D22_S3_A1_Develop a Spring Boot based web application for an Online Library System

```
package com.edutech.simplified_java_task2.model;

import com.fasterxml.jackson.annotation.JsonIgnore;

import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;

@Entity
public class Author {
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private Long authorId;
  private String firstName;
  private String lastName;
  private String email;
  @OneToMany(mappedBy = "author", cascade = CascadeType.ALL,orphanRemoval = true)
  @JsonIgnore
  private List<Book> books;

  public Author() {
    this.books = new ArrayList<>();
  }

  public Author(String firstName, String lastName, String email) {
    this.firstName = firstName;
    this.lastName = lastName;
    this.email = email;
    this.books = new ArrayList<>();
  }

  public Author(String firstName, String lastName, String email, List<Book> books) {
    this.firstName = firstName;
    this.lastName = lastName;
    this.email = email;
    this.books = books;
  }

  public Long getAuthorId() {
    return authorId;
  }

  public void setAuthorId(Long authorId) {
    this.authorId = authorId;
  }

  public String getFirstName() {
    return firstName;
  }

  public void setFirstName(String firstName) {
    this.firstName = firstName;
  }

  public String getLastName() {
    return lastName;
  }
```

```
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getEmail() {
        return email;
    }

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

    public List<Book> getBooks() {
        return books;
    }

    public void setBooks(List<Book> books) {
        this.books = books;
    }

}
package com.edutech.simplified_java_task2.model;

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

@Entity
public class Book {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long bookId;
   private String title;
   private String genre;
   private int publicationYear;
   @ManyToOne()
   @JoinColumn(name = "author_id", nullable = false)
   private Author author;

   public Book() {
   }

   public Book(String title, String genre, int publicationYear, Author author) {
     this.title = title;
     this.genre = genre;
     this.publicationYear = publicationYear;
     this.author = author;
   }

   public Book(Long bookId, String title, String genre, int publicationYear, Author author) {
     this.bookId = bookId;
     this.title = title;
     this.genre = genre;
     this.publicationYear = publicationYear;
     this.author = author;
   }
```

```
  public Long getBookId() {
    return bookId;
  }

  public void setBookId(Long bookId) {
    this.bookId = bookId;
  }

  public String getTitle() {
    return title;
  }

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

  public String getGenre() {
    return genre;
  }

  public void setGenre(String genre) {
    this.genre = genre;
  }

  public int getPublicationYear() {
    return publicationYear;
  }

  public void setPublicationYear(int publicationYear) {
    this.publicationYear = publicationYear;
  }

  public Author getAuthor() {
    return author;
  }

  public void setAuthor(Author author) {
    this.author = author;
  }

}
package com.edutech.simplified_java_task2.repository;

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

import com.edutech.simplified_java_task2.model.Author;


public interface AuthorRepository extends JpaRepository<Author,Long> {
}
package com.edutech.simplified_java_task2.repository;

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

import com.edutech.simplified_java_task2.model.Book;

public interface BookRepository extends JpaRepository<Book, Long> {
}
package com.edutech.simplified_java_task2.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.edutech.simplified_java_task2.model.Author;
import com.edutech.simplified_java_task2.repository.AuthorRepository;

@Service
public class AuthorService {
    @Autowired
    AuthorRepository ar;

    public AuthorService(AuthorRepository ar) {
        this.ar = ar;
    }

    public Author saveAuthor(Author author){
        return ar.save(author);
    }

}
package com.edutech.simplified_java_task2.service;

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

import com.edutech.simplified_java_task2.model.Author;
import com.edutech.simplified_java_task2.model.Book;
import com.edutech.simplified_java_task2.repository.AuthorRepository;
import com.edutech.simplified_java_task2.repository.BookRepository;

@Service
public class BookService {
    @Autowired
    BookRepository br;
    @Autowired
    AuthorRepository ar;

    public BookService(BookRepository br, AuthorRepository ar) {
        this.br = br;
        this.ar = ar;
    }

    public Book saveBook(Long authorId, Book book) {
        if (book == null || book.getTitle() == null || book.getTitle().trim().isEmpty()) {
            throw new IllegalArgumentException("Book title cannot be empty");
        }
        Author a = ar.findById(authorId)
                .orElseThrow(() -> new IllegalArgumentException("Invalid author Id: " + authorId));
        book.setAuthor(a);
