Complaint Redressal System

Phase 6 Project

Source Code

1. Main

```
package com.crs.main;


import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;


@SpringBootApplication
public class CrsApplication {


	public static void main(String[] args) {
		SpringApplication.run(CrsApplication.class, args);
	}


}
```

2. Controller

- Complaints Controller

```
package com.crs.main.controller;


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


import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
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.PutMapping;
```

```
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import com.crs.main.model.Complaints;
import com.crs.main.repository.ComplaintRepository;
//import com.crs.main.service.ComplaintService;
import com.crs.main.service.ComplaintServiceImpl;
//import com.crs.main.service.ComplaintsMySqlDAO;

@CrossOrigin("*")
@RestController
@RequestMapping(path="/complaints")
public class ComplaintsController {


	@Autowired
	ComplaintServiceImpl complaintServiceImpl;

	@Autowired
	ComplaintRepository complaintRepository;

	@GetMapping("/getAllComplaints")
	public List<Complaints> getAllComplaints(){
		List<Complaints> complaints = (List<Complaints>) complaintRepository.findAll();
		return complaints;
	}
	@GetMapping("/getAllComplaintsByEmail/{customerEmail}")
	public List<Complaints> getAllComplaintsByEmail(@PathVariable("customerEmail") String
customerEmail){
```

```
		System.out.println("inside complaints controller -- "+customerEmail);

		List<Complaints> complaints = (List<Complaints>)
complaintServiceImpl.findComplaintByEmail(customerEmail);
		return (List<Complaints>) complaints;
	}



	@PostMapping("/getAllComplaintsByTicketIds")
	public List<Complaints> getAllComplaintsByTicketIds(@RequestBody int[] ticketIds){

		System.out.println("ticket Ids from Backend -- "+ticketIds);
		List<Complaints> complaints = new ArrayList<Complaints>();
		for (int i=0; i<ticketIds.length;i++ ) {
			System.out.println(ticketIds[i]);
			complaints.add(complaintServiceImpl.findComplaintById(ticketIds[i]));
		}
		return  (List<Complaints>) complaints;


	}

	@GetMapping("/getComplaintById/{ticketId}")
	public Complaints getComplaintById(@PathVariable("ticketId") int ticketId){
		Complaints complaint = complaintServiceImpl.findComplaintById(ticketId);
		return complaint;
	}

	@PostMapping(path = "/addComplaint")
	public @ResponseBody void addComplaint(@RequestBody Complaints complaint) {
		complaintServiceImpl.saveComplaint(complaint);
```

```
	}

	@DeleteMapping(path = "/deleteComplaintById/{ticketId}")
	public @ResponseBody void deleteComplaintById(@PathVariable("ticketId") int ticketId) {
		Complaints complaint =complaintServiceImpl.findComplaintById(ticketId);
		complaintServiceImpl.deleteComplaint(complaint);

	}
	@PutMapping("/updateComplaint")
	public boolean updateComplaint(@RequestBody Complaints complaint)
	{
		System.out.println(complaint.getTicketId()+"-------");
		Complaints existingComplaint =
complaintServiceImpl.findComplaintById(complaint.getTicketId());
		existingComplaint.setComplaint(complaint.getComplaint());
		existingComplaint.setCustomerEmail(complaint.getCustomerEmail());
		existingComplaint.setPincode(complaint.getPincode());
		existingComplaint.setStatus(complaint.getStatus());
		complaintRepository.save(existingComplaint);

		return true;
	}

}
```

