```
package com.insurance.management;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.boot.autoconfigure.domain.EntityScan;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.data.jpa.repository.config.EnableJpaRepositories;


@SpringBootApplication

@ComponentScan({"com"})

@EnableJpaRepositories(basePackages= {"com.insurance.management.repository"})

@EntityScan("com.insurance.management.model")

public class ManagementApplication {

	public static void main(String[] args) {

		SpringApplication.run(ManagementApplication.class, args);

		System.out.println("started.....");

	}

}
```

Controller.java:-

```
package com.insurance.management.controller;

import java.util.List;

import java.util.Optional;

import javax.servlet.http.HttpServletRequest;

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

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.validation.annotation.Validated;
```

```java
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.RestController;

import com.insurance.management.model.Admin;

import com.insurance.management.model.Approvals;

import com.insurance.management.model.Approved;

import com.insurance.management.model.AuthenticationStatus;

import com.insurance.management.model.Category;

import com.insurance.management.model.Disapproved;

import com.insurance.management.model.Login;

import com.insurance.management.model.Policies;

import com.insurance.management.model.Queries;

import com.insurance.management.model.User;

import com.insurance.management.service.AdminService;

import com.insurance.management.service.ApprovalsService;

import com.insurance.management.service.ApprovedService;

import com.insurance.management.service.CategoryService;

import com.insurance.management.service.DisapprovedService;

import com.insurance.management.service.LoginService;

import com.insurance.management.service.PoliciesService;

import com.insurance.management.service.QueriesService;
import com.insurance.management.service.UserService;


@CrossOrigin("*")

@RequestMapping("/api/v1")

@RestController

public class Controller {

	@Autowired

	AdminService adminService;

	@Autowired

	private ApprovalsService approvalsService;

	@Autowired

	private ApprovedService approvedService;

	@Autowired

	CategoryService categoryService;

	@Autowired

	private DisapprovedService disapprovedService;

	@Autowired

	PoliciesService policiesService;

	@Autowired

	QueriesService queriesService;

	@Autowired

	UserService userService;

	@Autowired

	LoginService loginService;


	@PostMapping("/adminlogin")

	public ResponseEntity<AuthenticationStatus> validateAdminLogin(@RequestBody Admin
adminlogin) {
		System.out.println(adminlogin.getUsername() + " " + adminlogin.getPassword());

		AuthenticationStatus status =
adminService.validateAdminLogin(adminlogin.getUsername(),

				adminlogin.getPassword());

		return new ResponseEntity<AuthenticationStatus>(status, HttpStatus.OK);


	}


	@GetMapping("/approvals")

	public List<Approvals> getAllApprovals() {

		return approvalsService.getAllApprovals();

	}


	@GetMapping("/approvals/{userName}")

	public Optional<Approvals> getApproval(@PathVariable("userName") String userName) {

		return approvalsService.getApprovals(userName);

	}


	@PostMapping("/approvals")

	public String addApprovals(@RequestBody Approvals approvals) {

		approvalsService.addApproval(approvals);

		return "added";

	}


	@GetMapping("/approved")

	public List<Approved> getAllApproved() {

		return approvedService.getAllApproved();

	}
	@GetMapping("/approved/{userName}")
	public Optional<Approved> getApproved(@PathVariable("userName") String userName) {
		return approvedService.getApproved(userName);
	}

	@PostMapping("/approved")
	public String addApproved(@RequestBody Approved approved) {
		approvedService.addApproved(approved);
		return "added";
	}

	@GetMapping("/categories")
	public List<Category> getInsuranceCategory() {
		List<Category> list = categoryService.fetchCategory();
		return list;
	}

	@GetMapping("/category/{category}")
	public ResponseEntity<Category> getCategoryById(@PathVariable("category") String
category) {
		Category iCategory = categoryService.getCategory(category);
		return ResponseEntity.ok().body(iCategory);
	}

	@PostMapping("/category")
	public String addInsuranceCategory(@RequestBody Category category) {
		categoryService.SaveCategory(category);
		return "saved";
	}

	@DeleteMapping(value = "/category/{category}")
	public ResponseEntity<Object> deleteCategory(@PathVariable("category") String category) {

		categoryService.deleteCategory(category);
		return new ResponseEntity<>("Insurance category deleted successsfully",
HttpStatus.OK);
	}
	@GetMapping("/disapproved")
	public List<Disapproved> getAllDisapproved() {
		return disapprovedService.getAllDisapproved();
	}
	@GetMapping("/disapproved/{userName}")
	public Optional<Disapproved> getDisapproved(@PathVariable("userName") String
userName) {
		return disapprovedService.getDisapproved(userName);
	}

	@PostMapping("/disapproved")
