Warn user that record does not exist.	Successful update	Failed	No error handling done yet

TEST TITLE	PRIORITY	TEST CASE ID	TEST NUMBER	TEST DATE
REMOVE COURSE DETAILS	7	SRMS004	7	25/04/24

TEST DESCRIPTION	TEST DESIGNED BY	TEST EXECUTED BY	EXECUTION DATE
Verify that the system successfully and accurately removes student details from the data base	Marvin. S.	Emmanuel. A.	25/04/24

TEST DESCRIPTION	TEST DEPENDENCIES	TEST CONDITIONS	TEST CONTROL
Delete student details from the database which must include all records such as name, category, credits, level, course delivery, and course duration.	The database server should be up and running. The application establishes connection to the database. The necessary tables and columns remove course details.	Testing Valid inputs. Testing Invalid inputs. Testing deleted record.	Ensure there were records of course details in the database table and conditions for removing a course are adequately tested.

STEP ID	STEP DESCRIPTION	TEST DATE	EXPECTED RESULTS	ACTUAL RESULTS	PASS / FAIL	ADDITIONAL NOTES
7.1	Remove course details using valid inputs	25/04/24	Successful removal of course details	-	Failed	Removal warning with no removal
7.2	Remove course details using invalid inputs	25/04/24	Request the entry of valid inputs	-	Failed	No validation done yet
7.3	Remove an already deleted record	25/04/24	Warn user that record does not exist.	-	Failed	No validation done yet

TEST TITLE	PRIORITY	TEST CASE ID	TEST NUMBER	TEST DATE
ENROLMENT DETAILS	8	SRMS008	8	29/04/24

TEST DESCRIPTION	TEST DESIGNED BY	TEST EXECUTED BY	EXECUTION DATE
Verify that the system successfully and accurately displays details from the data base	Marvin. S.	Ihor. A.	30/04/24

TEST DESCRIPTION	TEST DEPENDENCIES	TEST CONDITIONS	TEST CONTROL
Display course enrolment details from the database to include course information and number of students enrolled for course.	The database server should be up and running. The application establishes connection to the database. The necessary tables and columns display enrolment details.	Testing display of accurate data from student and course tables.	Ensure there are records of enrolment in the database table and conditions for displaying enrolment are adequately tested.

STEP ID	STEP DESCRIPTION	TEST DATE	EXPECTED RESULTS	ACTUAL RESULTS	PASS / FAIL	ADDITIONAL NOTES
4.1	Accurate display of enrolment details.	30/04/24	Successful display of enrolment details	-	Failed	-