ព្រះរាជាណាចក្រកម្ពុជា
ជាតិ សាសនា ព្រះមហាក្សត្រ

Software Engineering

Project : Student Internship Management System

Group: B2

Name of Student	ID of Student	Score
SAM NANGALEX	e20220253	……
PENG SEYHA	e20220620	……
OUN RITHI	e20220420	……
SOEUN SEREYVATH	e20220907	……
YI MONIROM	e20220175	……..

Lecturer: Roeun Pacharoth

Academic year 2025-2026

Progress Report 2

Name: SAM NANGALEX

Login System + Security

- ☑ ~~Add form validation~~
- ☑ ~~Add error message handling (wrong password)~~
- ☑ ~~create custom AuthenticationSuccessHandler~~

```java
@Controller
public class AuthConstroller {

    @GetMapping("/login")
    public String loginPage(Model model, String error) {
        if (error != null) {
            model.addAttribute(attributeName: "error", attributeValue: "Invalid email or password");
        }
        return "/login";
    }
```

```java
package project.demo.security;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.method.configuration.EnableMethodSecur
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.security.web.SecurityFilterChain;


@EnableMethodSecurity
@Configuration
public class SecurityConfig{

    @Bean
    public PasswordEncoder passwordEncoder(){
        return new BCryptPasswordEncoder();
    }
    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception{
        http
            .authorizeHttpRequests(auth ->
                auth.requestMatchers(...patterns: "/login").permitAll()
                .requestMatchers(...patterns: "/supervisor/").hasRole(role: "/SUPERVISOR")
                .anyRequest().authenticated()
            )
            .formLogin(form -> form
                .loginPage(loginPage: "/login")
                .loginProcessingUrl(loginProcessingUrl: "/login")
                .usernameParameter(usernameParameter: "email")
                .passwordParameter(passwordParameter: "password")
                .defaultSuccessUrl(defaultSuccessUrl: "/home", alwaysUse: true)
                .permitAll()
            );

        return http.build();
    }
}
```

```java
package project.demo.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Service;
import project.demo.entity.User;
import project.demo.repository.UserRepository;

@Service
public class CustomServiceUser implements UserDetailsService{
    @Autowired
    private UserRepository UserRepository;

    @Override
    public UserDetails loadUserByUsername(String email) throws UsernameNotFoundException{

        User user = UserRepository.findUsersByEmail(email).orElseThrow(()->new UsernameNotFoundException(msg: "User not found"));

        return org.springframework.security.core.userdetails.User
                .withUsername(user.getEmail())
                .password(user.getPassword())
                .roles(...roles: "STUDENT")
                .disabled(!user.isEnabled())
                .build();
    }

}
```

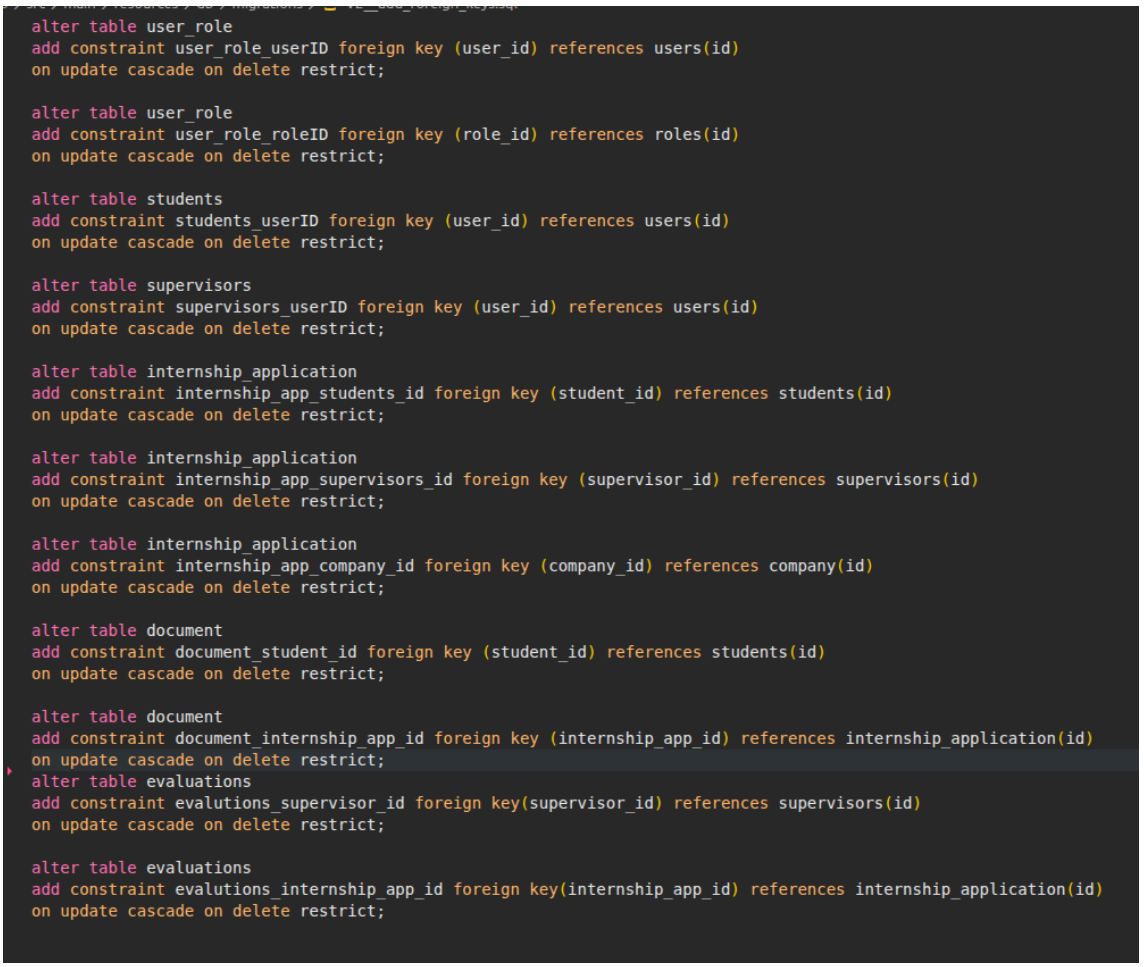
Database

- ☑ ~~role table~~
- ☑ ~~user_role table~~
- ☑ ~~Add foreign keys for relationships~~
- ☑ ~~Create internship_app table~~
- ☑ ~~Create evaluation table~~
- ☑ ~~Add foreign key student_id~~
- ☑ ~~Add foreign key supervisor_id~~
- ☑ ~~Add foreign key for evaluation~~
- ☑ ~~Create supervisor table~~
- ☑ ~~Add foreign key user_id → supervisor~~
- ☑ ~~Add constraints (unique user_id)~~

```sql
create table supervisors(
    id int auto_increment primary key,
    user_id int,
    full_name varchar(64) not null,
    email varchar(128) not null,
    phone_number int,
    department varchar(128)
);

create table internship_application(
    id int auto_increment primary key,
    student_id int,
    supervisor_id int,
    company_id int,
    title varchar(255) not null,
    description varchar(255),
    status enum('Pending', 'Approved', 'Rejected') default 'Pending' not null,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
);
```

```sql
CREATE TABLE document (
    id int PRIMARY KEY AUTO_INCREMENT,
    student_id int,
    internship_app_id int,
    file_name VARCHAR(255) NOT NULL,
    file_path VARCHAR(255) NOT NULL,
    uploaded_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

create table evaluations(
    id int primary key auto_increment,
    internship_app_id int,
    supervisor_id int,
    comment varchar(255),
    score int,
    evalutions_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