	public String addDisapproved(@RequestBody Disapproved disapproved) {
		disapprovedService.addDisapproved(disapproved);
		return "added";
	}

	@GetMapping("/policies")
	public List<Policies> getPolicies() {
		List<Policies> list = policiesService.fetchPolicies();
		return list;
	}

	@GetMapping("/policy/{PolicyId}")
	public ResponseEntity<Policies> getPolicyById(@PathVariable("policyId") int policyId) {
		Policies policy = policiesService.getPolicy(policyId);
		return ResponseEntity.ok().body(policy);
	}

	@PostMapping("/policy")
	public Policies addpolicy(@RequestBody Policies policies) {
		policiesService.savePolicy(policies);
		return policies;
	}

	@PutMapping("/policy/{id}")
	public ResponseEntity<Policies> updatePolicy(@PathVariable("id") int policyId,
			@RequestBody Policies policydategoryDetails) {
		Policies policy = policiesService.getPolicy(policyId);

		policy.setPolicyName(policydategoryDetails.getPolicyName());
		policy.setCategory(policydategoryDetails.getCategory());
		policy.setAmount(policydategoryDetails.getAmount());
		policy.setTenureInYears(policydategoryDetails.getTenureInYears());

		final Policies updatedPolicy = policiesService.savePolicy(policy);
		return ResponseEntity.ok(updatedPolicy);

	}


	@DeleteMapping(value = "/policy/{policyId}")

	public ResponseEntity<Object> deletePolicy(@PathVariable("policyId") int policyId) {

		policiesService.deletePolicy(policyId);

		return new ResponseEntity<>("policy deleted successsfully", HttpStatus.OK);

	}


	@GetMapping("/query")

	public List<Queries> getQueries() {

		List<Queries> list = queriesService.fetchQueries();


		return list;


	}


	@GetMapping("/query/{userName}")

	public ResponseEntity<Queries> getQueriesById(@PathVariable("userName") String
userName) {

		Queries queries = queriesService.getQuery(userName);

		return ResponseEntity.ok().body(queries);

	}


	@PostMapping("/query")

	public Queries addQueries(@RequestBody Queries queries) {

		queries = queriesService.saveQuery(queries);

		return queries;
	}

	@PutMapping("/query/{userName}")
	public ResponseEntity<Queries> updateQueries(@PathVariable("userName") String
userName,
				@RequestBody Queries queries) {
		Queries queries1 = queriesService.getQuery(userName);
		queries1.setQuestion(queries.getQuestion());
		queries1.setAnswer(queries.getAnswer());

		final Queries updatedQueries = queriesService.saveQuery(queries1);
		return ResponseEntity.ok(updatedQueries);
	}

	@DeleteMapping(value = "/query/{userName}")
	public ResponseEntity<Object> deleteQueries(@PathVariable("userName") String
userName) {
		queriesService.deleteQuery(userName);
		return new ResponseEntity<>("Query deleted successsfully", HttpStatus.OK);
	}

	@GetMapping("/users")
	public List<User> getUsers() {
		List<User> userList = userService.fetchUser();
		return userList;
	}

	@GetMapping("/user/{userId}")
	public ResponseEntity<User> getUserById(@PathVariable("userId") String userId) {

		User user = userService.getUser(userId);

		return ResponseEntity.ok().body(user);

	}


	@PostMapping("/user")

	public User addUser(@Validated @RequestBody User user) {

		user = userService.saveUser(user);

		return user;

	}


	@PutMapping("/user/{userId}")

	public ResponseEntity<User> updateUser(@PathVariable("userId") String userId,
@RequestBody User userDetails) {

		User user = userService.getUser(userId);

		user.setEmail(userDetails.getEmail());

		user.setMobile(userDetails.getMobile());

		user.setPassword(userDetails.getPassword());

		final User updatedUser = userService.updateUser(user);

		return ResponseEntity.ok(updatedUser);

	}


	@DeleteMapping(value = "/user/{userId}")

	public ResponseEntity<Object> deleteUser(@PathVariable("userId") String userId) {

		userService.deleteUser(userId);

		return new ResponseEntity<>("User deleted successsfully", HttpStatus.OK);

	}
	@PostMapping("/login")

	public ResponseEntity<AuthenticationStatus> validateLogin(@RequestBody Login login,
HttpServletRequest request) {


		AuthenticationStatus status = loginService.validateLogin(login.getUserName(),
login.getPassword());

		if (status.getUserName() != null && status.getPassword() != null &&
status.isAuthenticated() == true) {

			request.getSession().setAttribute("userName", status.getUserName());

		}

		return new ResponseEntity<AuthenticationStatus>(status, HttpStatus.OK);


	}


	@GetMapping("/logout")

	public void Logout(HttpServletRequest request) {

		System.out.println("logged out");

		request.getSession().invalidate();


	}


	@DeleteMapping("/approvals/{userName}")

	public void deleteApproval(@PathVariable("userName") String userName) {

		approvalsService.deleteApproval(userName);

	}

	@SuppressWarnings({ "unchecked", "null" })

	@GetMapping("/userpolicies")

	public List<Approved> getAllUserpolicies() {

		@SuppressWarnings("rawtypes")
```

```
                List list = null;

                list.add(approvedService.getAllApproved());

                list.add(disapprovedService.getAllDisapproved());

                return list;

        }

}
```