- CustomerController

```
package com.crs.main.controller;

import java.util.LinkedHashMap;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.web.bind.annotation.CrossOrigin;
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.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import com.crs.main.model.Customers;

import com.crs.main.repository.CustomerRepository;
import com.crs.main.service.CustomerService;

@CrossOrigin("*")
@RestController
@RequestMapping(path="/customers")
public class CustomerController {

	@Autowired
	CustomerService customerService;
	@Autowired
	CustomerRepository customerRepository;

	@GetMapping("/getAllCustomers")
	public List<Customers> getAllCustomers(){
		List<Customers> customers = (List<Customers>)
customerService.fetchAllCustomers();
```

```
		return customers;
	}

	@PostMapping("/login")
	public Boolean validateCustomer(@RequestBody Object loginDetails) throws
NoSuchFieldException {

			String customerEmail = (String) ((LinkedHashMap)
loginDetails).get("customerEmail");
			String customerPassword = (String) ((LinkedHashMap)
loginDetails).get("customerPassword");

			Boolean customerLoginStatus =
customerService.validateCustomer(customerEmail,customerPassword);
			return customerLoginStatus;
	}

	@PostMapping(path = "/addCustomer")
	public @ResponseBody void addCustomer(@RequestBody Customers customer) {
		customerService.saveCustomer(customer);

	}

	@DeleteMapping(path = "/deleteCustomer/{customerEmail}")
	public @ResponseBody void deleteCustomer(@PathVariable("customerEmail") String email)
{
		Customers customer =customerService.findCustomerById(email);
		customerService.deleteCustomer(customer);

	}
	@PutMapping("/updateCustomer/{customerEmail}")
	public boolean updateManager(@PathVariable("customerEmail") String
email,@RequestBody String newPincode) {
```

```
                System.out.println(newPincode+"----------");
                Customers customer =customerService.findCustomerById(email);
                customer.setCustomerPincode(newPincode);
                customerRepository.save(customer);
                return true;

        }
}
```

- EngineerController

```
package com.crs.main.controller;

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

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
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.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import com.crs.main.model.EngineerDuty;
import com.crs.main.model.Engineers;
import com.crs.main.service.EngineerDutyServiceImpl;
import com.crs.main.service.EngineerService;
```

```
@CrossOrigin("*")
@RestController
@RequestMapping(path="/engineers")
public class EngineerController {

	@Autowired
	EngineerService engineerService;

	@Autowired
	EngineerDutyServiceImpl engineerDutyServiceImpl;

	@GetMapping("/getAllEngineers")
	public List<Engineers> getAllEngineers(){
		List<Engineers> engineers = (List<Engineers>) engineerService.fetchAllEngineers();
		return engineers;
	}

	@GetMapping("/getAllEngineerMails")
	public List<String> getAllEngineerMails(){
		List<Engineers> engineers = (List<Engineers>) engineerService.fetchAllEngineers();
		List<String> engineerMails = new ArrayList<String>();
		for (int i =0; i< engineers.size(); i++) {
			engineerMails.add(engineers.get(i).getEngineerEmail());
		}
		return engineerMails;
	}

	@PostMapping("/login")
	public List<Integer> validateEngineer(@RequestBody Object loginDetails) throws
NoSuchFieldException {
```

```
                String engineerEmail = (String) ((LinkedHashMap)
loginDetails).get("engineerEmail");
                String engineerPassword = (String) ((LinkedHashMap)
loginDetails).get("engineerPassword");
                List<Integer> ticketIds = new ArrayList<Integer>();
                List<EngineerDuty> engineerDuties = (List<EngineerDuty>)
engineerDutyServiceImpl.findEngineerDutyByEmail(engineerEmail);

                Boolean engineerLoginStatus =
engineerService.validateEngineer(engineerEmail,engineerPassword );
                if(engineerLoginStatus) {
                        for(int i=0;i<engineerDuties.size();i++) {
                                ticketIds.add(engineerDuties.get(i).getTicketId());
                        }

                        return ticketIds;
                }
                return null;

        }


        @PostMapping(path = "/addEngineer")
        public @ResponseBody void addEngineer(@RequestBody Engineers engineer) {
                engineerService.saveEngineer(engineer);

        }

        @DeleteMapping(path = "/deleteEngineer/{engineerEmail}")
        public @ResponseBody void deleteEngineer(@PathVariable("engineerEmail") String email) {
                Engineers engineer =engineerService.findEngineerById(email);
```

```
                engineerService.deleteEngineer(engineer);

        }

}
```

