D23_S1_A3_Develop a Spring Boot based web application for a simplified Social Networking Platform

```
package com.edutech.simplified_java_task3.model;

import java.util.HashSet;
import java.util.Objects;
import java.util.Set;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.ManyToMany;
import javax.persistence.Table;

import com.fasterxml.jackson.annotation.JsonIgnoreProperties;

@Entity
@Table(name = "Groups")
public class Group {
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private Long groupId;
  private String groupName;
  private String description;
  @ManyToMany(mappedBy = "groups")
  @JsonIgnoreProperties("groups")
  Set<User> users = new HashSet<>();

  public Group() {
  }

  public Group(String groupName, String description) {
    this.groupName = groupName;
    this.description = description;
  }

  public Group(String groupName, String description, Set<User> users) {
    this.groupName = groupName;
    this.description = description;
    this.users = users;
  }

  public Group(Long groupId, String groupName, String description, Set<User> users) {
    this.groupId = groupId;
    this.groupName = groupName;
    this.description = description;
    this.users = users;
  }

  public Long getGroupId() {
    return groupId;
  }

  public void setGroupId(Long groupId) {
    this.groupId = groupId;
  }
```

```
    public String getGroupName() {
        return groupName;
    }

    public void setGroupName(String groupName) {
        this.groupName = groupName;
    }

    public String getDescription() {
        return description;
    }

    public void setDescription(String description) {
        this.description = description;
    }

    public Set<User> getUsers() {
        return users;
    }

    public void setUsers(Set<User> users) {
        this.users = users;
    }

    @Override
    public boolean equals(Object o) {
        Group other = (Group) o;
        return this == other || this.getGroupId() == other.getGroupId();
    }

    @Override
    public int hashCode() {
        return Objects.hash(this.getGroupId());
    }
}

package com.edutech.simplified_java_task3.model;

import java.util.HashSet;
import java.util.Objects;
import java.util.Set;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.Table;

import com.fasterxml.jackson.annotation.JsonIgnoreProperties;

@Entity
@Table(name = "Users")
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    Long userId;
    String username;
    String email;
    String password;
    @ManyToMany()
    @JoinTable(name = "users_groups", joinColumns = @JoinColumn(name = "user_id"), inverseJoinColumns = @JoinColumn(name = "group_id"))
    @JsonIgnoreProperties("users")
    Set<Group> groups = new HashSet<>();

    public User() {
    }

    public User(String username, String email, String password) {
        this.username = username;
        this.email = email;
        this.password = password;
    }

    public User(String username, String email, String password, Set<Group> groups) {
        this.username = username;
        this.email = email;
        this.password = password;
        this.groups = groups;
    }

    public User(Long userId, String username, String email, String password, Set<Group> groups) {
        this.userId = userId;
        this.username = username;
        this.email = email;
        this.password = password;
        this.groups = groups;
    }

    public Long getUserId() {
        return userId;
    }

    public void setUserId(Long userId) {
        this.userId = userId;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getEmail() {
        return email;
    }

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

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public Set<Group> getGroups() {
        return groups;
    }

    public void setGroups(Set<Group> groups) {
        this.groups = groups;
    }

    @Override
    public boolean equals(Object o) {
        User other = (User) o;
        return this == other || this.getUserId() == other.getUserId();
    }

    @Override
    public int hashCode() {
        return Objects.hash(this.getUserId());
    }
}

package com.edutech.simplified_java_task3.repository;

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

import com.edutech.simplified_java_task3.model.Group;

//Write required code
public interface GroupRepository extends JpaRepository<Group, Long> {
}

package com.edutech.simplified_java_task3.repository;

import java.util.Optional;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;

import com.edutech.simplified_java_task3.model.Group;
import com.edutech.simplified_java_task3.model.User;

//Write required code
public interface UserRepository extends JpaRepository<User, Long> {

    // @Query("SELECT g FROM Group g LEFT JOIN FETCH g.users WHERE g.groupId = :id")
    // Optional<Group> findByIdWithUsers(@Param("id") Long id);

}

package com.edutech.simplified_java_task3.service;

import java.util.Optional;

import javax.persistence.EntityNotFoundException;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.edutech.simplified_java_task3.model.Group;
import com.edutech.simplified_java_task3.repository.GroupRepository;

@Service
public class GroupService {
   // @Autowired
   GroupRepository gr;

   public GroupService(GroupRepository gr) {
