# I. Introduction:

Welcome to our comprehensive school management system, designed to streamline and enhance the educational experience for all stakeholders involved in the learning process. Our project caters to the diverse needs of four distinct user roles: teachers, students, school administrators, and super administrators, all working together to create a more efficient and organized educational environment.

In this era of technology-driven education, our school management application is a one-stop solution, offering a range of features that empower users in their specific roles. Teachers can efficiently manage subjects, classes, and student marks, while students can easily access important information about their academic journey. School administrators have the tools to oversee the entire school's operations, ensuring a smooth and coordinated educational process.

The super admin, as the highest authority, plays a pivotal role in managing users, subjects, semesters, school organizations, and the crucial task of generating reports. These reports cover various aspects, including ethnicity, gender distribution, subjects being taught, and student promotions, offering valuable insights into the school's performance and demographics.

Our project aims to simplify and optimize the complex administrative processes within educational institutions, promoting a more effective and data-driven approach to school management. We believe that by providing this advanced platform, we can enhance the educational experience for all involved, ultimately contributing to the success and growth of both students and schools.

# II. Technology and sort preview:

## II.1. Technology

Welcome to our advanced technological ecosystem that powers a sophisticated application designed to meet the dynamic needs of users in an efficient and secure manner. Our comprehensive stack encompasses both front-end and back-end technologies, along with crucial libraries, ensuring a feature-rich and responsive system.

### \*\*Back-End Technologies:\*\*

1. \*\*Java Spring Boot 2.7.15:\*\* We leverage the latest version of Spring Boot to provide a solid foundation for our back-end, enabling us to build scalable and maintainable applications.

2. \*\*PostgreSQL and pgAdmin4:\*\* Our choice of PostgreSQL as the relational database management system, coupled with pgAdmin4 as the administration tool, ensures reliable data storage and efficient database management.

3. \*\*IntelliJ Ultimate 2023.1.1:\*\* Our development environment is powered by IntelliJ IDEA Ultimate, which offers a wide array of tools and features to enhance productivity, code quality, and collaboration.

4. \*\*EhCache:\*\* Caching is pivotal for optimizing application performance, and we employ EhCache to manage and store frequently accessed data efficiently, reducing response times and enhancing the user experience.

### \*\*Front-End Technologies:\*\*

1. \*\*Thymeleaf:\*\* Our front-end development relies on Thymeleaf, a modern server-side Java-based template engine that seamlessly integrates with Spring Boot, allowing us to create dynamic, data-driven web pages.

2. \*\*Bootstrap 5:\*\* We utilize Bootstrap 5, the latest version of this popular front-end framework, to ensure our application is responsive and visually appealing, offering a consistent and user-friendly design across various devices and screen sizes.

3. \*\*JavaScript and jQuery:\*\* JavaScript and jQuery power our interactive and client-side functionality, ensuring a rich and engaging user experience.

4. \*\*Ajax:\*\* Asynchronous JavaScript and XML (Ajax) is fundamental for building responsive and real-time features, enhancing the interactivity of our user interface.

### \*\*Libraries and Frameworks:\*\*

- \*\*EhCache:\*\* For efficient caching and performance optimization.

- \*\*Spring JPA:\*\* Simplifying database access and management using the Java Persistence API.

- \*\*Spring Security:\*\* Ensuring data security and user authentication.

- \*\*Spring Thymeleaf:\*\* Seamlessly integrating Thymeleaf with Spring for dynamic web content.

- \*\*Spring Web Application:\*\* Leveraging Spring's extensive capabilities for web application development.

- \*\*PostgreSQL:\*\* Utilized for robust data storage.

- \*\*Lombok:\*\* Streamlining Java code by reducing boilerplate.

- \*\*OAuth2-Client:\*\* Enabling secure and convenient login via Gmail.

- \*\*Apache POI:\*\* Empowering the application to export data to Excel with precision.

### \*\*Specifications:\*\*

Our application is designed with meticulous attention to detail, following a set of specific requirements:

- Paginated Lists: All lists within the application support pagination for easy navigation and performance optimization.

- Rapid Login: Users must complete the login process within 15 minutes, and automatic logout occurs after this time to ensure security.

- Password Encryption: User passwords are stored in the database with SHA256 encryption for enhanced security.

- Password Renewal: Users are required to change their passwords every 6 months, with each new password being distinct from the previous one.

- Image Handling: User profile images are storing as path in our database to help system export to excel file easier and save database store memory.

- Multiple Login Methods: Users can log in either with a password or through their Gmail accounts.

- Caching: Categorically, the application employs caching for list retrieval, i am using ehcache in this project because Ehcache is suitable for simple, in-memory caching needs in Java-based applications, especially when you want to improve the performance of database queries and reduce the load on your database server. It may not be the best choice for distributed or large-scale caching like redis.

- Security: Spring Security is implemented to ensure data protection and user access control.

- Excel Export: The application supports data export to Excel using Apache POI, facilitating efficient data analysis and reporting.

- Active Status: Only categories with an active status can be used, enhancing data integrity and application functionality.

Our technology stack and adherence to specifications ensure that our application delivers a secure, efficient, and feature-rich experience for users, addressing the challenges and demands of modern software development.

## II.2. Sort Preview

### II.2.1 Login Screen

A screenshot of a computer

Description automatically generated

This screen is for login both by username, password and google login.