        return br.save(book);
    }

    public Book getBook(Long id) {
        return br.findById(id).orElseThrow(() -> new RuntimeException(""));
    }
}
package com.edutech.simplified_java_task2.controller;

import com.edutech.simplified_java_task2.model.Author;
import com.edutech.simplified_java_task2.service.AuthorService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
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;

@RestController
@RequestMapping("/authors")
public class AuthorController {
   @Autowired
   AuthorService as;

   @PostMapping()
   public ResponseEntity<Author> createAuthor(@RequestBody Author author){
      return new ResponseEntity<>(as.saveAuthor(author),HttpStatus.OK);
   }
}
package com.edutech.simplified_java_task2.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
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_task2.model.Book;
import com.edutech.simplified_java_task2.service.BookService;

@RestController
@RequestMapping("/books")
public class BookController {
  @Autowired
  BookService bs;

  @PostMapping("/{authorId}")
  public ResponseEntity<Book> createBook(@PathVariable Long authorId, @RequestBody Book book) {
    return new ResponseEntity<>(bs.saveBook(authorId, book), HttpStatus.OK);
  }

  @GetMapping("/{bookId}")
  public ResponseEntity<Book> getBook(@PathVariable Long bookId) {
    return new ResponseEntity<>(bs.getBook(bookId), HttpStatus.OK);
  }
}
```

D22_S2_A2_Library System using Spring Boot

```
package com.edutech.entity;

import java.util.ArrayList;
import java.util.List;

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

@Entity
public class Author {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;
   private String name;
   @OneToMany(mappedBy = "author", cascade = CascadeType.ALL, orphanRemoval = true)
   private List<Book> books;

   public Author() {
      this.books = new ArrayList<>();
   }

   public Author(String name) {
         this.name = name;
      this.books = new ArrayList<>();
   }

   public Author(String name, List<Book> books) {
         this.name = name;
      this.books = books;
   }

   public Author(Long id, String name, List<Book> books) {
         this.id = id;
         this.name = name;
      this.books = books;
   }

   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 List<Book> getBooks() {
      return books;
   }

   public void setBooks(List<Book> books) {
      this.books = books;
```

```
    }

}
package com.edutech.entity;

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

import com.fasterxml.jackson.annotation.JsonIgnore;

@Entity
public class Book {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String title;
    @ManyToOne
    @JoinColumn(name = "author_id")
    @JsonIgnore
    private Author author;

    public Book() {
    }

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

    public Book(String title, Author author) {
        this.title = title;
        this.author = author;
    }

    public Book(Long id, String title, Author author) {
            this.id = id;
        this.title = title;
        this.author = author;
    }

    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 Author getAuthor() {
        return author;
    }

    public void setAuthor(Author author) {
        this.author = author;
    }

}
package com.edutech.exceptionhandling;

import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.ResponseStatus;

@ResponseStatus(HttpStatus.NOT_FOUND)
public class ResourceNotFoundException extends RuntimeException {

    public ResourceNotFoundException(String string) {
        super(string);
    }

}
package com.edutech.repository;

import java.util.List;

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

import com.edutech.entity.Author;

public interface AuthorRepository extends JpaRepository<Author, Long> {
    @Query("SELECT a FROM Author a")
    List<Author> findAllAuthors();
}

package com.edutech.repository;

import java.util.List;

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

import com.edutech.entity.Book;

public interface BookRepository extends JpaRepository<Book, Long> {
    @Query("SELECT b FROM Book b JOIN FETCH b.author")
    List<Book> findAllBooksWithAuthors();
}

package com.edutech.service;

import com.edutech.entity.Author;
import com.edutech.exceptionhandling.ResourceNotFoundException;
import com.edutech.repository.AuthorRepository;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import java.util.List;

@Service
public class AuthorService {
    @Autowired
    private final AuthorRepository ar;

    public AuthorService(AuthorRepository ar) {
            this.ar = ar;
    }

    public Author createAuthor(Author author) {
        if (author == null || author.getName() == null || author.getName().trim().isEmpty()) {
            throw new RuntimeException("Invalid author");
        }
        return ar.save(author);
    }

    public void deleteAuthorById(Long id) {
        Author a = ar.findById(id).orElseThrow(
            ()-> new ResourceNotFoundException("Author not found"));
        if (a.getBooks() != null && !a.getBooks().isEmpty()) {