ResourceNOTException.java:-

```
package com.insurance.management.exception;

import org.springframework.http.HttpStatus;

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

@ResponseStatus(value = HttpStatus.NOT_FOUND)

public class ResourceNotFoundException extends Exception {

        private static final long serialVersionUID = 1L;


         public ResourceNotFoundException(String message){

            super(message);

         }

}
```

Admin.java:-

```
package com.insurance.management.model;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name="admin")
```

```
public class Admin {

        @Id

        @Column(name="aid")

        private String aid;

        @Column(name="username")

        private String username;

        @Column(name="password")

        private String password;


        public Admin() {


        }


        public String getAid() {

                return aid;

        }

        public void setAid(String aid) {

                this.aid = aid;

        }

        public Admin(String aid, String username, String password) {

                super();

                this.aid = aid;

                this.username = username;

                this.password = password;

        }

        public String getUsername(){

        return username;
```

```
	}
	public void setUsername(String username){
		this.username=username;
	}
	public String getPassword() {
		return password;
	}
	public void setPassword(String password) {
		this.password = password;
	}
}
```

Approvals.java-

```
package com.insurance.management.model;


import java.util.Date;


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


@Entity(name = "approvals")
public class Approvals {
	@Id
	@Column(name = "user_name")
	private String userName;
	@Column(name = "policy_id")

	private Integer policyId;


	@GeneratedValue(strategy = GenerationType.TABLE)

	@Column(name = "request_id")

	private Integer requestId;


	@Column(name = "date")

	private Date date;


	@Column(name = "status")

	private String status;

	public Approvals() {

	}

	public Approvals(String userName, Integer policyId, Integer requestId, Date date, String 
status) {

		super();

		this.userName = userName;

		this.policyId = policyId;

		this.requestId = requestId;

		this.date = date;

		this.status = status;

	}

	public String getUserName() {

		return userName;

	}

	public void setUserName(String userName) {
		this.userName = userName;

	}

	public Integer getPolicyId() {

		return policyId;

	}

	public void setPolicyId(Integer policyId) {

		this.policyId = policyId;

	}

	public Integer getRequestId() {

		return requestId;

	}

	public void setRequestId(Integer requestId) {

		this.requestId = requestId;

	}

	public Date getDate() {

		return date;

	}

	public void setDate(Date date) {

		this.date = date;

	}

	public String getStatus() {

		return status;

	}

	public void setStatus(String status) {

		this.status = status;

	}

}
```