### II.2.2 Student Role

#### *Dashboard*:

This is student dashboard that display student full name and navigation bar for 3 different activities:  
Class information:

A heart with bubbles on it

Description automatically generated

This screen showing class name and all student in current student class.

#### Change Class:

A screenshot of a computer

Description automatically generated

A white screen with black text

Description automatically generated

This is two screens for Change Class, if student hadn’t send any request in that semester that system will display menu for student select class change and either if student had been sent request for both accept and reject status.

#### Change *Class* History: A screenshot of a computer Description automatically generated

This is screen for Change Class History that view all history of that student change class.

#### Student *Transcript*: A screenshot of a computer Description automatically generated

This is screen for student Transcript that view all mark of selected semester and subject.

#### Student Information

A screenshot of a chat

Description automatically generated

This is screen for student update information.

### II.3. Teacher Role

#### Dashboard:

A screenshot of a computer

Description automatically generated

This is screen for teacher role dashboard.

#### Class Management:

A screenshot of a computer

Description automatically generated

This is screen for class management that accept teacher to update student information and change class for student in teacher responsibilities class.

#### Student update information:

A screenshot of a computer

Description automatically generated

This is screen for teacher update student information.

#### Teacher Change class for student in teacher responsibility class:

A white screen with black text

Description automatically generated

This is screen for teacher change class for student in teacher responsibility class similar to teacher role change class action.

#### Grade Management:

A screenshot of a computer

Description automatically generated

This is screen for teacher update/add student study result that accept teacher to choose class and subject they teach in each class, select teacher to update.

#### Grade *detail*: A screenshot of a computer Description automatically generated

This is screen for teacher add student study result and divide by each factor. If students transcript is not exist in database that system will automatically create new student transcript for teacher input mark action.

#### Teacher information update:

A screenshot of a computer

Description automatically generated

This is screen for teachers update information similar to student update information.

### II.4. School Admin Role

#### Dashboard:A white screen with text Description automatically generated

This is school admin dashboard that display school admin full name.

#### Class management:

A screenshot of a computer

Description automatically generated

This is screening that management all classes in school Organization. System provide update and delete, create more class action in this screen.

#### Update Class Information:

A screenshot of a computer

Description automatically generated

This is screen for update class information that allow schools admin to rename class name.

#### Delete Class Action:

A screenshot of a computer

Description automatically generated

This method will remove class from database but if there are any still in that class that will display error and cannot be delete.

#### Add Class Action:

A screenshot of a computer

Description automatically generated

This screen allows school admin to create more class in his/her school organization.

#### Subject Management: A screenshot of a computer Description automatically generated

This is subject management screen that display all subject in use of the school organization.

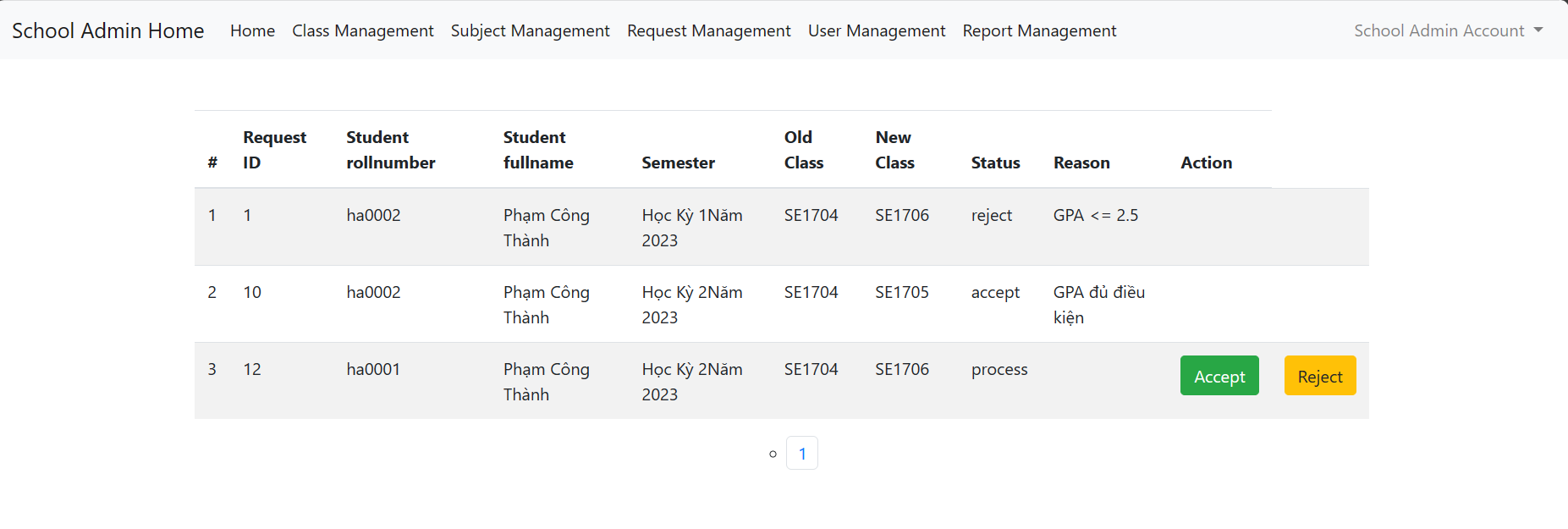
#### Create more subject action:

A screenshot of a computer

Description automatically generated

This screen allows school admin to create more subject in that school organization, the subject code can’t be duplicated.