            throw new ResourceNotFoundException("Author has associated books  found");
        }
        if (!a.getBooks().isEmpty()) {
            throw new ResourceNotFoundException("");
        }
        ar.deleteById(id);
    }

    public List<Author> getAllAuthors() {
        return ar.findAllAuthors();
    }
}

package com.edutech.service;

import com.edutech.entity.Author;
import com.edutech.entity.Book;
import com.edutech.exceptionhandling.ResourceNotFoundException;
import com.edutech.repository.AuthorRepository;
import com.edutech.repository.BookRepository;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import java.util.List;

@Service
public class BookService {
    @Autowired
    final BookRepository br;
    @Autowired
    final AuthorRepository ar;

    public BookService(BookRepository br, AuthorRepository ar) {
            this.br = br;
            this.ar = ar;
    }

    public Book createBook(Book book, Long authorId) {
        if (book == null || book.getTitle() == null || book.getTitle().trim().isEmpty()) {
            throw new IllegalArgumentException("Book title cannot be empty");
    }
    Author author = ar.findById(authorId).orElseThrow(() -> new ResourceNotFoundException("Author not found"));
    book.setAuthor(author);
    return br.save(book);
  }

  public void deleteBookById(Long id) {
    if (!br.existsById(id)) {
      throw new ResourceNotFoundException("Book not found");

    }
    br.deleteById(id);

  }

  public List<Book> getAllBooks() {
    return br.findAllBooksWithAuthors();
  }

}
```

D22_S2_A1_University Management System

```
package com.edutech.entity;

import javax.persistence.*;

@Entity
public class Course {
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private Long id;
  private String name;
  private int credits;
  @ManyToOne
  @JoinColumn(name = "department_id")
  private Department department;

  public Course() {
  }

  public Course(String name, int credits, Department department) {
        this.name = name;
    this.credits = credits;
    this.department = department;
  }

  public Course(Long id, String name, int credits, Department department) {
        this.id = id;

        this.name = name;
    this.credits = credits;
    this.department = department;
  }

  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 int getCredits() {
    return credits;
  }

  public void setCredits(int credits) {
    this.credits = credits;
  }

  public Department getDepartment() {
    return department;
  }

  public void setDepartment(Department department) {
    this.department = department;
  }

}
package com.edutech.entity;

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

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

  public Department() {
  }

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

  public Department(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.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.Department;

public interface DepartmentRepository extends JpaRepository<Department, Long> {
}
package com.edutech.service;

import java.util.List;

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

import com.edutech.ResourceNotFoundException;
import com.edutech.entity.Course;
import com.edutech.entity.Department;
import com.edutech.repository.CourseRepository;
import com.edutech.repository.DepartmentRepository;

@Service
public class CourseService {
    @Autowired
    CourseRepository cr;
    @Autowired
    DepartmentRepository dr;

    public Course saveCourse(Course course) {
        try {
            Department d = dr.findById(course.getDepartment().getId()).orElse(null);
            if (d != null) {
                course.setDepartment(d);
                return course;
            }
        } catch (Exception e) {
            throw new ResourceNotFoundException("Department not found");
        }
```

```java
        return null;
    }

    public Course getCourseById(Long id) {
        Course c = cr.findById(id).orElse(null);
        if (c == null) {
            throw new ResourceNotFoundException("Course not found");
        }
        return c;
    }
    public List<Course> getAllCourses(){
        return cr.findAll();
    }
}
package com.edutech.service;

import java.util.List;

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

import com.edutech.ResourceNotFoundException;
import com.edutech.entity.Department;
import com.edutech.repository.DepartmentRepository;

@Service
public class DepartmentService {
    @Autowired
    DepartmentRepository dr;

    public Department saveDepartment(Department department) {
        return dr.save(department);
    }

    public Department getDepartmentById(Long id) {
        Department d = dr.findById(id).orElse(null);
        if (d == null) {
            throw new ResourceNotFoundException("Department not found");
        }
        return d;
    }

    public List<Department> getAllDepartments() {
        return dr.findAll();
    }
}
package com.edutech;

public class ResourceNotFoundException extends RuntimeException {

    public ResourceNotFoundException(String string) {