           this.gr = gr;
   }

   public Group createGroup(Group group) {
      if (group.getGroupName() == null || group.getGroupName().isEmpty()) {
         throw new IllegalArgumentException("Group name cannot be empty");
      }
      return gr.save(group);
   }

   public Optional<Group> getGroupById(Long groupId) {
      Group g = gr.findById(groupId).orElseThrow(() -> new EntityNotFoundException("Group not found"));
      return Optional.of(g);
   }
}

package com.edutech.simplified_java_task3.service;

import java.util.Optional;
import java.util.Set;

import javax.persistence.EntityNotFoundException;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.edutech.simplified_java_task3.model.Group;
import com.edutech.simplified_java_task3.model.User;
import com.edutech.simplified_java_task3.repository.GroupRepository;
import com.edutech.simplified_java_task3.repository.UserRepository;

//Write required code
@Service
public class UserService {
   // @Autowired
   UserRepository ur;
   // @Autowired
   GroupRepository gr;

   public UserService(UserRepository ur, GroupRepository gr) {
           this.ur = ur;

           this.gr = gr;
   }

   public User registerUser(User user) {
      if (user == null || user.getUsername() == null || user.getUsername().isEmpty()) {
         throw new IllegalArgumentException("Username cannot be empty");
      }
```

```
        return ur.save(user);
    }

    public Optional<User> getUserById(Long userId) {
        return ur.findById(userId);
    }

    public String joinGroup(Long userId, Long groupId) {
        User usr = ur.findById(userId).orElseThrow(() -> new EntityNotFoundException("User not found"));
        Group grp = gr.findById(groupId).orElseThrow(() -> new EntityNotFoundException("Group not found"));
        Set<Group> groups = usr.getGroups();
        groups.add(grp);
        Set<User> users = grp.getUsers();
        users.add(usr);
        usr.setGroups(groups);
        grp.setUsers(users);
        ur.save(usr);
        gr.save(grp);
        return "User joined the group successfully.";

    }
}

package com.edutech.simplified_java_task3.controller;

import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.edutech.simplified_java_task3.model.Group;
import com.edutech.simplified_java_task3.service.GroupService;

//Write required code
@RestController
@RequestMapping("/groups")
public class GroupController {
    @Autowired
    private GroupService gs;

    @PostMapping()
    public Group createGroup(@RequestBody Group group){
        return gs.createGroup(group);
    }
    @GetMapping("/{groupId}")
    public Optional<Group> getGroup(@PathVariable Long groupId) {
        return gs.getGroupById(groupId);
    }

}

package com.edutech.simplified_java_task3.controller;

import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.edutech.simplified_java_task3.model.User;
import com.edutech.simplified_java_task3.service.UserService;

//Write required code
@RestController
@RequestMapping("/users")
public class UserController {
    @Autowired
    UserService us;

    @PostMapping()
    public User registerUser(@RequestBody User user) {
        return us.registerUser(user);
    }

    @GetMapping("/{userId}")
    public Optional<User> getUserById(@PathVariable Long userId){
        return us.getUserById(userId);
    }

    @PostMapping("/{userId}/groups/{groupId}")
    public String joinGroup(@PathVariable Long userId,@PathVariable Long groupId){
        return us.joinGroup(userId, groupId);
    }

}
```

D23_S1_A2_Student-Course Management System

```
package com.edutech.entity;

import java.util.HashSet;
import java.util.Set;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.ManyToMany;

import com.fasterxml.jackson.annotation.JsonIgnore;

@Entity
public class Course {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String title;
    @ManyToMany(mappedBy = "courses")
    @JsonIgnore
    private Set<Student> students= new HashSet<>();

    public Course() {
    }

    public Course(String title) {
        this.title = title;
    }

    public Course(String title, Set<Student> students) {
        this.title = title;
        this.students = students;
    }

    public Course(Long id, String title, Set<Student> students) {
            this.id = id;
        this.title = title;
        this.students = students;
    }

    public Long getId() {
        return id;
    }

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

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public Set<Student> getStudents() {
        return students;
    }

    public void setStudents(Set<Student> students) {
        this.students = students;
    }

}

package com.edutech.entity;

import java.util.HashSet;
import java.util.Set;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;

@Entity
public class Student {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String name;
    @ManyToMany(cascade = CascadeType.PERSIST)
    @JoinTable(name = "student_course", joinColumns = @JoinColumn(name = "student_id"), inverseJoinColumns = @JoinColumn(name = "course_id"))
    private Set<Course> courses = new HashSet<>();

    public Student() {
    }

    public Student(String name) {
            this.name = name;
    }

    public Student(String name, Set<Course> courses) {
            this.name = name;