- EngineerDutyController

```
package com.crs.main.controller;

import java.util.LinkedHashMap;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.GetMapping;
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.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import com.crs.main.model.Complaints;
import com.crs.main.model.EngineerDuty;
import com.crs.main.service.ComplaintServiceImpl;
import com.crs.main.service.EngineerDutyService;
import com.crs.main.service.EngineerDutyServiceImpl;

@CrossOrigin("*")
@RestController
@RequestMapping("/engineerDuty")
public class EngineerDutyController {
```

```
	@Autowired
	EngineerDutyService engineerDutyService;

	@Autowired
	EngineerDutyServiceImpl engineerDutyServiceImpl;

	@Autowired
	ComplaintServiceImpl complaintServiceImpl;



	@PostMapping("/addEngineerDuty")
	public boolean addEngineerDuty(@RequestBody Object engineersDutyAssigned) throws NoSuchFieldException{
		int ticketId = (int) ((LinkedHashMap)engineersDutyAssigned).get("ticketId");
		String customerEmail = (String)((LinkedHashMap) engineersDutyAssigned).get("customerEmail");
		String selectedEngineer = (String)((LinkedHashMap) engineersDutyAssigned).get("selectedEngineer");
		System.out.println(ticketId+"---"+customerEmail+"--"+selectedEngineer);

		EngineerDuty existingEngineersDuty = engineerDutyServiceImpl.getEngineerAssignedByTicketId(ticketId);
		if (existingEngineersDuty==null) {

			EngineerDuty engineerDuty = new EngineerDuty();

			engineerDuty.setTicketId(ticketId);
			engineerDuty.setCustomerEmail(customerEmail);
			engineerDuty.setEngineerEmail(selectedEngineer);

			engineerDutyService.saveEngineerDuty(engineerDuty);
```

```
                return true;
            }
            return false;
        }
        @PostMapping("/updateStatus")
        public boolean updateStatus(@RequestBody Object statusUpdate)throws
NoSuchFieldException{
            int ticketId = (int) ((LinkedHashMap)statusUpdate).get("ticketId");
            String newStatus = (String)((LinkedHashMap) statusUpdate).get("status");
            System.out.println("----------"+newStatus);
            Complaints complaint = complaintServiceImpl.findComplaintById(ticketId);
            if(complaint!=null) {
                complaint.setStatus(newStatus);
                complaintServiceImpl.saveComplaint(complaint);
                return true;
            }
            return false;
        }

}
```

- FeedbackController

```
package com.crs.main.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
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.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import com.crs.main.model.Complaints;
import com.crs.main.model.Feedbacks;
import com.crs.main.repository.FeedbackRepository;
import com.crs.main.service.FeedbackService;
import com.crs.main.service.FeedbackServiceImpl;

@CrossOrigin("*")
@RestController
@RequestMapping(path="/feedbacks")
public class FeedbackController {

	@Autowired
	FeedbackService feedbackService;

	@Autowired
	FeedbackServiceImpl feedbackServiceImpl;

	@Autowired
	FeedbackRepository feedbackRepository;

	@GetMapping("/getAllFeedbacks")
	public List<Feedbacks> getAllFeedbacks(){
		List<Feedbacks> feedbacks = (List<Feedbacks>) feedbackService.fetchAllFeedbacks();
		return feedbacks;
```

```
	}

	@PostMapping(path = "/addFeedback")
	public @ResponseBody void addFeedback(@RequestBody Feedbacks feedback) {
		System.out.println(feedback.getFeedback());
		feedbackService.saveFeedback(feedback);

	}

	@DeleteMapping(path = "/deleteFeedbackById/{feedbackId}")
	public @ResponseBody void deleteComplaintById(@PathVariable("feedbackId") int
feedbackId) {
		System.out.println(feedbackId+"---------- ");
		Feedbacks feedback =feedbackServiceImpl.findFeedbackById(feedbackId);
		feedbackServiceImpl.deleteFeedback(feedback);

	}
	@PutMapping("/updateFeedback")
	public boolean updateFeedback(@RequestBody Feedbacks feedback)
	{
		System.out.println(feedback.getTicketId()+"-------");
		Feedbacks existingFeedbacks =
feedbackServiceImpl.findFeedbackById(feedback.getFeedbackId());
		existingFeedbacks.setTicketId(feedback.getTicketId());
		existingFeedbacks.setCustomerEmail(feedback.getCustomerEmail());
		existingFeedbacks.setFeedback(feedback.getFeedback());

		feedbackRepository.save(existingFeedbacks);

		return true;
	}
}
```