        super(string);
    }

}
```

D22_S1_A3_Car and Engine Management REST API Using JPA

```java
package com.edutech.model;
```

```
import com.fasterxml.jackson.annotation.JsonManagedReference;
import javax.persistence.*;

@Entity
public class Car {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;
   private String model;
   @OneToOne(mappedBy = "car", cascade = CascadeType.ALL)
   @JsonManagedReference
   private Engine engine;

   public Car() {
   }

   public Car(String model) {
      this.model = model;
   }

   public Car(String model, Engine engine) {
      this.model = model;
      this.engine = engine;
   }

   public Car(Long id, String model, Engine engine) {
           this.id = id;
      this.model = model;
      this.engine = engine;
   }

   public Long getId() {
      return id;
   }

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

   public String getModel() {
      return model;
   }

   public void setModel(String model) {
      this.model = model;
   }

   public Engine getEngine() {
      return engine;
   }

   public void setEngine(Engine engine) {
      this.engine = engine;
   }

}
package com.edutech.model;

import com.fasterxml.jackson.annotation.JsonBackReference;
import javax.persistence.*;

@Entity
public class Engine {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;
   private String type;
   @OneToOne
   @JoinColumn(name = "car_id", unique = true)
   @JsonBackReference
   private Car car;

   public Engine() {
   }

   public Engine(String type) {
      this.type = type;
   }

   public Engine(String type, Car car) {
      this.type = type;
      this.car = car;
   }

   public Engine(Long id, String type, Car car) {
         this.id = id;
      this.type = type;
      this.car = car;
   }

   public Long getId() {
      return id;
   }

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

   public String getType() {
      return type;
   }

   public void setType(String type) {
      this.type = type;
   }

   public Car getCar() {
      return car;
   }

   public void setCar(Car car) {
      this.car = car;
   }

}
package com.edutech.repository;

import org.springframework.data.jpa.repository.JpaRepository;
import com.edutech.model.Car;

public interface CarRepository extends JpaRepository<Car,Long> {
}

package com.edutech.repository;

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

import com.edutech.model.Engine;

public interface EngineRepository extends JpaRepository<Engine,Long> {
}
package com.edutech.service;

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

import com.edutech.exception.ResourceNotFoundException;
import com.edutech.model.Car;
import com.edutech.repository.CarRepository;
import com.edutech.repository.EngineRepository;
import com.edutech.model.Engine;

import java.util.List;

@Service
public class CarService {
  @Autowired
  CarRepository cr;
  @Autowired
  EngineRepository er;

  public Car createCar(Car car) {
    return cr.save(car);
  }

  public Car updateCar(Long id, Car carDetails) {
    Car car = cr.findById(id).orElse(null);
    if (car == null || id == null) {
      throw new ResourceNotFoundException("Car not found");
    }
    return cr.save(carDetails);
  }

  public Car getCarById(Long id) {
    Car car = cr.findById(id).orElse(null);
    if (car == null || id == null) {
      throw new ResourceNotFoundException("Car not found");
    }
    return car;
  }

  public List<Car> getAllCars() {
    return cr.findAll();
  }

  public Car assignEngineToCar(Long carId, Long engineId) {
    Car car = cr.findById(carId).orElse(null);
    Engine engine = er.findById(engineId).orElse(null);
    if (engine.getCar() != null) {
            throw new RuntimeException("The engine is already assigned to another car");
        }
        car.setEngine(engine);
        engine.setCar(car);
        return car;
    }

}
package com.edutech.service;

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

import com.edutech.exception.ResourceNotFoundException;
import com.edutech.model.Car;
import com.edutech.model.Engine;
import com.edutech.repository.CarRepository;
import com.edutech.repository.EngineRepository;

import java.util.List;

@Service
public class EngineService {
    @Autowired
    EngineRepository er;
    @Autowired
    CarRepository cr;

    public Engine createEngine(Engine engine) {
        return er.save(engine);
    }

    public Engine updateEngine(Long id, Engine engineDetails) {
        Engine engine = er.findById(id).orElse(null);
        Car c1 = engine.getCar();
        Car c2 = engineDetails.getCar();
        if (c1 != null && c2 != null && c1.getId() != c2.getId()) {
            throw new IllegalArgumentException("Car cannot be modified from Engine update");
        }
        return er.save(engineDetails);
    }

    public Engine getEngineById(Long id) {
        Engine engine = er.findById(id).orElse(null);
        if (engine == null) {
            throw new ResourceNotFoundException("Engine not found");
        }
        return engine;
    }

    public List<Engine> getAllEngines() {