#### Request Management:



This screen allows school admin to accept or reject student change class requests and when school admin accept or reject student change request, then that request action will not display.

#### Accept/*Reject* adding reason: A screenshot of a computer Description automatically generated

This is screen for school admin input reason of decide.

#### User Management:

A screenshot of a social media chat

Description automatically generated

This is screen for school admin manager both teacher and student in school.

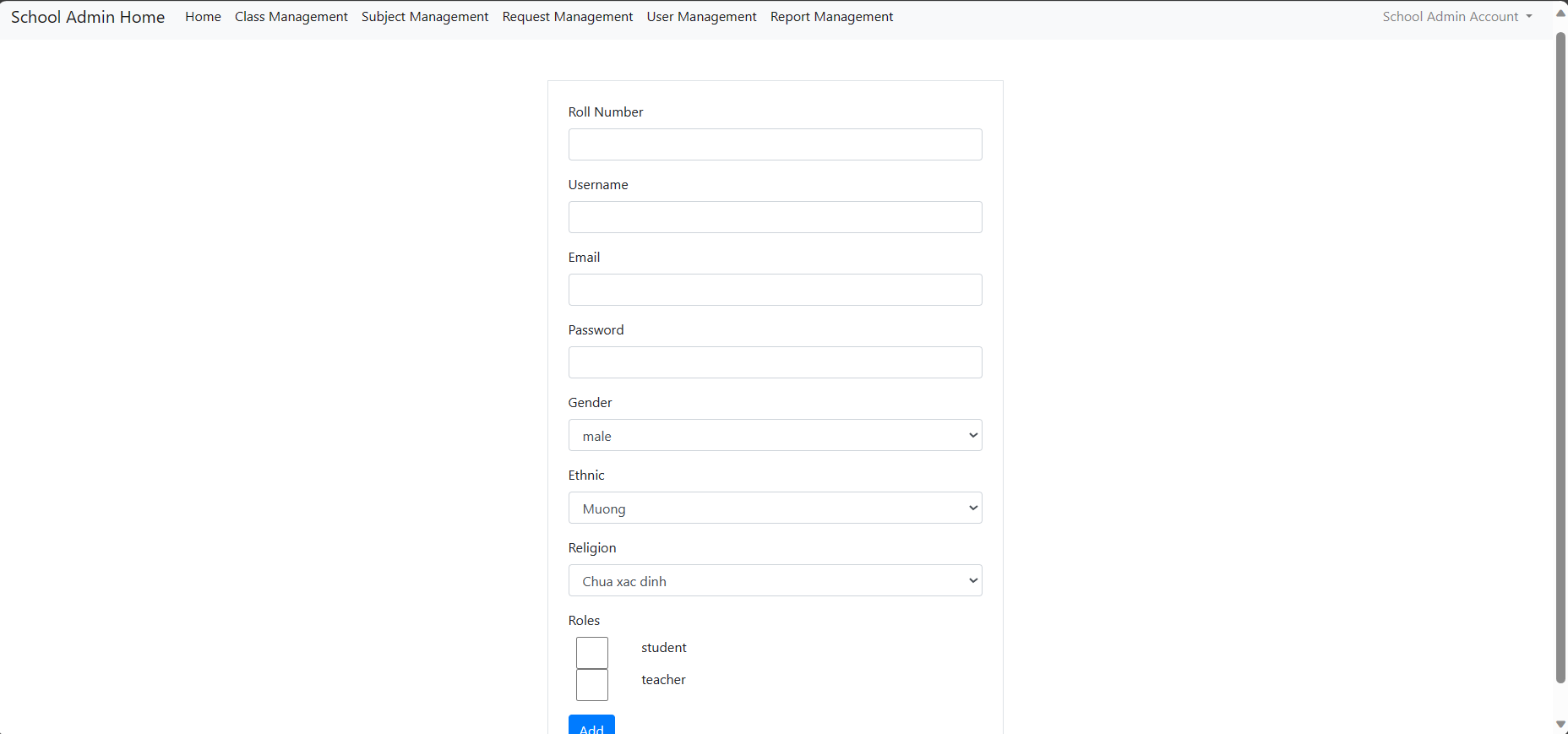
#### Update User information:

A screenshot of a computer

Description automatically generated

This is screen for school admin update user information, school admin can only set user role to student or teacher. And if user have student role, the Student class option will display and school admin can assign student to class, Either for teacher role and teacher class option. School Admin may update user twice, first time to update role and second time to change student/teacher class fit with the role.

#### Add More User:



This is screen for school admin to add more user, for both adding student and adding teacher. Rollnumber, username, and email can’t not be duplicate with other user in database. After added user, school admin need to update user for assign them to class.

#### Deactive User ActionA screenshot of a computer Description automatically generated

This is screen for deactive user that system will confirm school admin choice and if confirm is OK that system will change user status to deactive. Howover, school admin still can change status of user in update action.

#### Teacher Subject Management:

A screenshot of a computer

Description automatically generated

This is screen for Teacher Subject Management and only display for user have teacher role. This screen show all class and what subject selected teacher teach in each class.