```
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+------------------------+
| Tables_in_internship   |
+------------------------+
| company                |
| document               |
| evaluations            |
| flyway_schema_history  |
| internship_application |
| roles                  |
| students               |
| supervisors            |
| user_role              |
| users                  |
+------------------------+
10 rows in set (0.00 sec)

mysql> ~
```



```
alter table user_role
add constraint user_role_userID foreign key (user_id) references users(id)
on update cascade on delete restrict;

alter table user_role
add constraint user_role_roleID foreign key (role_id) references roles(id)
on update cascade on delete restrict;

alter table students
add constraint students_userID foreign key (user_id) references users(id)
on update cascade on delete restrict;

alter table supervisors
add constraint supervisors_userID foreign key (user_id) references users(id)
on update cascade on delete restrict;

alter table internship_application
add constraint internship_app_students_id foreign key (student_id) references students(id)
on update cascade on delete restrict;

alter table internship_application
add constraint internship_app_supervisors_id foreign key (supervisor_id) references supervisors(id)
on update cascade on delete restrict;

alter table internship_application
add constraint internship_app_company_id foreign key (company_id) references company(id)
on update cascade on delete restrict;

alter table document
add constraint document_student_id foreign key (student_id) references students(id)
on update cascade on delete restrict;

alter table document
add constraint document_internship_app_id foreign key (internship_app_id) references internship_application(id)
on update cascade on delete restrict;
alter table evaluations
add constraint evalutions_supervisor_id foreign key(supervisor_id) references supervisors(id)
on update cascade on delete restrict;

alter table evaluations
add constraint evalutions_internship_app_id foreign key(internship_app_id) references internship_application(id)
on update cascade on delete restrict;
```

Backend

- ☑ ~~Create Student /entity/repository~~
- ☑ ~~Create User /entity//repository~~
- ☑ ~~Create Role /entity/repository~~
- ☑ ~~Create User_role /entity/repository~~

UserRepository.java UserRoleId.java RoleName.java

demo > src > main > java > project > demo > enums > RoleName.java > ...

```java
package project.demo.enums;

public enum RoleName {
    ROLE_STUDENT,
    ROLE_ADMIN,
    ROLE_SUPERVISOR
}
```

application.properties Role.java StudentRepository.java CustomServiceUser.java

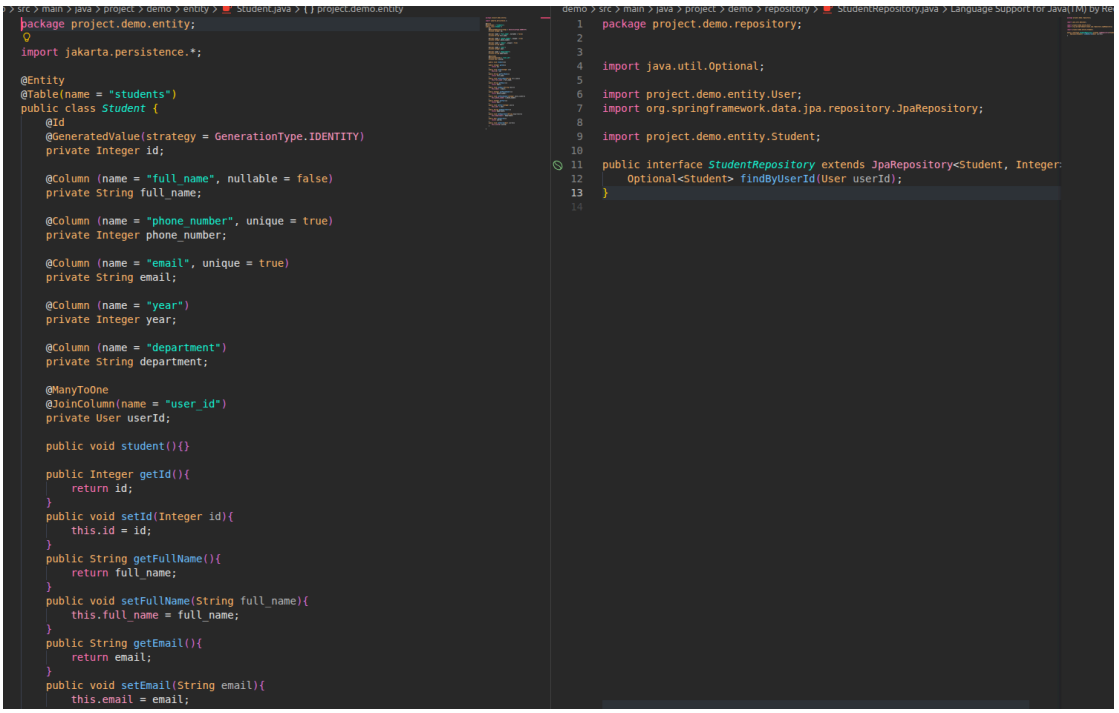
no > src > main > java > project > demo > entity > Role.java > Role > roleName

```java
package project.demo.entity;

import jakarta.persistence.*;
import project.demo.enums.RoleName;

@Entity
@Table(name = "roles")
public class Role {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @Enumerated(EnumType.STRING)
    @Column(name = "role_name", nullable = false, unique = true)
    private RoleName roleName;