- ManagerController

```
package com.crs.main.controller;

import java.util.LinkedHashMap;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
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.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import com.crs.main.model.Complaints;
import com.crs.main.model.Managers;
import com.crs.main.repository.ManagerRepository;
import com.crs.main.service.ManagerService;
import com.crs.main.service.ManagerServiceImpl;

@CrossOrigin("*")
@RestController
@RequestMapping(path="/managers")
public class ManagerController {

	@Autowired
	ManagerService managerService;
```

```
	@Autowired
	ManagerRepository managerRepository;

	@Autowired
	ManagerServiceImpl managerServiceImpl;

	@GetMapping("/getAllManagers")
	public List<Managers> getAllManagers(){
		List<Managers> managers = (List<Managers>) managerRepository.findAll();
		return managers;
	}

	@PostMapping(path = "/addManager")
	public @ResponseBody void addManager(@RequestBody Managers manager) {
		managerService.saveManager(manager);

	}

	@PostMapping("/login")
	public Managers validateManager(@RequestBody Object loginDetails) throws
NoSuchFieldException {

			String managerEmail = (String) ((LinkedHashMap)
loginDetails).get("managerEmail");
			String managerPassword = (String) ((LinkedHashMap)
loginDetails).get("managerPassword");

			Boolean managerLoginStatus =
managerService.validateManager(managerEmail,managerPassword );
			if (managerLoginStatus) {
				Managers manager=
managerRepository.findById(managerEmail).get();
```

```
                    return manager;
                }
                return null;
        }

        @GetMapping("/getAllComplaintsByPincode/{managerPincode}")
        public List<Complaints> getAllComplaintsByPincode(@PathVariable("managerPincode")
String managerPincode){
                System.out.println("inside managers controller -- "+managerPincode);

                List<Complaints> complaints = (List<Complaints>)
managerServiceImpl.findComplaintByPincode(managerPincode);
                return (List<Complaints>) complaints;
        }

        @DeleteMapping(path = "/deleteManager/{managerEmail}")
        public @ResponseBody void deleteManager(@PathVariable("managerEmail") String email) {
                Managers manager =managerService.findManagerById(email);
                managerService.deleteManager(manager);

        }
        @PutMapping("/updateManager/{managerEmail}")
        public boolean updateManager(@PathVariable("managerEmail") String
email,@RequestBody String newPincode) {
                System.out.println(newPincode+"----------");
                Managers manager =managerService.findManagerById(email);
                manager.setManagerPincode(newPincode);
                managerRepository.save(manager);
                return true;

        }
}
```

3. Models

- Complaints

```
package com.crs.main.model;


import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;


import lombok.Getter;

import lombok.Setter;


@Entity

@Table(name="tbl_complaints")

@Setter

@Getter

public class Complaints {

	@Id

	@GeneratedValue(strategy = GenerationType.AUTO)

	private int ticketId;


	private String customerEmail;

	private String pincode;

	private String complaint;

	private String status = "raised";


}
```

- Customers

```
package com.crs.main.model;
```

```
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

import lombok.Getter;
import lombok.Setter;

@Entity
@Table(name="tbl_customer")
@Setter
@Getter

public class Customers {

	@Id
	private String customerEmail;

	private String customerPassword;
	private String customerName;
	private String customerMobile;
	private String customerAddress;
	private String customerPincode;
}
```

- EngineersDuty

```
package com.crs.main.model;

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

```
import lombok.Getter;
import lombok.Setter;

@Entity
@Table(name="tbl_engineerDuty")
@Setter
@Getter
public class EngineerDuty {

	@Id
	@GeneratedValue(strategy = GenerationType.AUTO)
	private int engineerDutyId;

	private String engineerEmail;
	private int ticketId;
	private String customerEmail;

}
```

- Engineers

```
package com.crs.main.model;