Approved.java-

```
package com.insurance.management.model;


import java.util.Date;


import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;


@Entity(name = "approved")

public class Approved {

	@Id

	@Column(name = "user_name")

	private String userName;




	@Column(name = "policy_id")

	private Integer policyId;

	@GeneratedValue(strategy = GenerationType.AUTO)

	@Column(name = "request_id")

	private Integer requestId;


	@Column(name = "date")

	private Date date;


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

	public Approved() {

	}

	public Approved(String userName, Integer policyId, Integer requestId, Date date, String
status) {
		super();
		this.userName = userName;
		this.policyId = policyId;
		this.requestId = requestId;
		this.date = date;
		this.status = status;
	}

	public String getUserName() {
		return userName;
	}

	public void setUserName(String userName) {
		this.userName = userName;
	}

	public Integer getPolicyId() {
		return policyId;
	}
```

```
}

public void setPolicyId(Integer policyId) {
	this.policyId = policyId;
}

public Integer getRequestId() {
	return requestId;
}
public void setRequestId(Integer requestId) {
	this.requestId = requestId;
}

public Date getDate() {
	return date;
}

public void setDate(Date date) {
	this.date = date;
}
public String getStatus() {
	return status;
}
public void setStatus(String status) {
	this.status = status;
}
```

```
}
```

Authentication.java:-

```
package com.insurance.management.model;

public class AuthenticationStatus {
	private String userName;
	private String password;
	private boolean authenticated;

	public AuthenticationStatus() {

	}

	public AuthenticationStatus(String userName,
String password, boolean authenticated) {
		super();
		this.userName = userName;
		this.password = password;
		this.authenticated = authenticated;
	}

	public String getUserName() {
		return userName;
	}

	public void setUserName(String userName) {
		this.userName = userName;
	}

	public String getPassword() {
		return password;
	}

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

	public boolean isAuthenticated() {
		return authenticated;
	}
```

```
		public void setAuthenticated(boolean
authenticated) {
			this.authenticated = authenticated;
		}

	}
```

Category.java:-

```
package com.insurance.management.model;


import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.Table;


@Entity

@Table(name = "category")

public class Category {


	@Id

	@Column(name = "category")

	private String category;


	public Category() {

		super();


	}


	public Category(String category) {
        super();
        this.category = category;
    }
    public String getCategory() {
        return category;
    }
    public void setCategory(String category) {
        this.category = category;
    }
}
```

Disapproved.java:-

```
package com.insurance.management.model;


import java.util.Date;


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


@Entity(name = "disapproved")
public class Disapproved {
    @Id
    @Column(name = "user_name")
    private String userName;
```

```
	@Column(name = "policy_id")
	private Integer policyId;
	@GeneratedValue(strategy = GenerationType.AUTO)
	@Column(name = "request_id")
	private Integer requestId;

	@Column(name = "date")
	private Date date;

	@Column(name = "status")

	private String status;

	public Disapproved() {

	}

	public Disapproved(String userName, Integer policyId, Integer requestId, Date date, String
status) {
		super();
		this.userName = userName;
		this.policyId = policyId;
		this.requestId = requestId;
		this.date = date;
		this.status = status;
	}
```

```
public String getUserName() {

            return userName;

}


public void setUserName(String userName) {

            this.userName = userName;

}


public Integer getPolicyId() {

            return policyId;

}


public void setPolicyId(Integer policyId) {

            this.policyId = policyId;

}


public Integer getRequestId() {

            return requestId;

}


public void setRequestId(Integer requestId) {

            this.requestId = requestId;

}


public Date getDate() {

            return date;

}
```

```
	public void setDate(Date date) {

		this.date = date;

	}

	public String getStatus() {

		return status;

	}

	public void setStatus(String status) {

	this.status = status;

	}

}
```