    public Role() {}

    public Integer getId() {
        return id;
    }

    public RoleName getRoleName() {
        return roleName;
    }

    public void setRoleName(RoleName roleName) {
        this.roleName = roleName;
    }
}
```

RoleRepository.java

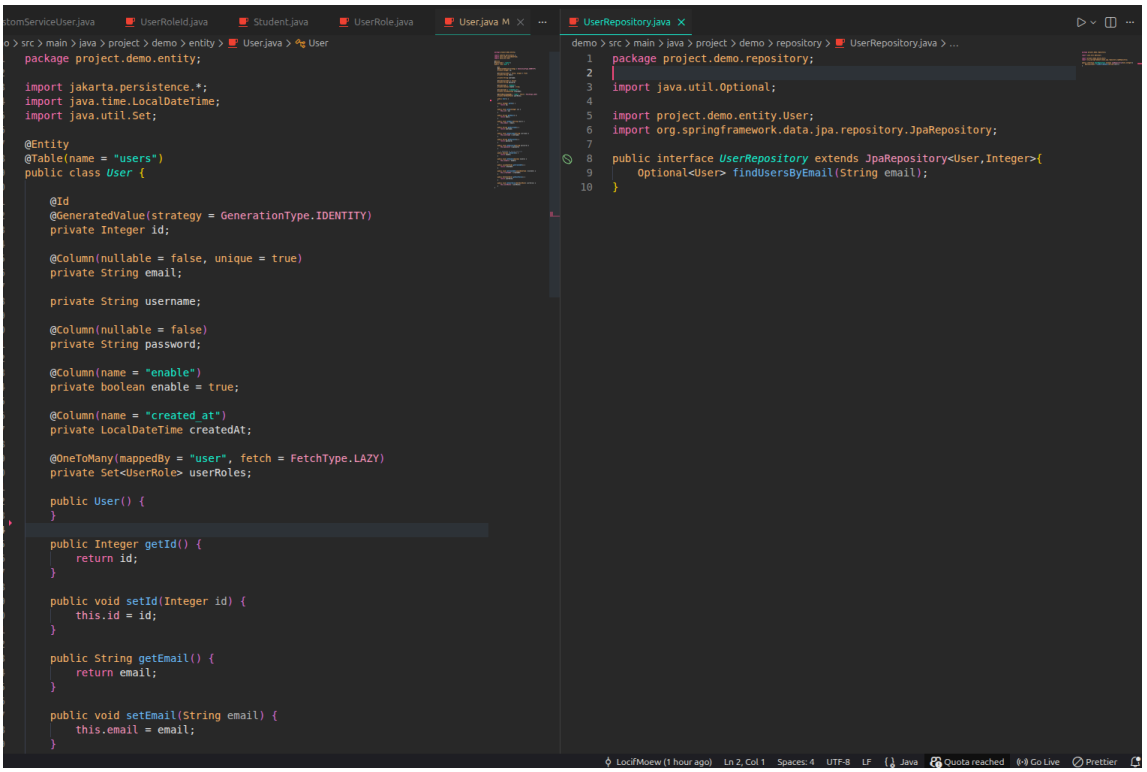
demo > src > main > java > project > demo > repository > RoleRepository.java > Language Support for Ja

```java
package project.demo.repository;

import project.demo.entity.Role;
import project.demo.enums.RoleName;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;

public interface RoleRepository extends JpaRepository<Role, Integer>{
    List<Role> findRoleByRoleName(RoleName roleName);
}
```



```java
package project.demo.entity;

import jakarta.persistence.*;

@Entity
@Table(name = "students")
public class Student {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @Column (name = "full_name", nullable = false)
    private String full_name;

    @Column (name = "phone_number", unique = true)
    private Integer phone_number;

    @Column (name = "email", unique = true)
    private String email;

    @Column (name = "year")
    private Integer year;

    @Column (name = "department")
    private String department;

    @ManyToOne
    @JoinColumn(name = "user_id")
    private User userId;

    public void student(){}

    public Integer getId(){
        return id;
    }
    public void setId(Integer id){
        this.id = id;
    }
    public String getFullName(){
        return full_name;
    }
    public void setFullName(String full_name){
        this.full_name = full_name;
    }
    public String getEmail(){
        return email;
    }
    public void setEmail(String email){
        this.email = email;
```

```java
package project.demo.repository;


import java.util.Optional;

import project.demo.entity.User;
import org.springframework.data.jpa.repository.JpaRepository;

import project.demo.entity.Student;

public interface StudentRepository extends JpaRepository<Student, Integer>
    Optional<Student> findByUserId(User userId);
}
```



```java
package project.demo.entity;

import jakarta.persistence.*;
import java.time.LocalDateTime;
import java.util.Set;

@Entity
@Table(name = "users")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @Column(nullable = false, unique = true)
    private String email;

    private String username;

    @Column(nullable = false)
    private String password;

    @Column(name = "enable")
    private boolean enable = true;

    @Column(name = "created_at")
    private LocalDateTime createdAt;

    @OneToMany(mappedBy = "user", fetch = FetchType.LAZY)
    private Set<UserRole> userRoles;

    public User() {
    }

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }
```

```java
package project.demo.repository;

import java.util.Optional;

import project.demo.entity.User;
import org.springframework.data.jpa.repository.JpaRepository;

public interface UserRepository extends JpaRepository<User,Integer>{
    Optional<User> findUsersByEmail(String email);
}
```

Name: OUN RHITI

Backend

- ☑ ~~Implement create company entity and repository~~
- ☑ ~~Implement view all companies~~
- ☑ ~~Implement update company~~
- ☑ ~~Implement delete company~~
- ☑ ~~Test using Postman~~

```
> src > main > java > project > demo > controller > CompanyController.java > ...
import java.util.List;

import org.springframework.web.bind.annotation.*;

import project.demo.entity.Company;
import project.demo.repository.CompanyRepository;

@RestController
@RequestMapping("/api/companies")
public class CompanyController {

    private final CompanyRepository companyRepository;

    public CompanyController(CompanyRepository companyRepository) {
        this.companyRepository = companyRepository;
    }

    @PostMapping
    public Company createCompany(@RequestBody Company company) {
        return companyRepository.save(company);
    }

    @GetMapping
    public List<Company> getAllCompanies() {
        return companyRepository.findAll();
    }

    @PutMapping("/{id}")
    public Company updateCompany(@PathVariable int id,
                                 @RequestBody Company company) {
        company.setId(id);
        return companyRepository.save(company);
    }

    // DELETE company
    @DeleteMapping("/{id}")
    public void deleteCompany(@PathVariable Integer id) {
        companyRepository.deleteById(id);
    }
}
```

src > main > java > project > demo > entity > Company.java > Company

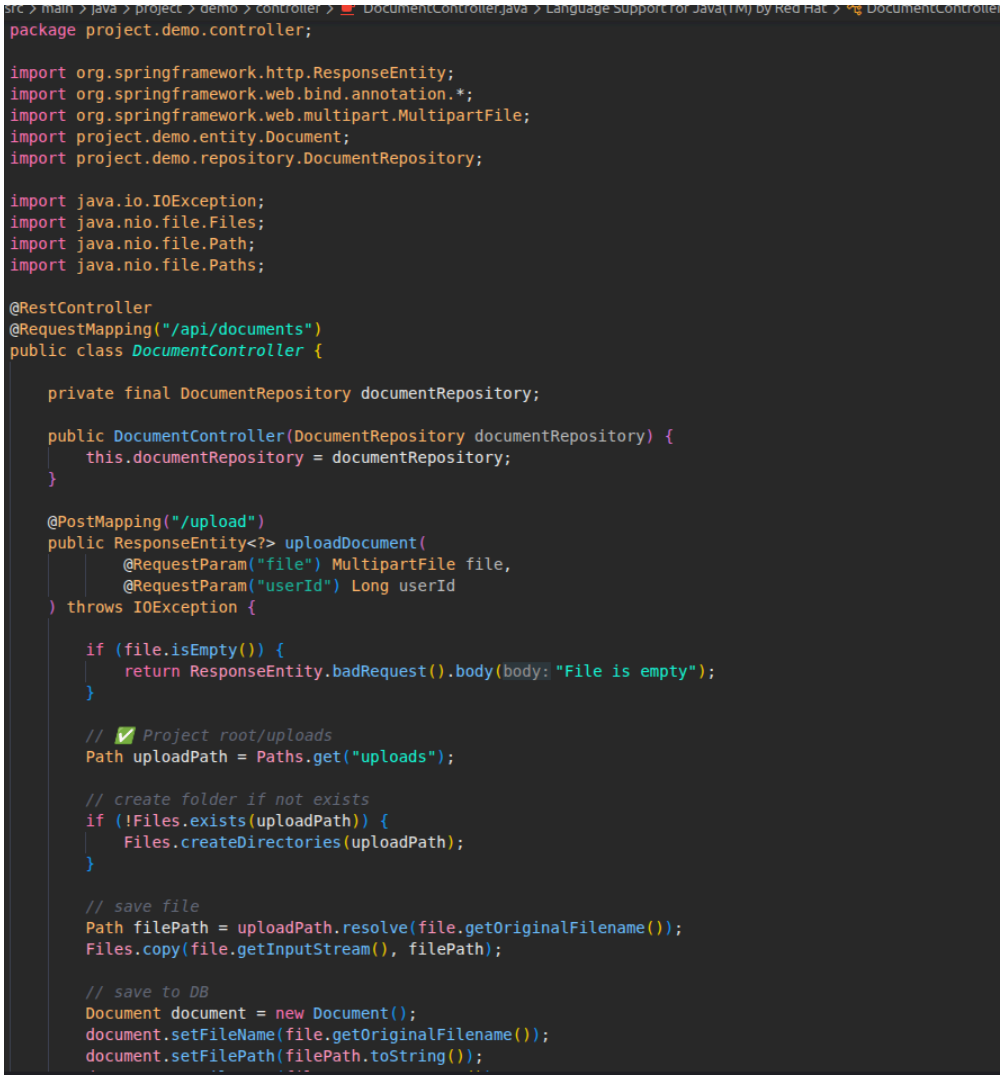
```java
package project.demo.entity;