        return er.findAll();
    }

}
package com.edutech.exception;

import java.util.Date;

public class ErrorDetails {
    private Date timestamp;
    private String message;
    private String details;

    public ErrorDetails() {
    }

    public ErrorDetails(Date timestamp, String message, String details) {
        this.timestamp = timestamp;
        this.message = message;
        this.details = details;
    }

    public Date getTimestamp() {
        return timestamp;
    }

    public void setTimestamp(Date timestamp) {
        this.timestamp = timestamp;
    }

    public String getMessage() {
        return message;
    }

    public void setMessage(String message) {
        this.message = message;
    }

    public String getDetails() {
        return details;
    }

    public void setDetails(String details) {
        this.details = details;
    }

}
package com.edutech.exception;

import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.context.request.WebRequest;

import java.time.Instant;
import java.util.Date;

@ControllerAdvice
public class GlobalExceptionHandler {
    @ExceptionHandler(ResourceNotFoundException.class)
    public ResponseEntity<?> resourceNotFoundException(ResourceNotFoundException ex, WebRequest request) {
        Date timeStamp = new Date(Instant.now().toEpochMilli());
        String message = ex.getMessage();
        String details = request.getDescription(false);
        ErrorDetails e = new ErrorDetails(timeStamp, message, details);
        return new ResponseEntity<>(e, HttpStatus.NOT_FOUND);
    }

    @ExceptionHandler(Exception.class)
    public ResponseEntity<?> globalExceptionHandler(Exception ex, WebRequest request) {
        Date timeStamp = new Date(Instant.now().toEpochMilli());
        String message = ex.getMessage();
        String details = request.getDescription(false);
        ErrorDetails e = new ErrorDetails(timeStamp, message, details);
        return new ResponseEntity<>(e, HttpStatus.INTERNAL_SERVER_ERROR);

    }
}
package com.edutech.exception;

import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.ResponseStatus;

@ResponseStatus(HttpStatus.NOT_FOUND)
public class ResourceNotFoundException extends RuntimeException {

    public ResourceNotFoundException(String string) {
        super(string);
    }

}
```

D22_S1_A2_Passport Management Spring Boot Rest API

```
package com.edutech.model;

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

import com.fasterxml.jackson.annotation.JsonIgnore;

@Entity
public class Person {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String name;
    @OneToOne(mappedBy = "person", cascade = CascadeType.ALL)
    @JsonIgnore
    private Passport passport;

    public Person() {
    }



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


    public Person(String name, Passport passport) {
            this.name = name;
        this.passport = passport;
    }



    public Person(Long id, String name, Passport passport) {
            this.id = id;
            this.name = name;
        this.passport = passport;
    }

    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 Passport getPassport() {
        return passport;
    }

    public void setPassport(Passport passport) {
        this.passport = passport;
    }
}

package com.edutech.model;

import javax.persistence.*;

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

    @Column(unique = true)
    private String passportNumber;

    @OneToOne
    @JoinColumn(name = "person_id", unique = true)
    private Person person;

    public Passport() {
    }



    public Passport(String passportNumber) {
        this.passportNumber = passportNumber;
    }

    public Passport(String passportNumber, Person person) {
        this.passportNumber = passportNumber;
        this.person = person;
    }

    public Passport(Long id, String passportNumber, Person person) {
            this.id = id;
        this.passportNumber = passportNumber;
        this.person = person;
    }

    public Long getId() {
        return id;
    }

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

    public String getPassportNumber() {
        return passportNumber;
    }

    public void setPassportNumber(String passportNumber) {
        this.passportNumber = passportNumber;
    }

    public Person getPerson() {
        return person;
    }

    public void setPerson(Person person) {
        this.person = person;
    }
}
package com.edutech.exception;

import javax.persistence.EntityNotFoundException;

import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;

@ControllerAdvice
public class GlobalExceptionHandler {
    @ExceptionHandler(EntityNotFoundException.class)
    public ResponseEntity<String> handleEntityNotFoundException(EntityNotFoundException ex) {
        return new ResponseEntity<>(ex.getMessage(), HttpStatus.NOT_FOUND);
    }

    @ExceptionHandler(Exception.class)
    public ResponseEntity<String> handleGeneralException(Exception ex) {
        return new ResponseEntity<>(ex.getMessage(), HttpStatus.INTERNAL_SERVER_ERROR);
    }