Login.java:-

```
package com.insurance.management.model;



import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name="User")

public class Login {

	@Id

	@Column(name="user_name")

	private String userName;

	@Column(name="password")

	private String password;
	public Login() {


	}

	public String getUserName(){

		return userName;

	}

	public void setUserName(String userName){

		this.userName=userName;

	}

	public String getPassword() {

		return password;

	}


	public void setPassword(String password) {

		this.password = password;

	}

}
```

Policies.java:-

```
package com.insurance.management.model;



import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;
```

```
@Entity

@Table(name = "policies")

public class Policies {


	@Id

	@GeneratedValue(strategy = GenerationType.AUTO)

	@Column(name = "policy_id")

	private int policyId;


	@Column(name = "policy_name")

	private String policyName;


	@Column(name = "category")

	private String category;


	@Column(name = "amount")

	private Double amount;


	@Column(name = "tenure_in_years")

	private int tenureInYears;


	public Policies() {


	}


	public Policies(int policyId, String policyName, String category, Double amount, int
tenureInYears) {

		super();
```

```
        this.policyId = policyId;

        this.policyName = policyName;

        this.category = category;

        this.amount = amount;

        this.tenureInYears = tenureInYears;

}


public int getPolicyId() {

        return policyId;

}


public void setPolicyId(int policyId) {

        this.policyId = policyId;

}


public String getPolicyName() {

        return policyName;

}


public void setPolicyName(String policyName) {

        this.policyName = policyName;

}


public String getCategory() {

        return category;

}
```

```
	public void setCategory(String category) {

		this.category = category;

	}


	public Double getAmount() {

		return amount;

	}

	public void setAmount(Double amount) {

		this.amount = amount;

	}

	public int getTenureInYears() {

		return tenureInYears;

	}

	public void setTenureInYears(int tenureInYears) {

		this.tenureInYears = tenureInYears;

	}

}
```

Queries.java:-

```
package com.insurance.management.model;


import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.Table;


@Entity

@Table(name = "queries")
```

```
public class Queries {


	@Id

	@Column(name = "userName")

	private String userName;


	@Column(name = "query_id")

	private int queryId;


	@Column(name = "question")

	private String question;


	@Column(name = "answer")

	private String answer;


	public Queries() {


	}


	public Queries(String userName, int queryId, String question, String answer) {

		super();

		this.userName = userName;

		this.queryId = queryId;

		this.question = question;

		this.answer = answer;

	}
```

```
public String getUserName() {

		return userName;

}


public void setUserName(String userName) {

		this.userName = userName;

}


public int getQueryId() {

		return queryId;

}


public void setQueryId(int queryId) {

		this.queryId = queryId;

}


public String getQuestion() {

		return question;

}


public void setQuestion(String question) {

		this.question = question;

}


public String getAnswer() {

		return answer;

}
```

```
	public void setAnswer(String answer) {

	this.answer = answer;

	}

}
```

User.java:-

```
package com.insurance.management.model;


import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;


@Entity

@Table(name = "User")

public class User {


	@GeneratedValue(strategy = GenerationType.AUTO)

	@Column(name = "user_id")

	private int userid;

	@Id

	@Column(name = "user_name")

	private String userName;


	@Column(name = "email")
```

```
private String email;


@Column(name = "mobile")

private long mobile;


@Column(name = "password")

private String password;


public User() {


}


public User(int userid, String userName, String email, long mobile, String password) {

	super();

	this.userid = userid;

	this.userName = userName;

	this.email = email;

	this.mobile = mobile;

	this.password = password;

}


public int getUserid() {

	return userid;

}


public void setUserid(int 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 long getMobile() {
	return mobile;
}

public void setMobile(long mobile) {
	this.mobile = mobile;
}
public String getPassword() {
	return password;
```

```
        }

        public void setPassword(String password) {

                this.password = password;

        }

}
```