import jakarta.persistence.*;

@Entity
@Table(name = "company")
public class Company {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @Column(name = "name", nullable = false)
    private String name;

    @Column(name = "address")
    private String address;

    @Column(name = "phone_number")
    private Integer phoneNumber;

    // Constructors
    public Company() {}

    // Getters and Setters
    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getAddress() {
        return address;
    }

    public void setAddress(String address) {
        this.address = address;
    }
```

src > main > java > project > demo > repository > CompanyRepository.java > ...

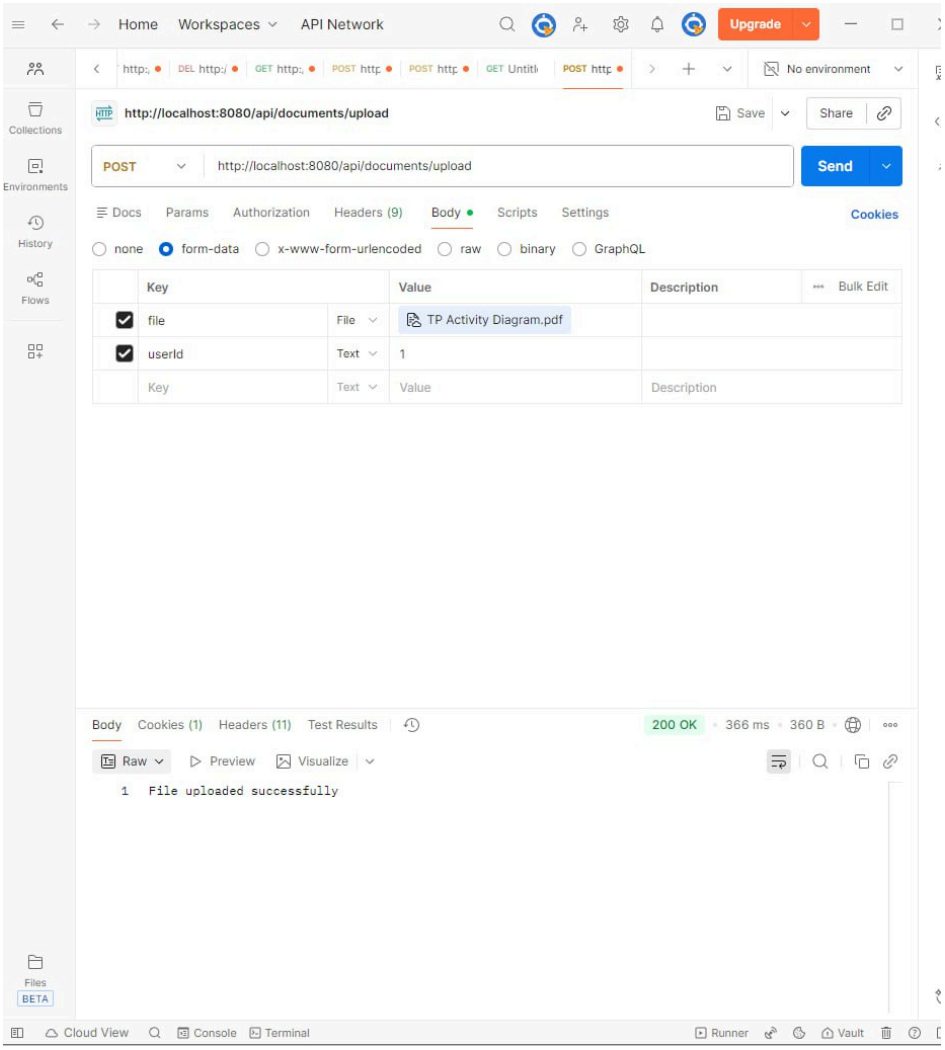
```java
package project.demo.repository;

import org.springframework.data.jpa.repository.JpaRepository;
import project.demo.entity.Company;

public interface CompanyRepository extends JpaRepository<Company, Integer>
}
```

- ☑ ~~Create document model~~
- ☑ ~~Create document enitity / repository~~
- ☑ ~~Create upload controller~~
- ☑ ~~Add file validation (size, type: pdf, docx, etc.)~~
- ☑ ~~Add file storage logic~~

- [x] ~~Save file path to DB~~
- [x] ~~Link document → user or internship~~

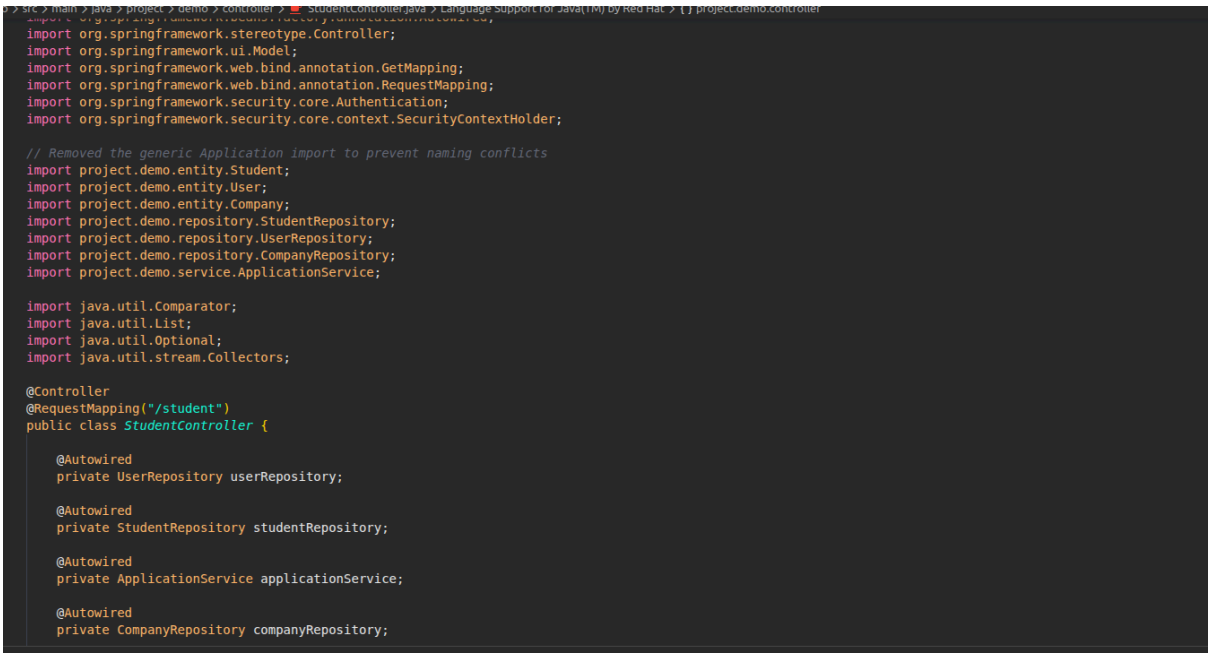