        this.courses = courses;
    }

    public Student(Long id, String name, Set<Course> courses) {
            this.id = id;
            this.name = name;
        this.courses = courses;
    }

    public Long getId() {
        return id;
    }

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

    public String getName() {
        return name;
    }

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

    public Set<Course> getCourses() {
        return courses;
    }

    public void setCourses(Set<Course> courses) {
        this.courses = courses;
    }

}
package com.edutech.exception;

import javax.persistence.EntityNotFoundException;

import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.RestControllerAdvice;
```

```java
@RestControllerAdvice
public class GlobalExceptionHandler {



}

package com.edutech.service;

import java.util.List;

import javax.persistence.EntityNotFoundException;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.edutech.entity.Course;
import com.edutech.repository.CourseRepository;
import com.edutech.repository.StudentRepository;

@Service
public class CourseService {
    @Autowired
    CourseRepository cr;
    @Autowired
    StudentRepository sr;

    public CourseService(CourseRepository cr,StudentRepository sr) {
            this.cr = cr;

            this.sr = sr;
    }

    public Course saveCourse(Course course) {
        return cr.save(course);
    }

    public void deleteCourse(Long id) {
        Course course = cr.findById(id).orElseThrow(
                () -> new EntityNotFoundException("Course not found"));
        if (!course.getStudents().isEmpty()) {
            throw new IllegalStateException("Cannot delete course with enrolled students");
        }
        cr.deleteById(id);
    }

    public List<Course> getAllCourses() {
        return cr.findAll();
    }

}


package com.edutech.service;

import java.util.List;

import javax.persistence.EntityNotFoundException;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.edutech.entity.Student;
import com.edutech.repository.CourseRepository;
import com.edutech.repository.StudentRepository;

@Service
public class StudentService {
   @Autowired
   StudentRepository sr;
   @Autowired
   CourseRepository cr;

   public StudentService(StudentRepository sr, CourseRepository cr) {
          this.sr = sr;

          this.cr = cr;
   }

   public Student saveStudent(Student student) {
      return sr.save(student);
   }

   public void deleteStudent(Long id) {
      if (!sr.existsById(id)) {
         throw new EntityNotFoundException("Student not found");
      }
      sr.deleteById(id);
   }

   public List<Student> getAllStudents() {
      return sr.findAll();
   }

}

package com.edutech.repository;

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

import com.edutech.entity.Course;

public interface CourseRepository extends JpaRepository<Course, Long> {
}
package com.edutech.repository;

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

import com.edutech.entity.Student;

public interface StudentRepository extends JpaRepository<Student, Long> {
}
```

D23_S1_A1_Project -Employee relationship management

```
package com.ltedutech.entity;

import java.util.HashSet;
import java.util.Set;

import javax.persistence.*;
```

```
@Entity
public class Employee {
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private Long id;
  private String name;
  @ManyToMany()
  @JoinTable(name = "Employee_Project", joinColumns = @JoinColumn(name = "employee_id"), inverseJoinColumns = @JoinColumn(name =
"project_id"))
  private Set<Project> projects = new HashSet<>();

  public Employee() {
  }

  public Employee(String name) {
      this.name = name;
  }

  public Employee(Set<Project> projects) {
    this.projects = projects;
  }

  public Employee(String name, Set<Project> projects) {
      this.name = name;
    this.projects = projects;
  }

  public Employee(Long id, String name, Set<Project> projects) {
      this.id = id;
      this.name = name;
    this.projects = projects;
  }

  public Long getId() {
    return id;
  }

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

  public String getName() {
    return name;
  }

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

  public Set<Project> getProjects() {
    return projects;
  }

  public void setProjects(Set<Project> projects) {
    this.projects = projects;
  }
}
package com.ltedutech.entity;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity
public class Project {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;
   private String name;

   public Project() {
   }

   public Project(String name) {
           this.name = name;
   }

   public Project(Long id, String name) {
           this.id = id;
           this.name = name;
   }

   public Long getId() {
      return id;
   }

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

   public String getName() {
      return name;
   }

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

}

package com.ltedutech.repository;

import com.ltedutech.entity.Employee;
import org.springframework.data.jpa.repository.JpaRepository;


public interface EmployeeRepository extends JpaRepository<Employee,Long> {
}
package com.ltedutech.repository;