}
```

```
package com.edutech.repository;

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

import com.edutech.model.Passport;

public interface PassportRepository extends JpaRepository<Passport,Long> {
}

package com.edutech.repository;

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

import com.edutech.model.Person;

public interface PersonRepository extends JpaRepository<Person, Long> {

  @Query("SELECT p FROM Person p WHERE p.passport.passportNumber = :passportNumber")
  Person findByPassportNumber(@Param("passportNumber") String passportNumber);


}
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.model.Passport;
import com.edutech.model.Person;
import com.edutech.repository.PassportRepository;
import com.edutech.repository.PersonRepository;

@Service
public class PassportService {
  @Autowired
  PassportRepository passportRepo;
  @Autowired
  PersonRepository personRepo;

  public Passport savePassport(Passport passport) {
    Person per = passport.getPerson();
    if (per == null || per.getId() == null) {
      throw new EntityNotFoundException(
          "Person ID not found, please provide correct person id or you can create a new person");
    }
    passportRepo.save(passport);
    return passport;
  }

  public Passport assignPersonToPassport(Long passportId, Long personId) {
    Passport pass = passportRepo.findById(passportId).orElse(null);
    Person per = personRepo.findById(personId).orElse(null);
    if (personId == null) {
      throw new EntityNotFoundException(
          "Person ID not found, please provide correct person id or you can create a new person");
    }
```

```
        if (pass == null) {
            throw new EntityNotFoundException("Passport not found with id: " + passportId);
        }
        if (per == null) {
            throw new EntityNotFoundException("Person not found with id: " + personId);
        }
        pass.setPerson(per);
        per.setPassport(pass);
        personRepo.save(per);
        passportRepo.save(pass);
        return pass;
    }

    public void deletePassportById(Long id) {
        Passport pass = passportRepo.findById(id).orElse(null);
        if (pass == null) {
            throw new EntityNotFoundException("Passport not found with id: " + id);
        }
        String passPortNumber = pass.getPassportNumber();
        Person per = personRepo.findByPassportNumber(passPortNumber);
        passportRepo.deleteById(id);
        if (per != null) {
            per.setPassport(null);
            personRepo.save(per);
        }
    }

    public List<Passport> getAllPassports() {
        return passportRepo.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.model.Person;
import com.edutech.repository.PassportRepository;
import com.edutech.repository.PersonRepository;

@Service
public class PersonService {
    @Autowired
    PersonRepository pr;
    @Autowired
    private PassportRepository ppr;

    public Person savePerson(Person person) {
        person.setPassport(null);
        return pr.save(person);
    }

    public void deletePersonById(Long id) {
        Person person = pr.findById(id).orElse(null);
        if (person == null) {
            throw new EntityNotFoundException("Person not found with id: " + id);
        }
        if (person.getPassport() != null) {
            throw new IllegalStateException(
                    "Cannot delete person with id: " + id + " as it has an associated passport.");
        }


        pr.delete(person);
    }

    public List<Person> getAllPersons() {
        return pr.findAll();
    }
}
package com.edutech.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import com.edutech.model.Passport;
import com.edutech.service.PassportService;

import java.util.List;

@RestController
@RequestMapping("/passports")
public class PassportController {
    @Autowired
    PassportService ps;

    @PostMapping()
    public ResponseEntity<Passport> createPassport(@RequestBody Passport passport) {
        return new ResponseEntity<>(ps.savePassport(passport), HttpStatus.CREATED);
    }

    @PostMapping("/{passportId}/assign-person/{personId}")
    public ResponseEntity<Passport> assignPersonToPassport(@PathVariable Long passportId, @PathVariable Long personId) {
        return new ResponseEntity<>(ps.assignPersonToPassport(passportId, personId), HttpStatus.OK);
    }

    @DeleteMapping("/{id}")
    public ResponseEntity<Void> deletePassport(@PathVariable Long id) {
        ps.deletePassportById(id);
        return ResponseEntity.noContent().build();
    }

    @GetMapping()
    public ResponseEntity<List<Passport>> getAllPassports() {
        return new ResponseEntity<>(ps.getAllPassports(), HttpStatus.OK);
    }
}
package com.edutech.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.DeleteMapping;
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.model.Person;
import com.edutech.service.PersonService;

@RestController
@RequestMapping("/persons")
public class PersonController {
   @Autowired
   PersonService ps;

   @PostMapping()
   public ResponseEntity<Person> createPerson(@RequestBody Person person) {