```java
package project.demo.controller;

import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.multipart.MultipartFile;
import project.demo.entity.Document;
import project.demo.repository.DocumentRepository;

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;

@RestController
@RequestMapping("/api/documents")
public class DocumentController {

    private final DocumentRepository documentRepository;

    public DocumentController(DocumentRepository documentRepository) {
        this.documentRepository = documentRepository;
    }

    @PostMapping("/upload")
    public ResponseEntity<?> uploadDocument(
            @RequestParam("file") MultipartFile file,
            @RequestParam("userId") Long userId
    ) throws IOException {

        if (file.isEmpty()) {
            return ResponseEntity.badRequest().body(body: "File is empty");
        }

        // ✅ Project root/uploads
        Path uploadPath = Paths.get("uploads");

        // create folder if not exists
        if (!Files.exists(uploadPath)) {
            Files.createDirectories(uploadPath);
        }

        // save file
        Path filePath = uploadPath.resolve(file.getOriginalFilename());
        Files.copy(file.getInputStream(), filePath);

        // save to DB
        Document document = new Document();
        document.setFileName(file.getOriginalFilename());
        document.setFilePath(filePath.toString());
```


Home
Workspaces
API Network
Upgrade
No environment
Collections
Environments
History
Flows
Files
BETA
http://localhost:8080/api/documents/upload
Save
Share
POST
http://localhost:8080/api/documents/upload
Send
Docs
Params
Authorization
Headers (9)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
Key
Value
Description
Bulk Edit
file
File
TP Activity Diagram.pdf
userId
Text
1
Key
Text
Value
Description
Body
Cookies (1)
Headers (11)
Test Results
200 OK
366 ms
360 B
Raw
Preview
Visualize
1 File uploaded successfully
Cloud View
Console
Terminal
Runner
Vault

Name: PENG SEYHA

- [x] ~~Create Student controller~~



```java
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;

// Removed the generic Application import to prevent naming conflicts
import project.demo.entity.Student;
import project.demo.entity.User;
import project.demo.entity.Company;
import project.demo.repository.StudentRepository;
import project.demo.repository.UserRepository;
import project.demo.repository.CompanyRepository;
import project.demo.service.ApplicationService;

import java.util.Comparator;
import java.util.List;
import java.util.Optional;
import java.util.stream.Collectors;

@Controller
@RequestMapping("/student")
public class StudentController {

    @Autowired
    private UserRepository userRepository;

    @Autowired
    private StudentRepository studentRepository;

    @Autowired
    private ApplicationService applicationService;