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

import lombok.Getter;
import lombok.Setter;

@Entity
@Table(name="tbl_engineer")
```

```
@Setter
@Getter
public class Engineers {

	@Id
	private String engineerEmail;

	private String engineerPassword;
	private String engineerName;

}
```

- Feedbacks

```
package com.crs.main.model;

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

import lombok.Getter;
import lombok.Setter;

@Entity
@Table(name="tbl_feedbacks")
@Setter
@Getter
public class Feedbacks {

	@Id
	@GeneratedValue(strategy = GenerationType.AUTO)
```

```
	private int feedbackId;


	private int ticketId;
	private String customerEmail;
	private String feedback;
}
```

- Managers

```
package com.crs.main.model;

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

import lombok.Getter;
import lombok.Setter;

@Entity
@Table(name="tbl_manager")
@Setter
@Getter
public class Managers {


	@Id
	private String managerEmail;
	private String managerPassword;
	private String managerName;
	private String managerPincode;
}
```

4. Repository

- ComplaintRepository

```
package com.crs.main.repository;
```

```
import java.util.List;

import org.springframework.data.repository.CrudRepository;

import com.crs.main.model.Complaints;

public interface ComplaintRepository extends CrudRepository<Complaints, Integer> {

//	Iterable<Complaints> findAll(Sort sort);
	List<Complaints> findComplaintByCustomerEmail(String customerEmail);

	List<Complaints> findComplaintByPincode(String managerPincode);

}
```

- CustomerRepository

```
package com.crs.main.repository;

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

import com.crs.main.model.Customers;

@Repository
public interface CustomerRepository extends JpaRepository<Customers, String> {

}
```

- EngineerDutyRepository

```
package com.crs.main.repository;

import java.util.List;
```

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

import com.crs.main.model.EngineerDuty;

public interface EngineerDutyRepository  extends CrudRepository<EngineerDuty, Integer> {

	List<EngineerDuty> findEngineerDutyByEngineerEmail(String engineerEmail);

	EngineerDuty findEngineerDutyByTicketId(int ticketId);

}
```

- EngineerRepository

```
package com.crs.main.repository;

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

import com.crs.main.model.Engineers;

public interface EngineerRepository extends JpaRepository<Engineers, String> {

}
```

- FeedbackRepository

```
package com.crs.main.repository;

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

import com.crs.main.model.Feedbacks;

public interface FeedbackRepository extends JpaRepository<Feedbacks, Integer> {
```

}

- ManagerRepository

```
package com.crs.main.repository;

import java.util.List;

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

import com.crs.main.model.Complaints;
import com.crs.main.model.Managers;

public interface ManagerRepository extends CrudRepository<Managers, String> {

}
```

5. Service

- complaintServiceImpl

```
package com.crs.main.service;

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

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

import com.crs.main.model.Complaints;
import com.crs.main.repository.ComplaintRepository;

@Service
```

```
public class ComplaintServiceImpl  {

	@Autowired
	ComplaintRepository complaintRepository;

//	@Override
//	public List<Complaints> fetchAllComplaints() {
//		return (List<Complaints>) complaintRepository.findAll();
//	}

//	@Override
	public void saveComplaint(Complaints complaint) {
		complaintRepository.save(complaint);

	}

//	@Override
	public Complaints findComplaintById(int ticketId) {
		Complaints complaint = complaintRepository.findById(ticketId).orElse(null);
		return complaint;
	}

//	@Override
	public void deleteComplaint(Complaints complaint) {
		complaintRepository.delete(complaint);

	}

//	@Override
	public List<Complaints> findComplaintByEmail(String customerEmail) {
		return complaintRepository.findComplaintByCustomerEmail(customerEmail);
```

```
        }

}
```

- CustomerService

```
package com.crs.main.service;

import java.util.List;

import com.crs.main.model.Customers;

public interface CustomerService {

        List <Customers> fetchAllCustomers();
        void saveCustomer(Customers customer);
        Customers findCustomerById(String email);
        void deleteCustomer(Customers customer);
        Boolean validateCustomer(String customerEmail, String customerPassword);
}
```