#### Update Subject Teaching:

A screenshot of a computer

Description automatically generated

This is screen for update the class and subject for selected teacher. Teacher Class option will show all classes in that school organization and similar for teacher subject. After school admin Save, system will check if are there any other teacher teach selected subject in selected class and if its not that update will successfully.

#### Delete Teacher Subject Class:

A screenshot of a computer

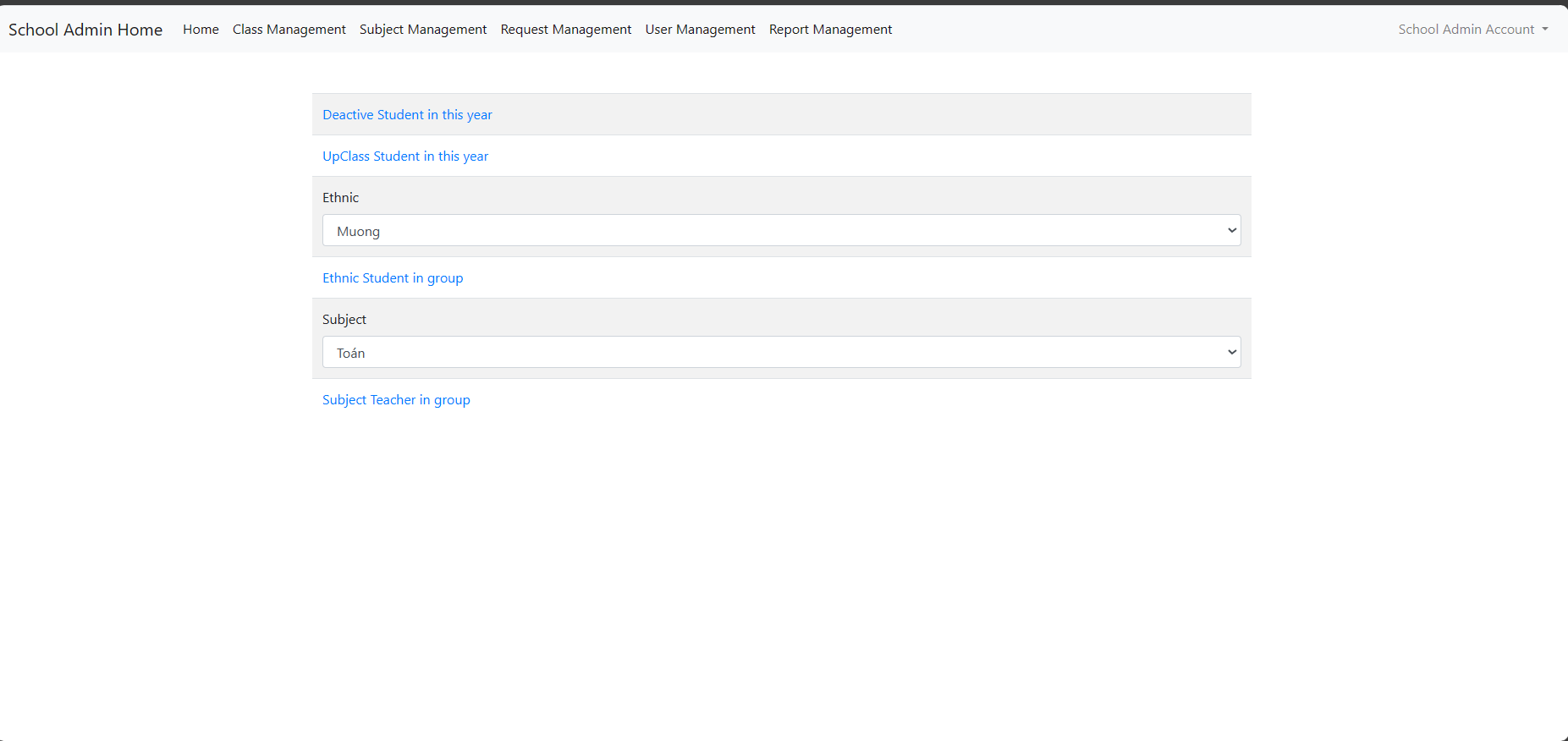
Description automatically generated

System will display confirm alert for school admin and if result is OK that system will change status of the record to deactive.

#### Assign to New Class: A screenshot of a computer Description automatically generated

This is screen for school admin assign selected teacher to new class and subject relationship. The teacher class option will show all class in current school organization and similar for teacher subject. After School admin adding, system will check if there are any other teacher teach that selected subject in selected class and adding if there aren’t.

#### Report Management:



This is screen display all report that school admin can export to Excel. Deactive Student is report for students that have status is deactive in this year in school. UpClass Student is report for students in school that have have mark average >=5. Ethnic Student in group is for user in school that have ethnic selected. Subject teacher in group is for teachers in school that teach selected subject.

# III. Models use in project:

## ChangeClass:

Use to handle change class management for student. Admin will know what semester and what class that students move to. Properties: id, student, oldClass, newClass, semester, status, and reason. The reason field is for school admin response that why accept or reject. When the Student sent request to change class, the default status of change class request is “process”. This model has 3 main statuses: process, accept and reject.

## City:

Use to save data of City. Properties: id, cityname, status.

## Class:

Use to manager class in school. A school will have many classes and each class just in one school -> relationship to school is ManyToOne. Properties: id, classname, classTeacher, classOrganization. classTeacher is mapping to Teacher\_Subject\_Class table to manager teacher teach subject and in what class. classOrganization is for school management, the system will know that class in what organization.