    @Autowired
    private CompanyRepository companyRepository;
```

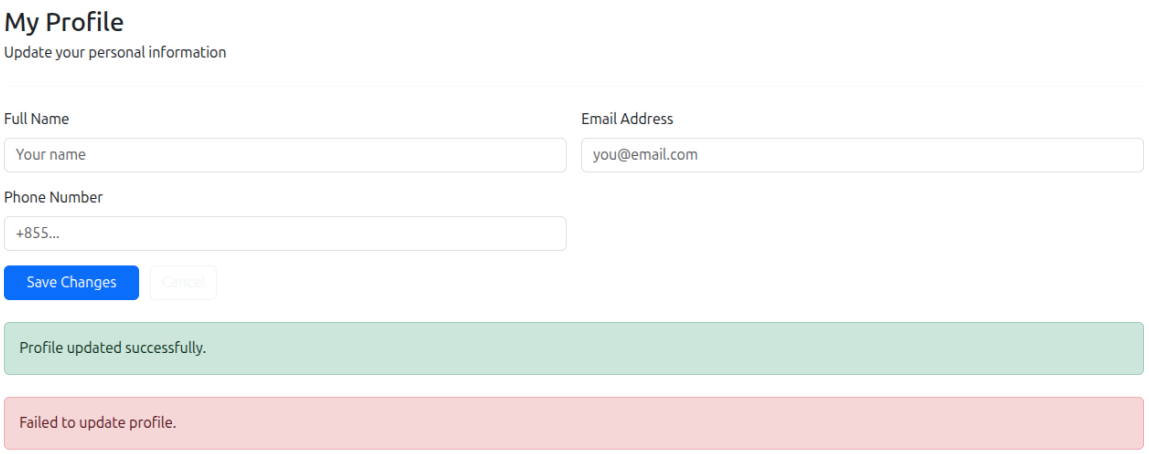
Frontend

Student Dashboard

- ☑ ~~Dashboard UI layout~~
- ☑ ~~Show basic information about student~~
- ☑ ~~Add menu links (Profile, Application, Evaluation, Status)~~

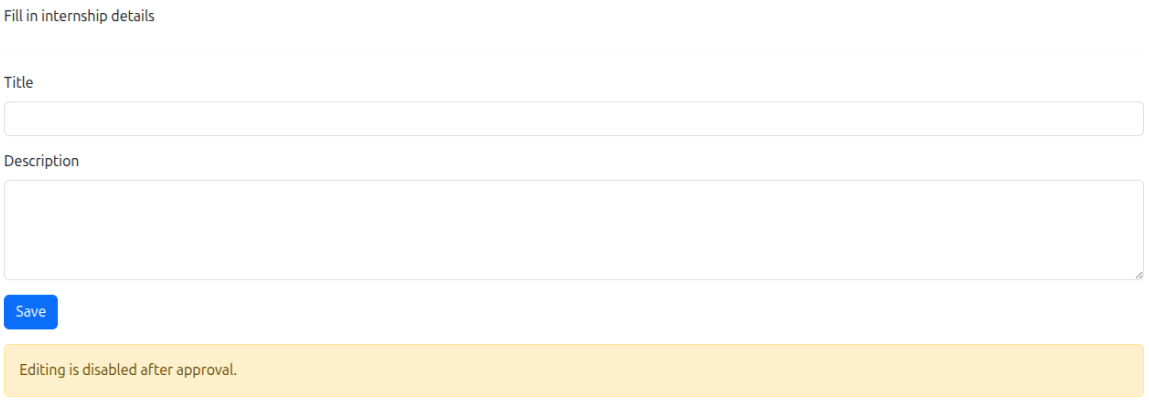
Student Forms

- ☑ ~~Student registration or edit profile form~~
- ☑ ~~Validation (email, phone, required fields)~~
- ☑ ~~Add success/error messages~~



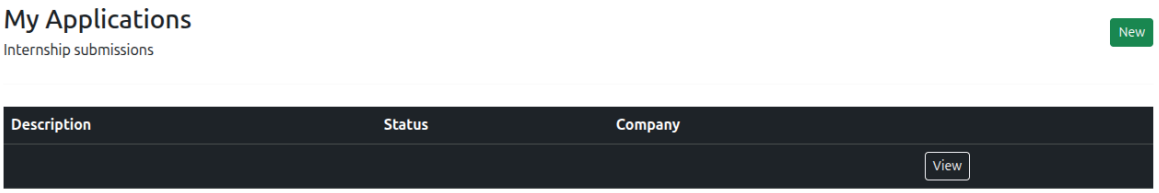
Application Form

- [x] ~~Create form UI~~
- [x] ~~Validation for required fields~~
- [x] ~~Integrate API to submit~~



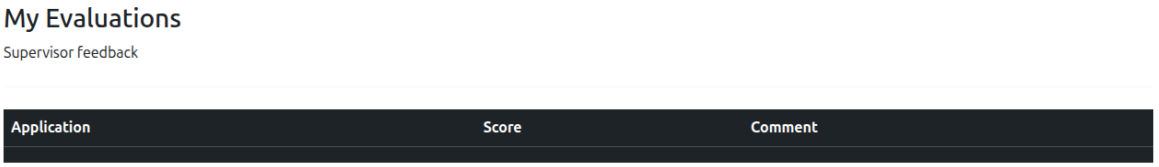
Application Detail Page

- [x] ~~Display all application info~~
- [x] ~~Show status~~
- [x] ~~Show company assigned~~



Evaluation List Page

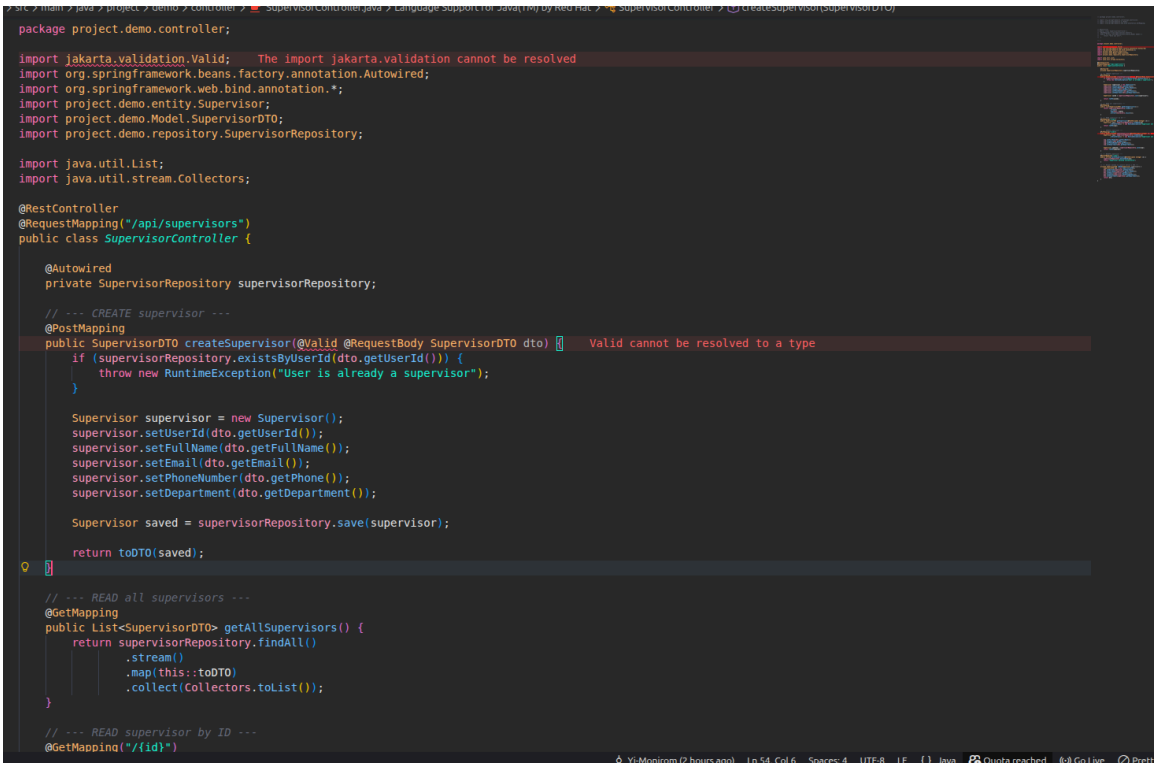
- [x] ~~Table listing all evaluations~~
- [x] ~~Filter by student/supervisor~~
- [x] ~~View evaluation detail~~
- [x] ~~Edit evaluation (supervisor only)~~



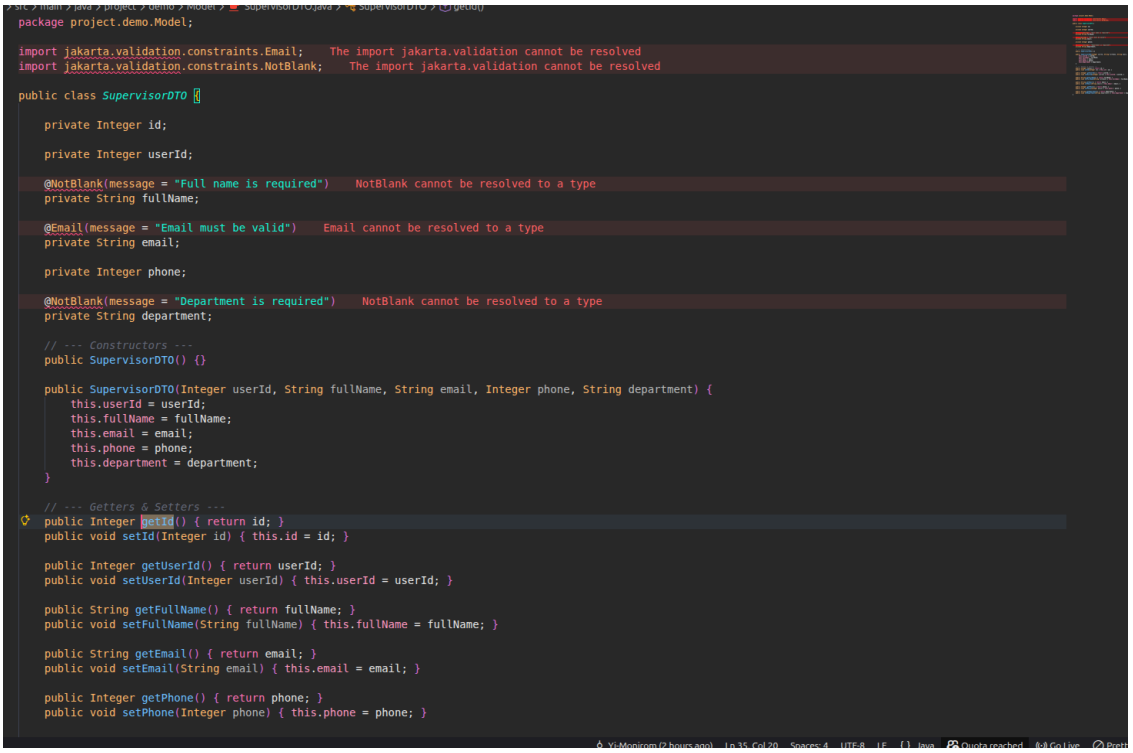
Name: YI MONIROM

Backend

- [x] ~~Create supervisor service~~
- [x] ~~Create supervisor repository~~
- [x] ~~Create supervisor entity~~
- [x] ~~Implement Create supervisor~~
- [x] ~~Implement Read supervisor~~
- [x] ~~Implement Update supervisor~~
- [x] ~~Implement Delete supervisor~~
- [x] ~~Validate inputs (e.g., name, email, dept)~~