Admin.Repo.java:-

```
package com.insurance.management.repository;

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

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

import com.insurance.management.model.Admin;

public interface AdminRepository extends JpaRepository<Admin, String> {

        @Query("SELECT a FROM Admin a WHERE a.username =?1 and a.password=?2")

        public  Admin validateAdmin(String username,String password);

        }
```

Approve.Repo.java:-

```
package com.insurance.management.repository;

import org.springframework.data.jpa.repository.support.JpaRepositoryImplementation;

import org.springframework.stereotype.Repository;

import com.insurance.management.model.Approvals;

@Repository

public interface ApprovalsRepository extends JpaRepositoryImplementation<Approvals, String> {

}
```

Approved.repo.java:-

```
package com.insurance.management.repository;


import org.springframework.data.jpa.repository.support.JpaRepositoryImplementation;

import org.springframework.stereotype.Repository;
import com.insurance.management.model.Approved;

@Repository

public interface ApprovedRepository extends JpaRepositoryImplementation<Approved, String> {

}
```

Category.repo.java:-

```
package com.insurance.management.repository;

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

import org.springframework.stereotype.Repository;

import com.insurance.management.model.Category;

@Repository

public interface CategoryRepository extends JpaRepository<Category, String> {

}
```

Dissaproved.repo:-

```
package com.insurance.management.repository;

import org.springframework.data.jpa.repository.support.JpaRepositoryImplementation;

import org.springframework.stereotype.Repository;

import com.insurance.management.model.Disapproved;

@Repository

public interface DisapprovedRepository extends JpaRepositoryImplementation<Disapproved, String>
{

}
```

LoginRepo.java:-

```
package com.insurance.management.repository;

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

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

import com.insurance.management.model.Login;
```

```
public interface LoginRepository extends JpaRepository<Login, String>{

        @Query("SELECT dl FROM Login dl WHERE dl.userName =?1 and dl.password=?2")

        public Login validateLogin(String userName,String password);

}
```

PoliciesRepo.java:-

```
package com.insurance.management.repository;

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

import org.springframework.stereotype.Repository;

import com.insurance.management.model.Policies;

@Repository

public interface PoliciesRepository extends JpaRepository<Policies, Integer> {

}
```

Queries.repo.java:-

```
package com.insurance.management.repository;

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

import org.springframework.stereotype.Repository;

import com.insurance.management.model.Queries;

@Repository

public interface QueriesRepository extends JpaRepository<Queries, String> {

}
```

UserRepo.java:-

```
package com.insurance.management.repository;

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

import org.springframework.stereotype.Repository;

import com.insurance.management.model.User;
```

```
@Repository

public interface UserRepository extends JpaRepository<User, String> {

}
```

AdminService.java:-

```
package com.insurance.management.service;

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

import org.springframework.stereotype.Service;


import com.insurance.management.repository.AdminRepository;

import com.insurance.management.model.Admin;

import com.insurance.management.model.AuthenticationStatus;

@Service

public class AdminService {


	@Autowired

	AdminRepository adminRepository;

	public AuthenticationStatus validateAdminLogin(String username, String password) {

		AuthenticationStatus status = null;

		Admin admin = adminRepository.validateAdmin(username, password);

		if(admin!=null) {

			status = new AuthenticationStatus(admin.getUsername(),
admin.getPassword(), true);

		}

		else {

			status = new AuthenticationStatus(null, null, false);				);

		}

		return status;

	}	}
```

ApprovalsService.java:-

```
package com.insurance.management.service;


import java.util.Date;

import java.util.List;

import java.util.Optional;


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

import org.springframework.stereotype.Service;


import com.insurance.management.model.Approvals;

import com.insurance.management.repository.ApprovalsRepository;




@Service

public class ApprovalsService {

	@Autowired

	ApprovalsRepository approvalsRepository;


	public List<Approvals> getAllApprovals() {

		return approvalsRepository.findAll();

	}


	public Optional<Approvals> getApprovals(String userName) {

		return approvalsRepository.findById(userName);

	}
	public void addApproval(Approvals approvals) {

		Date date = new Date();

		 approvals.setDate(date);

		 String status="pending";

		approvals.setStatus(status);

		approvalsRepository.save(approvals);

	}

	public void deleteApproval(String userName) {

		approvalsRepository.deleteById(userName);

	}

}
```