## District:

Use to save data of District. Properties: id, districtname, status and city that map to city table.

## Ethnic:

Use to save data of Ethnic. Properties: Id, ethnic, status.

## Gender:

Use to save data of Gender. Properties: Id, gender, status.

## Mark:

Use to save data for student mark. Properties: Id, mark, weight. There are 3 kinds of weigh such as 0.1 for coefficient 1, 0.2 for coefficient 2, 0.3 for coefficient 3.

## Organization:

Use to handle Organization Management. Properties: Id, schoolcode, schoolname, wardorganization, ward, status, operating day, classOrganization. This model is design to handle school and ward management. SchoolCode and SchoolName, wardorganization, status, operatingday are school information otherwise ward is storing ward information.

## Religion:

Use to save data of Religion. Properties: Id, religion, status.

## Role:

Use to save data of Role. Properties: Id, role name. there are four main roles: student, teacher, school admin and super admin.

## Semester:

Use to save data of Semester. Properties: Id, Semester, Year. This model is general model use for all system and management by super admin. When the Super Admin start new semester, database will create new semester and change status in table teacher\_class that close all the record in old semester.

## Student Transcript:

Use to store student transcript. Properties: id, student, semester, subject, marks. This model relates to user and subject, mark, semester model to management the study results of student in each subject and each different semester.

## Subject:

Use to handle information of subject. Properties: subjectcode, subjectname, organizations, subjectTeachers. This model relates to teacher\_class table to manager the subject and semester that teacher teach; organizations for locate what school organizations in use the subject.

## TeacherClassSubject

Use to manager teacher teach what subject and in what class in each semester. Properties: id, teacher, classTeaching , subjectTeaching, status. It does not have the semester field but still relate to semester that when super admin starts new semester, all record in this table will change status from active to inactive and in case that the old student continues teach that subject in that class next semester, school admin can update status of record in Update Function.

## Users

This is the most important model of School Management System that have many data to handle. Properties: rollnumber, fullname, username, password, email, gender, ethnic, religions, picture, hobbies, address, hometown, parrentName, status, roles, teacherclass, studentclass, lastchangepassword, deactivetime, teacherClassSubject, schoolOrganization. Rule: each user must have one rollnumber, username and email. The status must be active or deactive. Roles in this model is related to role model and one user can have more than one role. Picture is String type that locates the path to avatar picture locate in database. And because of user can have more than once role that teacherclass is for teacher role that manager what class that teacher responsibilities, while student class is for student role that manager what class that student in. lastchangepassword is Date type that save the Data last time user update, change password, this feature use to checking if user changed password last time after 6 months ago, user can’t be logging in and must change password. Deactivetime use to export report for school admin and super admin role, the report will include students in current year that be deactive.

## Ward

Use to stored data of ward. Properties: Id, wardname, status, district.

# IV. Feature

## Auth Controller:

### Definition:

This Controller use to handle login to system. Two-way login is by username, password, and google authentication.

### Using:

When user input username and password, the system will first be checking in database if the user with username is exist. If the user is not existed, the system will call back to login site with error logging fail, otherwise, if the user exists, the system will be using Password Encoder to check if password use provide match to encode password in db. If password is matching that system will check if that user change password in time or expired and redirect to select role page or reset password page.

## Excel Package:

### ExcelDownloadController

#### a.1: Definition

This Controller use to handle download excel report for each situation.

#### a.2: Using

##### a.2.1: method dowloadDeactiveStudentReport(response, session)

This method will get all student in school organizations that have status is inactive and the timedeactive of the user must be in current year. This student list then will be export to Excel and accept user to download directly from website by using response.

##### a.2.2: method downloadEthnicStudentReport(response, session)

This method will get all student in school organization that have ethnic selected provide by website. This student list then will be export to Excel and accept user to download directly from website by using response.

##### a.2.3: method downloadSubjectTeacherReport(response, session)

This method will get all Teacher in school organization that the teacher teaches the subject selected provide by website. This teacher list then will be export to Excel and accept user to download directly from website using response.

##### a.2.4: method downloadStudentUpClass(response, session)

This method will get all Student in school organization that the average mark of both semester is >=5. The system will calculate average mark automatically and export to Excel with average mark of each student then accept user to download directly from website using response.

##### a.2.5: method downloadSADeactiveReport(response)

This method is similar to method downloadDeactiveStudentReport except that the student list will be retrieve from all organizations not in one school organization.

##### a.2.6: method downloadSAUpClassReport(response)

This method is similar to method downloadStudentUpClass except that the student list will be retrieve from all organizations not in one school organization.

##### a.2.7: method downloadSASubjectTeacherReport(response)

This method is similar to method dowloadSubjectTeacherReport except that the teacher list will be retrieve from all organizations not in one school organization.

##### a.2.8: method dowloadSAEthnicStudentReport(response)

This method is similar to method downloadEthnicStudentReport except that the student list will be retrieve from all organizations not in one school organization.

## Location Package

### Location Controller

#### a.1. Definition:

This package handle exchange data with data base to get cities, districts and wards.

#### a.2. Using:

##### a.2.1: method getDistrictsbyCity(cityId):

This method will find the city by cityId in the database and return list of districts in found city into json.