```java
package project.demo.controller;

import jakarta.validation.Valid;    The import jakarta.validation cannot be resolved
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import project.demo.entity.Supervisor;
import project.demo.Model.SupervisorDTO;
import project.demo.repository.SupervisorRepository;

import java.util.List;
import java.util.stream.Collectors;

@RestController
@RequestMapping("/api/supervisors")
public class SupervisorController {

    @Autowired
    private SupervisorRepository supervisorRepository;

    // --- CREATE supervisor ---
    @PostMapping
    public SupervisorDTO createSupervisor(@Valid @RequestBody SupervisorDTO dto) {    Valid cannot be resolved to a type
        if (supervisorRepository.existsByUserId(dto.getUserId())) {
            throw new RuntimeException("User is already a supervisor");
        }

        Supervisor supervisor = new Supervisor();
        supervisor.setUserId(dto.getUserId());
        supervisor.setFullName(dto.getFullName());
        supervisor.setEmail(dto.getEmail());
        supervisor.setPhoneNumber(dto.getPhone());
        supervisor.setDepartment(dto.getDepartment());

        Supervisor saved = supervisorRepository.save(supervisor);

        return toDTO(saved);
    }

    // --- READ all supervisors ---
    @GetMapping
    public List<SupervisorDTO> getAllSupervisors() {
        return supervisorRepository.findAll()
                .stream()
                .map(this::toDTO)
                .collect(Collectors.toList());
    }

    // --- READ supervisor by ID ---
    @GetMapping("/{id}")
```



```java
package project.demo.Model;

import jakarta.validation.constraints.Email;    The import jakarta.validation cannot be resolved
import jakarta.validation.constraints.NotBlank;    The import jakarta.validation cannot be resolved

public class SupervisorDTO {

    private Integer id;

    private Integer userId;

    @NotBlank(message = "Full name is required")    NotBlank cannot be resolved to a type
    private String fullName;

    @Email(message = "Email must be valid")    Email cannot be resolved to a type
    private String email;

    private Integer phone;

    @NotBlank(message = "Department is required")    NotBlank cannot be resolved to a type
    private String department;

    // --- Constructors ---
    public SupervisorDTO() {}

    public SupervisorDTO(Integer userId, String fullName, String email, Integer phone, String department) {
        this.userId = userId;
        this.fullName = fullName;
        this.email = email;
        this.phone = phone;
        this.department = department;
    }

    // --- Getters & Setters ---
    public Integer getId() { return id; }
    public void setId(Integer id) { this.id = id; }

    public Integer getUserId() { return userId; }
    public void setUserId(Integer userId) { this.userId = userId; }

    public String getFullName() { return fullName; }
    public void setFullName(String fullName) { this.fullName = fullName; }

    public String getEmail() { return email; }
    public void setEmail(String email) { this.email = email; }

    public Integer getPhone() { return phone; }
    public void setPhone(Integer phone) { this.phone = phone; }
```

```java
package project.demo.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import jakarta.persistence.criteria.CriteriaBuilder.In;    The import jakarta.persistence.criteria.CriteriaBuilder.In is never used
import project.demo.entity.Supervisor;

public  interface SupervisorRepository extends JpaRepository<Supervisor,Integer> {
    boolean existsByUserId(Integer userId);


}
```

Name: SOEUN SEREYVATH

- ☑ ~~Create student service~~
- ☑ ~~Create Admin Controller~~
- ☑ ~~Create Admin Controller on Supervisor~~

```java
package project.demo.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

@Controller
public class AdminPagesController {

    @GetMapping("/admin/dashboard")
    public String dashboard(Model model) {
        model.addAttribute(attributeName: "studentCount", attributeValue: 0);
        model.addAttribute(attributeName: "supervisorCount", attributeValue: 0);
        model.addAttribute(attributeName: "applicationCount", attributeValue: 0);
        model.addAttribute(attributeName: "recentLogs", java.util.Collections.emptyList());
        return "admin-dashboard";
    }

    @GetMapping("/admin/assign-supervisor")
    public String assignSupervisor() {
        return "assign-supervisor";
    }

    @GetMapping("/admin/approvals")
    public String approvals() {
        return "approvals";
    }
}
```

```java
import java.util.List;

@Service
public class StudentService {

    private final StudentRepository repo;

    public StudentService(StudentRepository repo) {
        this.repo = repo;
    }

    public List<Student> getAll() {
        return repo.findAll();
    }

    public Student getById(Long id) {
        return repo.findById(id)    The method findById(Integer) in the type CrudRepository<Student,Integer> is not applicable for the arguments (Long)
                .orElseThrow(() -> new IllegalArgumentException("Student not found"));
    }

    public Student create(Student s) {
        if (repo.existsByEmail(s.getEmail())) {    The method existsByEmail(String) is undefined for the type StudentRepository
            throw new IllegalArgumentException("Email already exists");
        }
        return repo.save(s);
    }

    public Student update(Long id, Student s) {
        Student existing = getById(id);
        existing.setFullName(s.getFullName());
        existing.setEmail(s.getEmail());
        return repo.save(existing);
    }