ApprovedService.java:-

```
package com.insurance.management.service;


import java.util.Date;

import java.util.List;

import java.util.Optional;


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

import org.springframework.stereotype.Service;


import com.insurance.management.model.Approved;

import com.insurance.management.repository.ApprovedRepository;



@Service

public class ApprovedService {
```

```
            @Autowired

            ApprovedRepository approvedRepository;


            public List<Approved> getAllApproved() {

                        return approvedRepository.findAll();

            }

            public Optional<Approved> getApproved(String userName) {

                        return approvedRepository.findById(userName);

            }

            public String addApproved(Approved approved) {

        Date date=new Date();

        String status="approved";

                         approved.setDate(date);

                         approved.setStatus(status);

                        approvedRepository.save(approved);

                        return "added";

            }

}
```

Category.service.java:-

```
package com.insurance.management.service;


import java.util.List;

import java.util.Optional;


import javax.transaction.Transactional;


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


import com.insurance.management.model.Category;
import com.insurance.management.repository.CategoryRepository;


@Service
public class CategoryService {
	@Autowired
	CategoryRepository categoryRepository;


	@Transactional
	public List<Category> fetchCategory() {
		List<Category> list = categoryRepository.findAll();
		return list;


	}


	@Transactional
	public Category SaveCategory(Category category) {


		return categoryRepository.save(category);


	}


	@Transactional
	public String deleteCategory(String category) {
		categoryRepository.deleteById(category);
		return "deleted";
	}
	@Transactional
	public Category getCategory(String Category) {
		Optional<Category> optional = categoryRepository.findById(Category);
		Category insuranceCategory = optional.get();
		return insuranceCategory;
	}
}
```

Dissapproved.service.java:-

```
package com.insurance.management.service;


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


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


import com.insurance.management.model.Disapproved;
import com.insurance.management.repository.DisapprovedRepository;



@Service
public class DisapprovedService {
	@Autowired
	DisapprovedRepository disapprovedRepository;
	public List<Disapproved> getAllDisapproved() {

		return disapprovedRepository.findAll();

	}


	public Optional<Disapproved> getDisapproved(String userName) {

		return disapprovedRepository.findById(userName);

	}


	public String addDisapproved(Disapproved disapproved) {

	   Date date=new Date();

		String status="disapproved";

		disapproved.setDate(date);

		disapproved.setStatus(status);

		disapprovedRepository.save(disapproved);

		return "added";

	}

}
```

Login.service.java:-

```
package com.insurance.management.service;

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

import org.springframework.stereotype.Service;

import com.insurance.management.repository.LoginRepository;

import com.insurance.management.model.AuthenticationStatus;

import com.insurance.management.model.Login;

@Service

public class LoginService {
	@Autowired

	LoginRepository loginRepository;

	public AuthenticationStatus validateLogin(String userName, String password) {

		AuthenticationStatus status = null;

		Login login = loginRepository.validateLogin(userName, password);

		if(login!=null) {

			status = new AuthenticationStatus(login.getUserName(),
login.getPassword(), true);

		}

		else {

			status = new AuthenticationStatus(null, null, false);

		}

		return status;

	}

}
```