##### a.2.2: method getWardsByDistrict(districtId):

This method will find the district by districtId in the database and return list of wards in found district into json.

##### a.2.3: method getOrganizationsByWard(wardId):

This method will find the ward by wardId in the database and return list of Organization in found ward into json.

## SchoolAdmin Package

### SchoolAdminClassManagementController

#### a.1. Definition:

This controller handle class management that provides all class in school organizations and accept user to update, create and delete class.

#### a.2. Using

##### a.2.1: method viewAllClass(session, model)

This method will validate user login session and view list of all class in the school organization.

##### a.2.2: method addClass(session, model)

This method will validate user login session and redirect user to information provide page to add class.

##### a.2.3: method addNewClass(classname, session, model)

This method will validate user login session and validate user input data. If the classname is correct format and not duplicated that this method will create new class into database.

##### a.2.4: method getUpdateClassInfor(classid, session, model)

This method will validate user login session and redirect user to page that provide detail information of selected class for update.

##### a.2.5: method updateClass(classid, classname, session, model)

This method will validate user login session and validate user input data. If classname is correct format and not duplicate so that the system will update new information for the selected class.

##### a.2.6: method deleteClass(classId, session, model)

This method will remove class from database.

### SchoolAdminHomeController

#### b.1. Definition

This controller handle validates user login session and display notification for user.

#### b.2. Using

##### b.2.1: method viewHomePage(session, model)

This method use to validate user login session and redirect user to homepage.

### SchoolAdminReportManagementController

#### c.1. Definition:

This controller handle report management.

#### c.2. Using:

##### c.2.1: method viewAllReportLink(session, model)

This method will validates user logging session and get list of subject and ethnic in school organizations for downloading report.

### SchoolAdminRequestManagementController

#### d.1. Definition:

This controller handle validated user login session and manager student change class request. School Admin can accept, reject request will reason response for student in that school.

#### d.2. Using

##### d.2.1: method viewAllRequest(session, model, page, pagesize)

This method uses to display all student change class request by pagination in that school organization.

##### d.2.2: method viewAcceptRequest(requestId, session, model)

This method uses to display information of request and provide environment for user to response reason.

##### d.2.3: method viewRejectRequest(requestId, session, model)

This method uses to display information of request and provide environment for user to response reason.

##### d.2.4: method acceptRequest(requestId, reason, session, model)

This method handle changes the request status to accept and save the reason data to database. The student who sent the request which has been accepted will be move to new class.

##### d.2.5: method rejectRequest(requestId, reason, session, model)

This method handle changes the request status to reject and save the reason data to database.

##### d.2.6: method isValidReason(reason)

This method will check the reason is in correct format

### SchoolAdminSubjectManagementController

#### e.1. Definition:

This controller handle Subject Management for School Admin. School Admin can only add more subject to current school organization because of one subject can have more than one organization can use, if school want to update or not use a subject that school admin has to ask super admin to update.

#### e.2.Using:

##### e.2.1: method viewAllSubject(session, model)

This method will get all subject of that school organization in database and view for user.

##### e.2.2: method viewFormAddSubject(session, model)

This method will provides environment for user to input data.

##### e.2.3: method addSubject(session, model, subjectcode, subjectname)

This method will get data user provide and validate data. If Subject code is not duplicate and in correct format, subject name is in correct format that the system will add new Subject to database.

##### e.2.4: method isValidSubjectCode(subjectcode)

This method will check if subject code is in correct format.

##### e.2.5: method isValidSubjectName(subjectname)

This method will check if subject name is in correct format.

##### e.2.6: method isDupplicateSubjectCode(subjectcode, subjects)

This method will scans in subjects list that return true if subjectcode is duplicated.

### SchoolAdminUserManagementController

#### f.1. Definition

This Controller handle user management for School Admin role. School Admin can update, delete, create new user in current school organization. Then school admin have permission to assign teacher to new class, view all teacher class and update relationships with teachers, class and subject they teach.

#### f.2.Using

##### f.2.1: method viewAllUsers(session, model, page, pagesize)

This method will get all Users of school organization in database and view for update, create and delete actions. The list of all Users will be pagination.

##### f.2.2: method deleteUser(rollNumber, session, model)

This method allows school admin to change status of user from active to inactive. That user account will not have granted authorization to login system.

##### f.2.3: method viewFormUpdate(rollNumber, session, model)

This method will get relate data from gender, ethnic, religions table for user update information, school admin just have permission to set role for student and teacher.

##### f.2.4: method updateUser(rollnumber, picture, fullname, newpassword, roles, status, gender, ethnic, religion, parrentname, address, hometown, hobbies, studentclass, teacherclass, email)

This method will validate data from user input and update new information of user that if user is student, studentclass option will display and accept school admin to change class of student. Either teacherclass option will only display if user is teacher and accept school admin to change class of teacher responsibility. Email and RollNumber is disable because of security and conflict with database.

##### f.2.5: method isFullNameValid(fullName)

This method checks if fullname is correct format or not.

##### f.2.6: method isPasswordValid(password)

This method checks if password is correct format or not.

##### f.2.7: method isEmailValid(email)

This method checks if email is correct format or not.

##### f.2.8: method isRollNumberValid(rollnumber)

This method checks if rollnumber is correct format or not.

##### f.2.9: method viewFormAddUser(session, model)

This method redirect user to input data page.