    public void delete(Long id) {
        repo.deleteById(id);    The method deleteById(Integer) in the type CrudRepository<Student,Integer> is not applicable for the arguments (Long)
    }
}
```

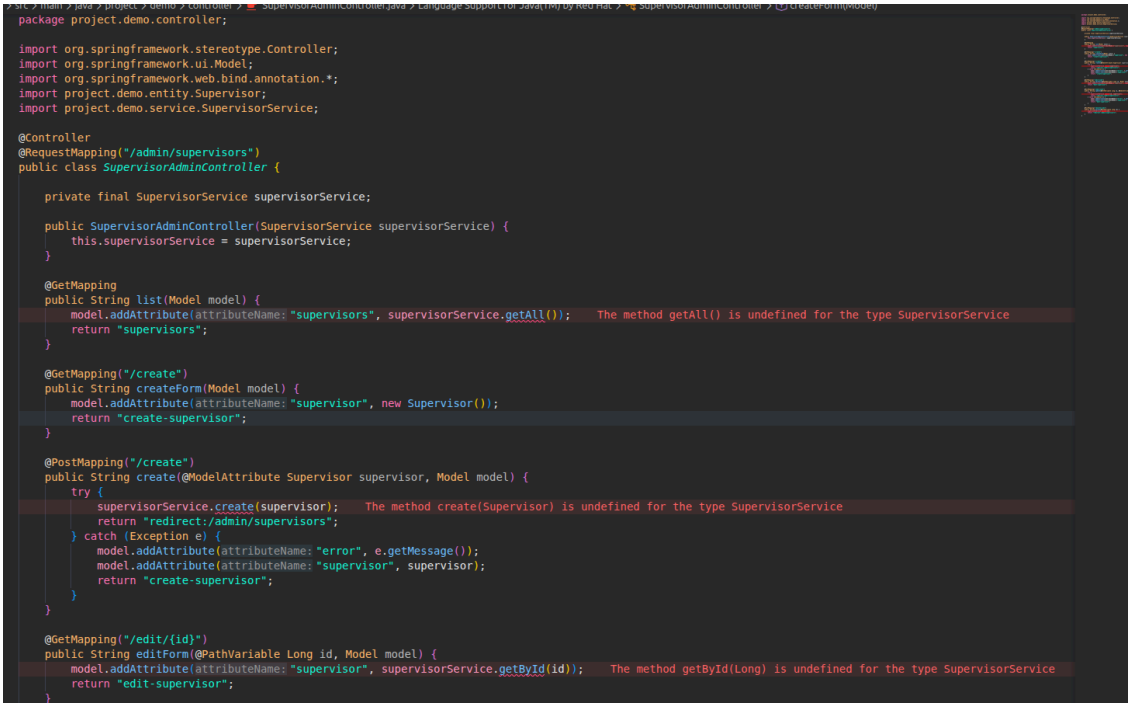
Admin CRUD Students

- [x] ~~List all students~~
- [x] ~~Create student service~~
- [x] ~~Create new student~~
- [x] ~~Edit student~~
- [x] ~~Delete student~~

```java
import org.springframework.web.bind.annotation.*;
import project.demo.entity.Student;
import project.demo.service.StudentService;

@Controller
@RequestMapping("/admin/students")
public class StudentAdminController {

    private final StudentService studentService;

    public StudentAdminController(StudentService studentService) {
        this.studentService = studentService;
    }

    // LIST
    @GetMapping
    public String list(Model model) {
        model.addAttribute(attributeName: "students", studentService.getAll());
        return "students";
    }

    // CREATE FORM
    @GetMapping("/create")
    public String createForm(Model model) {
        model.addAttribute(attributeName: "student", new Student());
        return "create-student";
    }

    // CREATE SUBMIT
    @PostMapping("/create")
    public String create(@ModelAttribute Student student, Model model) {
        try {
            studentService.create(student);
            return "redirect:/admin/students";
        } catch (Exception e) {
            model.addAttribute(attributeName: "error", e.getMessage());
            model.addAttribute(attributeName: "student", student);
            return "create-student";
        }
    }

    // EDIT FORM
    @GetMapping("/edit/{id}")
    public String editForm(@PathVariable Long id, Model model) {
        model.addAttribute(attributeName: "student", studentService.getById(id));
        return "edit-student";
    }
```

Admin CRUD on supervisor

- [x] ~~List supervisors~~
- [x] ~~Add supervisor~~
- [x] ~~Edit supervisor~~
- [x] ~~Delete supervisor~~

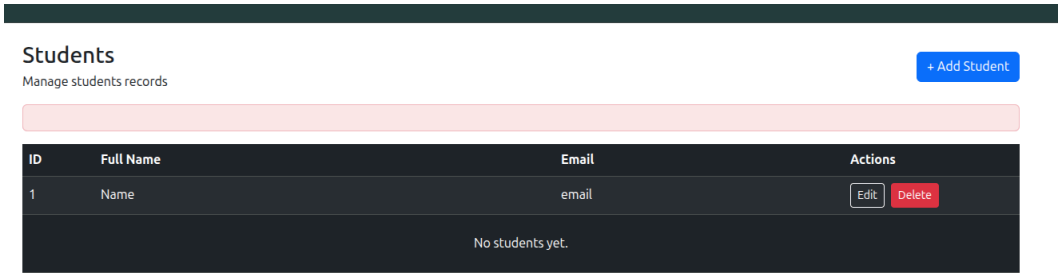


```java
package project.demo.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import project.demo.entity.Supervisor;
import project.demo.service.SupervisorService;

@Controller
@RequestMapping("/admin/supervisors")
public class SupervisorAdminController {

    private final SupervisorService supervisorService;

    public SupervisorAdminController(SupervisorService supervisorService) {
        this.supervisorService = supervisorService;
    }

    @GetMapping
    public String list(Model model) {
        model.addAttribute(attributeName: "supervisors", supervisorService.getAll());    The method getAll() is undefined for the type SupervisorService
        return "supervisors";
    }

    @GetMapping("/create")
    public String createForm(Model model) {
        model.addAttribute(attributeName: "supervisor", new Supervisor());
        return "create-supervisor";
    }

    @PostMapping("/create")
    public String create(@ModelAttribute Supervisor supervisor, Model model) {
        try {
            supervisorService.create(supervisor);    The method create(Supervisor) is undefined for the type SupervisorService
            return "redirect:/admin/supervisors";
        } catch (Exception e) {
            model.addAttribute(attributeName: "error", e.getMessage());
            model.addAttribute(attributeName: "supervisor", supervisor);
            return "create-supervisor";
        }
    }

    @GetMapping("/edit/{id}")
    public String editForm(@PathVariable Long id, Model model) {
        model.addAttribute(attributeName: "supervisor", supervisorService.getById(id));    The method getById(Long) is undefined for the type SupervisorService
        return "edit-supervisor";
    }
```

Frontend

Admin Dashboard

- [x] ~~Show statistics (students, supervisors, applications)~~
- [x] ~~Students management UI~~
- [x] ~~Supervisors management UI~~
- [x] ~~Supervisor assignment UI~~
- [x] ~~Approval UI~~



Admin Dashboard

Student Internship Management System

Tip: Use the menu to manage data quickly.

Students
0
Manage Students

Supervisors
0
Manage Supervisors

Applications
0
Approvals Assign

Recent Updates / Logs

Latest activity

No logs yet.

Time	Action	Actor	Target	Detail
time	action	actor	target	detail

Supervisors

Manage supervisors records

+ Add Supervisor

ID	Full Name	Email	Actions
1	Name	email	Edit Delete
No supervisors yet.			