import com.ltedutech.entity.Project;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
```

```
@Repository
public interface ProjectRepository extends JpaRepository<Project,Long> {
}

package com.ltedutech.service;

import com.ltedutech.entity.Employee;

import java.util.List;
import java.util.Optional;

public interface EmployeeService {
 Employee saveEmployee(Employee employee);
 Optional<Employee> getEmployeeById(Long id);
 List<Employee> getAllEmployees();
 Optional<Employee> updateEmployee(Long id, Employee employeeDetails);
}

package com.ltedutech.service;

import com.ltedutech.entity.Employee;
import com.ltedutech.entity.Project;
import com.ltedutech.repository.EmployeeRepository;
import com.ltedutech.repository.ProjectRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.HashSet;
import java.util.List;
import java.util.Optional;
import java.util.Set;

@Service
public class EmployeeServiceImpl implements EmployeeService {
  @Autowired
  EmployeeRepository er;

  public EmployeeServiceImpl(EmployeeRepository er) {
        this.er = er;
  }

  @Override
  public Employee saveEmployee(Employee employee) {
    return er.save(employee);
  }

  @Override
  public Optional<Employee> getEmployeeById(Long id) {
    return er.findById(id);
  }

  @Override
  public List<Employee> getAllEmployees() {
    return er.findAll();
  }

  @Override
  public Optional<Employee> updateEmployee(Long id, Employee employeeDetails) {
    Employee employee = er.findById(id).orElse(null);
```

```
        if (employee == null) {
            throw new RuntimeException("null");
        }
        employee.setName(employeeDetails.getName());
        employee.setProjects(employeeDetails.getProjects());
        return Optional.of(er.save(employee));
    }

}
package com.ltedutech.service;

import com.ltedutech.entity.Project;

import java.util.List;
import java.util.Optional;

public interface ProjectService {
   Project saveProject(Project project);

   Optional<Project> getProjectById(Long id);

   List<Project> getAllProjects();
}

package com.ltedutech.service;

import com.ltedutech.entity.Project;
import com.ltedutech.repository.ProjectRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;
import java.util.Optional;

@Service
public class ProjectServiceImpl implements ProjectService {
    @Autowired
    ProjectRepository pr;

    public ProjectServiceImpl(ProjectRepository pr) {
            this.pr = pr;
    }

    @Override
    public Project saveProject(Project project) {
        return pr.save(project);
    }

    @Override
    public Optional<Project> getProjectById(Long id) {
        return pr.findById(id);
    }

    @Override
    public List<Project> getAllProjects() {
        return pr.findAll();
    }

}

package com.ltedutech.controller;
```

```
import com.ltedutech.entity.Employee;
import com.ltedutech.service.EmployeeService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;
import java.util.Optional;

@RestController
@RequestMapping("/employees")
public class EmployeeController {
  @Autowired
  EmployeeService es;

  @PostMapping("/add")
  public ResponseEntity<Employee> createEmployee(@RequestBody Employee employee) {
    return new ResponseEntity<>(es.saveEmployee(employee), HttpStatus.OK);
  }

  @GetMapping("/{id}")
  public ResponseEntity<Employee> getEmployeeById(@PathVariable Long id) {
    Employee employee = es.getEmployeeById(id).orElse(null);
    if (employee == null) {
      throw new RuntimeException();
    }
    return new ResponseEntity(employee, HttpStatus.OK);
  }

  @GetMapping("/all")
  public ResponseEntity<List<Employee>> getAllEmployees() {
    return new ResponseEntity<>(es.getAllEmployees(), HttpStatus.OK);
  }

  @PutMapping("/update/{id}")
  public ResponseEntity<Employee> updateEmployee(@PathVariable Long id, @RequestBody Employee employeeDetails) {
    return new ResponseEntity<>(es.updateEmployee(id, employeeDetails).get(), HttpStatus.OK);
  }

  @ExceptionHandler(RuntimeException.class)
  public ResponseEntity<String> handleRuntimeException(RuntimeException ex) {
    return new ResponseEntity<>(ex.getMessage(), HttpStatus.NOT_FOUND);
  }
}

package com.ltedutech.controller;

import com.ltedutech.entity.Project;
import com.ltedutech.service.ProjectService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
import java.util.List;
import java.util.Optional;

@RestController
@RequestMapping("/projects")
```

```
public class ProjectController {
  @Autowired
  ProjectService ps;

  public ProjectController(ProjectService ps) {
        this.ps = ps;
  }

  @PostMapping("/add")
  ResponseEntity<Project> createProject(@RequestBody Project project) {
    return new ResponseEntity<>(ps.saveProject(project), HttpStatus.OK);
  }

  @GetMapping("/{id}")
  ResponseEntity<Project> getProjectById(@PathVariable Long id) {
    Project p = ps.getProjectById(id).orElse(null);
    if (p==null) {
      throw new RuntimeException();
    }
    return new ResponseEntity<>(p, HttpStatus.OK);
  }

  @GetMapping("/all")
  ResponseEntity<List<Project>> getAllProjects() {
    return new ResponseEntity<>(ps.getAllProjects(), HttpStatus.OK);
  }

  @ExceptionHandler(RuntimeException.class)
  public ResponseEntity<String> handleRuntimeException(RuntimeException ex) {
    return new ResponseEntity<>(ex.getMessage(), HttpStatus.NOT_FOUND);
  }
}
```