##### f.2.10: method addUser(rollnumber, username, password, email, gender, ethnic, religion, roles, session, model)

This method will validate user input data and if rollnumber, username, password, email is correct format and rollnumber, username, email is not duplicate that system will add new user to database of that school organization. If that user has only one student or teacher role that studentclass or teacher class will be null and school admin must update user information to assign teacher/student to class. Either, if that user has both role that studentclass and teacherclass will be null and school admin must update user information to assign that user to class.

##### f.2.11: method assignTeacher(rollnumber, session, model)

This method will get all Information of that teacher, what class he/she in and what subject he/she teach.

##### f.2.12: method deleteTeacherClassSubject(teacherClassId, session, model)

This method will change status of select TeacherClassSubject to deactive to explain that the teacher does not teach subject in that class anymore.

##### f.2.13: method viewFormAddTeacherClassSubject(teacherrollnumber, session, model)

This method will view all class and teacher that teacher can assign in, because of one subject must be teach in one class and if other teachers had been teaching that subject in the class so school admin can’t assign more teachers to that subject class.

##### f.2.14: method addTeacherClassSubject(teacherrollnumber, teacherclass, teachersubject, session, model)

This method will get data provide by user and validate. If in the select class have no teacher teach select subject that database will save new record.

##### f.2.15: method viewFormUpdateTeacherClassSubject(teacherSubjectClassId, session, model)

This method will get data of select teachersubjectclass and provide environment for user to input new data.

##### f.2.16: method updateTeacherClassSubject(recordId, teacherrollnumber, teacherclass, teachersubject, session, model)

This method will get and validate user input data and check if the select class have no teacher teach select subject that database will update current record.

## Student Package

### ChangeClassController

#### a.1.Definition:

This controller handle Change Class Request Management that provides environment for students to change class and view history.

#### a.2.Using:

##### a.2.1. method viewPage(session, model)

This method uses to view request of student change class, provide relate information about old and new class if student hadn’t send any request in there semester, the new class student want to move must have < 50 student. In one semester, students must send one request, both accept or reject. If student has sent one request that semester that the history button will display to accept change class history of that student.

##### a.2.2. method sendRequest(classname, session, model)

This method handles user input data, validate data and then create new record of classchange table in database. The default status of new request will be process and school admin can change this when decide to accept or reject request.

##### a.2.3. method viewHistory(session, model)

This method uses to get newest 10 requests of that student. Then, it returns to web and view all 10 newest requests of student.

### ClassInformation

#### b.1.Definition

This controller uses to handle ClassInformation viewing of student role. Student can view all other student information in his/her class.

#### b.2.Using

##### b.2.1: method viewInformation():

This controller uses to get all students list in current class that the student are in.

### StudentHomeController

#### c.1. Definition

This controller uses to display home page of student and notification.

#### c.2. Using

##### c.2.1: method viewHomePage(session, model)

This method validates user login session and user ‘s fullname information.

### StudentInformation

#### d.1.Definition

This controller handle student information management that student can view current information and update new information in the database.

#### d.2.Using

##### d.2.1: method getStudentInformation(session, model)

This method uses to get all information of student in database and view form to for student to update action.

##### d.2.2: method updateStudentprofile(picture, fullname, oldpassword, newpassword, gender, ethnic, religion, parrentname, address, hometown, hobbies, session, model)

This method gets all user input data and validate. The new data will update once and only when oldpassword match to password in database and all field are in correct format. Hobbies and picture can be null. RollNumber and Email are disable to secure system privacy and ensure that program not catch error.

### StudentTranscriptController

#### e.1.Definition:

This controller handle student transcript action. This controller will display all study result of student but not calculate that student can be up class or not. This controller just show the mark that students have in that times.

#### e.2.Using

##### e.2.1: method getAllSemester(session, model)

This method will find all Semester that student have mark and return to a list, then redirect user to choose semester.

##### e.2.2: method showStudentSubjectList(semesterId, model, session)

This method will find all Subject that student learn in Semester base on semesterId and return list of subject for student select.

##### e.2.2: method getStudentTranscript(semesterId, subjectId, model, session)

This method will find all marks and sort each mark to correct coefficients of subject base on subjectId and Semester base on semesterId. Average mark will be calculate by coefficients of each mark, mark with coefficient 1 will be multiplied by 0.1, coefficient 2 will be multiplied by 0.2 and coefficient 3 will be multiplied by 0.3.

## SuperAdmin Package

### SuperAdminHomeController

#### a.1.Definition:

This controller uses to display home page of student and notification.

#### a.2. Using

##### a.2.1: method viewHomePage(session, model)

This method validates user login session and super admin ‘s fullname information.

### SuperAdminOrganizationController

#### b.1.Definition:

This controller handle Organization management include add more school organizations and update school organizations information.

#### b.2. Using

##### b.2.1 method viewPage(session, model, city, district, ward, page, pageSize)

This method gets all cities, districts and ward from database and provide an filter engine that display all Organization in selected ward and display by pagination.

##### b.2.2 method viewFormUpdate(city, district, ward, organizationId, session, model)

This method gets all information about selected school organization and provide environment for user to input data.

##### b.2.3 method updateSchool(city, district, ward, schoolid, schoolname, status, operatingday, session, model)

This method will get all user input data, validate data and if all fields are correct that system will update new information of school to database. If the wardOrganization is not exist in database that system will automatically create new ward Organization and add School Organization to that wardOrganization.