- CustomerServiceImpl

```
package com.crs.main.service;

import java.util.List;

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

import com.crs.main.model.Customers;
import com.crs.main.repository.CustomerRepository;

@Service
public class CustomerServiceImpl implements CustomerService {
```

```
@Autowired
CustomerRepository customerRepository;

@Override
public List<Customers> fetchAllCustomers() {
	return customerRepository.findAll();
}

@Override
public void saveCustomer(Customers customer) {
	customerRepository.save(customer);
}

@Override
public Customers findCustomerById(String email) {
	Customers customer = customerRepository.findById(email).orElse(null);
	return customer;
}

@Override
public void deleteCustomer(Customers customer) {
	customerRepository.delete(customer);
}

@Override
public Boolean validateCustomer(String customerEmail, String customerPassword) {

	if (customerRepository.findById(customerEmail).isPresent()) {
		Customers customer = customerRepository.findById(customerEmail).get();
		String dbPassword = customer.getCustomerPassword().toString();
```

```
                    if (dbPassword.equals(customerPassword)) {
                        return true;
                    }
                }
                return false;
        }

}
```

- EngineersDutyService

```
package com.crs.main.service;

import com.crs.main.model.EngineerDuty;

public interface EngineerDutyService {
	void saveEngineerDuty(EngineerDuty engineerDuty);

}
```

- EngineersDutyServiceImpl

```
package com.crs.main.service;

import java.util.List;

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

import com.crs.main.model.EngineerDuty;
import com.crs.main.repository.EngineerDutyRepository;

@Service
public class EngineerDutyServiceImpl implements EngineerDutyService {

	@Autowired
	EngineerDutyRepository engineerDutyRepository;
```

```
	@Override
	public void saveEngineerDuty(EngineerDuty engineerDuty) {
		engineerDutyRepository.save(engineerDuty);
	}
	public List<EngineerDuty> findEngineerDutyByEmail(String engineerEmail) {

		return engineerDutyRepository.findEngineerDutyByEngineerEmail(engineerEmail);
	}
	public EngineerDuty getEngineerAssignedByTicketId(int ticketId) {

		return engineerDutyRepository.findEngineerDutyByTicketId(ticketId);
	}

}
```

- EngineerServiceImpl

```
package com.crs.main.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.crs.main.model.Engineers;
import com.crs.main.repository.EngineerRepository;

@Service
public class EngineerSerivceImpl implements EngineerService {

	@Autowired
	EngineerRepository engineerRepository;

	@Override
```

```
public List<Engineers> fetchAllEngineers() {
	return engineerRepository.findAll();
}

@Override
public void saveEngineer(Engineers engineer) {
	engineerRepository.save(engineer);

}

@Override
public Engineers findEngineerById(String email) {
	Engineers engineer = engineerRepository.findById(email).orElse(null);
	return engineer;
}

@Override
public void deleteEngineer(Engineers engineer) {
	engineerRepository.delete(engineer);

}

@Override
public Boolean validateEngineer(String engineerEmail, String engineerPassword) {
	System.out.println(engineerEmail + " --- "+engineerPassword);
	if (engineerRepository.findById(engineerEmail).isPresent()) {
		Engineers engineer= engineerRepository.findById(engineerEmail).get();
		String dbPassword = engineer.getEngineerPassword().toString();
		if (dbPassword.equals(engineerPassword)) {
			return true;
		}
```

```
                }
                return false;
        }

}
```

- engineersService

```
package com.crs.main.service;

import java.util.List;

import com.crs.main.model.Engineers;

public interface EngineerService {

        List <Engineers> fetchAllEngineers();
        void saveEngineer(Engineers engineer);
        Engineers findEngineerById(String email);
        void deleteEngineer(Engineers engineer);
        Boolean validateEngineer(String engineerEmail, String engineerPassword);
}
```

- FeedbackService

```
package com.crs.main.service;

import java.util.List;

import com.crs.main.model.Feedbacks;

public interface FeedbackService {
        List<Feedbacks> fetchAllFeedbacks();
        void saveFeedback (Feedbacks feedback);
}
```