      return new ResponseEntity<>(ps.savePerson(person), HttpStatus.CREATED);
   }

   @DeleteMapping("/{id}")
   public ResponseEntity<String> deletePerson(@PathVariable Long id) {
      ps.deletePersonById(id);
      return new ResponseEntity<>("The person with id " + id + " is deleted", HttpStatus.OK);
   }

   @GetMapping()
   public ResponseEntity<List<Person>> getAllPersons() {
      return new ResponseEntity<>(ps.getAllPersons(), HttpStatus.OK);
   }
}
```

D22_S1_A1_User Management System

```
package com.edutech.model;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.OneToOne;
import javax.persistence.Table;

@Entity
@Table(name = "users")
public class Users {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;
   @Column(nullable = false)
   private String name;

   @OneToOne
   @JoinColumn(name = "profile_id", referencedColumnName = "id", unique = true, nullable = false)
   private Profiles profile;
```

```
    public Users() {
    }

    public Users(String name, Profiles profile) {
        this.name = name;
        this.profile = profile;
    }

    public Users(Long id, String name, Profiles profile) {
        this.id = id;
        this.name = name;
        this.profile = profile;
    }

    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 Profiles getProfile() {
        return profile;
    }

    public void setProfile(Profiles profile) {
        this.profile = profile;
    }

}

package com.edutech.model;

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

@Entity
@Table(name = "profiles")
public class Profiles {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String email;
    private String phoneNumber;
    public Profiles() {
    }

    public Profiles(String email, String phoneNumber) {
        this.email = email;
        this.phoneNumber = phoneNumber;
    }

    public Profiles(Long id, String email, String phoneNumber) {
            this.id = id;
        this.email = email;
        this.phoneNumber = phoneNumber;
    }

    public Long getId() {
        return id;
    }

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

    public String getEmail() {
        return email;
    }

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

    public String getPhoneNumber() {
        return phoneNumber;
    }

    public void setPhoneNumber(String phoneNumber) {
        this.phoneNumber = phoneNumber;
    }

}


package com.edutech.exception;

import javax.validation.ConstraintViolationException;

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

@RestControllerAdvice
public class GlobalExceptionHandler {

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

    @ExceptionHandler(ConstraintViolationException.class)
    public ResponseEntity<String> handleConstraintViolationException(ConstraintViolationException ex) {
        return new ResponseEntity<>(ex.getMessage(), HttpStatus.BAD_REQUEST);
    }
}

package com.edutech.exception;


public class ProfileNotFoundException extends Exception{

    public ProfileNotFoundException(String message) {
        super(message);
    }


}
package com.edutech.repository;

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

import com.edutech.model.Profiles;

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

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

import com.edutech.model.Users;

public interface UserRepository extends JpaRepository<Users, Long> {
}

package com.edutech.service;

import java.util.List;

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

import com.edutech.model.Profiles;
import com.edutech.repository.ProfileRepository;

@Service
public class ProfileService {
    @Autowired
    private ProfileRepository pr;

    public Profiles saveProfile(Profiles profile) {
        return pr.save(profile);
    }

    public List<Profiles> getAllProfiles() {
        return pr.findAll();
    }
}
package com.edutech.service;

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

import com.edutech.exception.ProfileNotFoundException;
import com.edutech.model.Profiles;
import com.edutech.model.Users;
import com.edutech.repository.ProfileRepository;
import com.edutech.repository.UserRepository;

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

import javax.validation.ConstraintViolationException;

@Service
public class UserService {
 @Autowired
 UserRepository ur;
 @Autowired
 ProfileRepository pr;

 public Users saveUser(Users user) {
  try {
   return ur.save(user);
  } catch (Exception e) {
   throw new ConstraintViolationException("A database constraint was violated, Profile already mapped to other user",
     null);
  }
 }

 public Optional<Users> getUserById(Long id) {
  return ur.findById(id);
 }

 public List<Users> getAllUsers() {
  return ur.findAll();
 }

 public void processUserProfile(Users user) throws ProfileNotFoundException {
  Long profileId = user.getId();
  Profiles profile = pr.findById(profileId).orElse(null);
  if (profile == null) {
   throw new ProfileNotFoundException("Profile not found with ID: " + profileId);
  }
  user.setProfile(profile);
 }
}
```