##### b.2.4 method isValidSchoolName(schoolname)

This method returns true if schoolname is in correct format or false if it isn’t.

##### b.2.5 method viewFormAddSchool(wardid, session, model)

This method redirect user to page that can input data to add School to an wardOrganization.

##### b.2.6 method addOrganization(wardid, schoolcode, schoolname, session, model)

This method will get all user input data, validate and add new School Organization to database base on wardOrganization. If wardOrganization is not exist in Database that system will automatically create new wardOrganization base on wardid.

##### b.2.7 method isValidSchoolCode(schoolcode)

This method returns true if schoolcode is in correct format or false if it isn’t.

### SuperAdminReportController

#### c.1.Definition

This Controller handle display page for download Excel report.

#### c.2.Using

##### c.2.1 method viewPage(session, model)

This method validates user login session and get subjects and ethnics list for user choosing.

### SuperAdminSemesterController

#### d.1.Definition

This Controller handle Semester Management that accept super admin to start new semester of all system and display old semester in pagination.

#### d.2.Using

##### d.2.1 method viewPage(session, model)

This method gets all semester in database and display for super admin by pagination.

##### d.2.2 method addSemester(session, model)

This method accepts super admin start new semester and check that there are only 2 semesters in one year. If the new semester starts so that all teacher that teach subject in old semester will have status deactive.

## Teacher Package

### ClassManagementController

#### a.1.Definition:

This Controller handle Student in teacher responsibility class. List all 50 students in his/her class.

#### a.2. Using

##### a.2.1. method getAllStudent(model, session)

This method will gets all student in the teacher responsibility class and view all for teacher action.

### GradeManagementController

#### b.1.Definition:

This Controll accept teacher to evaluate student study result in each semester.

#### b.2.Using:

##### b.2.1 method getAllClass(session, model)

This method handle gets all Class that teacher teach and returns a list for teacher select.

##### b.2.2 method getAllSubjectInClass(classId, session, model)

This method uses to get all subject that teacher teach in select class and returns a list for teacher select.

##### b.2.3 method getAllStudentInClass(classid, subjectcode, session, model)

This method uses to get all Student in select Class for teacher to give grade of each student.

##### b.2.4 method updateStudentGrade(rollnumber, classid, subjectcode, session, model)

This method handles display current mark of chosen student. Each kind of mark will display in different coefficient sector.

##### b.2.5 method saveGradeForStudent(factor1Values, factor2Values, factor3Values, rollnumber, classId, subjectcode, session, model)

This method handles saving study mark of student. If the mark is exist already that teacher will accept to update grade for student or if the mark is null that system will create new student transcript record and allow teachers to input mark for student.

##### b.2.6 method isDouble(String dbString)

This method returns true if String input is a double and false if either.

### TeacherChangeClassController

#### c.1.Definition

This Controller handle Change Class for Student Management. It show the change class history of student and accept teacher to change class for that student if in that semester, the student hadn’t send any request yet.

#### c.2.Using

##### c.2.1 method viewPage(rollnumber, session, model)

This method uses to view menu for student change class. The new Class list include classes that have the amount of student less than 50 and not showing the current class of select students. It will check if in current semester that student had any request yet, if he/she have been sent that will display notifications and teacher just can see the change class history of that student.

##### c.2.2 method sendRequest(rollnumber, classname, session, model)

This method will create a new change class request in database and the default of request status is process.

##### c.2.3 method viewHistory(rollnumber, session, model)

This method will get all change class history of selected student and showing 10 first request.

### TeacherHomeController

#### d.1.Definition

This Controller handle showing dash board and notifications for teacher role.

#### d.2.Using

##### d.2.1 method viewHomePage(session, model)

This method validate user login session and get teacher ‘s fullname information.

### TeacherInformationController

#### e.1. Definition

This Controller handle update Information for teacher role.

#### e.2.Using

##### e.2.1 method getTeacherInformation(session, model)

This method gets all Information of teacher and redirect to page environment for update action.

##### e.2.2 method updateTeacherprofile(picture, fullname, oldpassword, newpassword, gender, ethnic, religion, address, hometown)

This method gets all teacher input data and validation. Oldpassword must match with encode password in database for update action.

##### e.2.3 method isFullNameValid(fullname)

This method returns true if fullname is in correct format or false either.

##### e.2.4 method isPasswordValid(password)

This method returns true if password is in correct format or false either.

##### e.2.5 method isAddressValid(address)

This method returns true if address is in correct format or false either.

### UpdateStudentInformationController

#### f.1.Definition

This Controller handle showing selected student information and provide update action

#### f.2.Using

##### f.2.1 method getStudentInformation(rollnumber, session, model)

This method gets all student information update action, it also gets gender, ethnic and religion list for chosen action.

##### f.2.2 method updateStudentprofile(picture, fullname, newpassword, rollnumber, gender, ethnic, religion, parrentname, address, hometown, hobbies, session, model)

This method gets user input data and validation. Then, the system will update student information if all fields are correct format.

##### f.2.3 method isFullNameValid(fullname)

This method returns true if fullname is in correct format or false either.

##### f.2.4 method isPasswordValid(password)

This method returns true if password is in correct format or false either.

##### f.2.5 method isAddressValid(address)

This method returns true if address is in correct format or false either.