- FeedbackServiceImpl

```
package com.crs.main.service;

import java.util.List;

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

import com.crs.main.model.Feedbacks;
import com.crs.main.repository.FeedbackRepository;

@Service
public class FeedbackServiceImpl implements FeedbackService {

	@Autowired
	FeedbackRepository feedbackRepository;

	@Override
	public List<Feedbacks> fetchAllFeedbacks() {
		return feedbackRepository.findAll();
	}

	@Override
	public void saveFeedback(Feedbacks feedback) {
		feedbackRepository.save(feedback);

	}

	public Feedbacks findFeedbackById(int feedbackId) {
		Feedbacks feedback = feedbackRepository.findById(feedbackId).orElse(null);
		return feedback;
```

```
	}

	public void deleteFeedback(Feedbacks feedback) {
		feedbackRepository.delete(feedback);

	}

}
```

- ManagerService

```
package com.crs.main.service;

import java.util.List;

import com.crs.main.model.Managers;

public interface ManagerService {

//	List <Managers> fetchAllManagers();
	void saveManager(Managers manager);
	Managers findManagerById(String email);
	void deleteManager(Managers manager);
	Boolean validateManager(String managerEmail, String managerPassword);
}
```

- ManagerServiceImpl

```
package com.crs.main.service;

import java.util.List;

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

```
import com.crs.main.model.Complaints;
import com.crs.main.model.Managers;
import com.crs.main.repository.ComplaintRepository;
import com.crs.main.repository.ManagerRepository;

@Service
public class ManagerServiceImpl implements ManagerService {

	@Autowired
	ManagerRepository managerRepository;

	@Autowired
	ComplaintRepository complaintRepository;

//	@Override
//	public List<Managers> fetchAllManagers() {
//		return managerRepository.findAll();
//	}

	@Override
	public void saveManager(Managers manager) {
		managerRepository.save(manager);

	}

	@Override
	public Managers findManagerById(String email) {
		Managers manager = managerRepository.findById(email).orElse(null);
		return manager;
	}
```

```
	@Override
	public void deleteManager(Managers manager) {
		managerRepository.delete(manager);

	}

	@Override
	public Boolean validateManager(String managerEmail, String managerPassword) {
		System.out.println(managerEmail + " --- "+managerPassword);
		if (managerRepository.findById(managerEmail).isPresent()) {
			Managers manager= managerRepository.findById(managerEmail).get();
			String dbPassword = manager.getManagerPassword().toString();
			if (dbPassword.equals(managerPassword)) {
				return true;
			}
		}
		return false;
	}

	public List<Complaints> findComplaintByPincode(String managerPincode) {
		return complaintRepository.findComplaintByPincode(managerPincode);
	}

}
```

6. Application Properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/ComplaintRedressalSystem?useSSL=
false
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.jpa.database=mysql
spring.jpa.hibernate.ddl-auto=update
```

7. Pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
	<modelVersion>4.0.0</modelVersion>
	<parent>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-parent</artifactId>
		<version>2.6.3</version>
		<relativePath/> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.crs</groupId>
	<artifactId>CRS</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>CRS</name>
	<description>Complaint Redressal System</description>
	<properties>
		<java.version>1.8</java.version>
	</properties>
	<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-jpa</artifactId>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-web</artifactId>
		</dependency>
	 <dependency>
	  <groupId>org.projectlombok</groupId>
	  <artifactId>lombok</artifactId>
	  <version>1.18.20</version>
	  <scope>provided</scope>
    </dependency>

		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-devtools</artifactId>
			<scope>runtime</scope>
			<optional>true</optional>
		</dependency>
		<dependency>
			<groupId>mysql</groupId>
			<artifactId>mysql-connector-java</artifactId>
			<scope>runtime</scope>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-test</artifactId>
			<scope>test</scope>
		</dependency>
		<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.6</version>
</dependency>
```

```
	</dependencies>

	<build>
		<plugins>
			<plugin>
				<groupId>org.springframework.boot</groupId>
				<artifactId>spring-boot-maven-plugin</artifactId>
			</plugin>
		</plugins>
	</build>

</project>
```