PoliciesService.java:-

```
package com.insurance.management.service;



import java.util.List;

import java.util.Optional;


import javax.transaction.Transactional;


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

import org.springframework.stereotype.Service;
import com.insurance.management.model.Policies;

import com.insurance.management.repository.PoliciesRepository;


@Service

public class PoliciesService {

	@Autowired

	PoliciesRepository policiesRepository;


	@Transactional

	public List<Policies> fetchPolicies() {

		List<Policies> list=policiesRepository.findAll();

		return list;


	}

	@Transactional

	public Policies savePolicy(Policies policies) {


		return policiesRepository.save(policies);


	}

	@Transactional

	public void updatePolicy(Policies policies) {

		policiesRepository.save(policies);


	}


	@Transactional
```

```
            public void deletePolicy(int policyId) {
            policiesRepository.deleteById(policyId);
            }
            @Transactional
              public Policies getPolicy(int policyId) {
              Optional<Policies> optional=policiesRepository.findById(policyId);
              Policies policies=optional.get();
              return policies;
}
}
```

QueriesService.java:-

```
package com.insurance.management.service;
import java.util.List;
import java.util.Optional;

import javax.transaction.Transactional;

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

import com.insurance.management.model.Queries;
import com.insurance.management.repository.QueriesRepository;

@Service
public class QueriesService {
            @Autowired
            QueriesRepository queriesRepository;
```

```
@Transactional

public List<Queries> fetchQueries() {

        List<Queries> queriesList=queriesRepository.findAll();

        return queriesList;


}

@Transactional

public Queries saveQuery(Queries queries) {


        return queriesRepository.save(queries);


}

@Transactional

public void updateQuery(Queries queries) {

        queriesRepository.save(queries);


}


@Transactional

public void deleteQuery(String userName) {

        queriesRepository.deleteById(userName);


}

@Transactional

  public Queries getQuery(String userName) {

  Optional<Queries> optional= queriesRepository.findById(userName);
		Queries queries=optional.get();

		return queries;

		}

}
```

UserService.java:-

```
package com.insurance.management.service;


import java.util.List;

import java.util.Optional;


import javax.transaction.Transactional;


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

import org.springframework.stereotype.Service;


import com.insurance.management.model.User;

import com.insurance.management.repository.UserRepository;


@Service

public class UserService {


	@Autowired

	UserRepository userRepository;


	@Transactional

	public List<User> fetchUser() {

		List<User> userList = userRepository.findAll();
		return userList;

	}

	@Transactional
	public User saveUser(User user) {

		return userRepository.save(user);

	}

	@Transactional
	public User updateUser(User user) {
		userRepository.save(user);
		return user;

	}

	@Transactional
	public void deleteUser(String userName) {
		userRepository.deleteById(userName);

	}
	@Transactional
	public User getUser(String userName) {
		Optional<User> optional = userRepository.findById(userName);
		User user = optional.get();
```

```
                return user;

        }

}
```

pom.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.insurance</groupId>
	<artifactId>management</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>management</name>
	<description>Demo project for Spring Boot</description>
	<properties>
		<java.version>11</java.version>
	</properties>
	<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter</artifactId>
		</dependency>
		<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.springframework.boot</groupId>
			<artifactId>spring-boot-devtools</artifactId>
			<scope>runtime</scope>
			<optional>true</optional>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-jdbc</artifactId>
		</dependency>
```

```xml
            <dependency>
                  <groupId>org.springframework.session</groupId>
                  <artifactId>spring-session-core</artifactId>
            </dependency>
            <dependency>
                  <groupId>org.springframework.session</groupId>
                  <artifactId>spring-session-jdbc</artifactId>
            </dependency>
            <dependency>
                  <groupId>mysql</groupId>
                  <artifactId>mysql-connector-java</artifactId>
                  <scope>runtime</scope>
            </dependency>
            <dependency>
                  <groupId>junit</groupId>
                  <artifactId>junit</artifactId>
                  <scope>test</scope>
            </dependency>
            <dependency>
                  <groupId>org.springframework.boot</groupId>
                  <artifactId>spring-boot-starter-test</artifactId>
                  <scope>test</scope>
            </dependency>
            <dependency>
                  <groupId>org.springframework.boot</groupId>
                  <artifactId>spring-boot-starter-tomcat</artifactId>
                  <scope>provided</scope>
            </dependency>
      </dependencies>
      <build>
            <plugins>
                  <plugin>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-maven-plugin</artifactId>
                  </plugin>
            </plugins>
      </build>
</project>
```