Online Quiz Portal Using REST API’s

Three rest API :

1) Auth service
2) Admin service
3) Quiz service

1) Auth-service

application.properties

```
spring.application.name=auth-service
server.port = 9950
spring.datasource.url= jdbc:mysql://localhost:3306/quiz
spring.datasource.driverClassName=com.mysql.cj.jdbc.Driver
spring.datasource.username=root
spring.datasource.password=Ponmalai11
spring.mvc.pathmatch.matching-strategy=ant_path_matcher
```

Main method - AuthServiceApplication

```
package com.learn.quiz.authService;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication
@ComponentScan("com.learn.quiz.*")
public class AuthServiceApplication {

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

}
```

SpringConfig

```
package com.learn.quiz.authService;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

import springfox.documentation.builders.PathSelectors;
import springfox.documentation.builders.RequestHandlerSelectors;
import springfox.documentation.spi.DocumentationType;
import springfox.documentation.spring.web.plugins.Docket;
import springfox.documentation.swagger2.annotations.EnableSwagger2;

@Configuration
@EnableSwagger2
public class SpringFoxConfig {
	@Bean
	public Docket api() {
		return new Docket(DocumentationType.SWAGGER_2).select()

	.apis(RequestHandlerSelectors.basePackage("com.learn.quiz.controller
")).paths(PathSelectors.any())
					.build();
	}
}
```

Entity - User.java

```
package com.learn.quiz.entity;

import java.util.Date;

public class User {

	private Long id;

	private String firstName;

	private String lastName;

	private String emailId;

	private String password;

	private String mobileNumber;

	private Character isAdmin = 'N';

	private Date createdOn;

	private Date updatedOn;

	public User() {
		super();
	}

	public User(Long id, String firstName, String lastName, String
emailId, String password, String mobileNumber,
			Character isAdmin, Date createdOn, Date updatedOn) {
		super();
		this.id = id;
		this.firstName = firstName;
		this.lastName = lastName;
		this.emailId = emailId;
		this.password = password;
		this.mobileNumber = mobileNumber;
		this.isAdmin = isAdmin;
		this.createdOn = createdOn;
		this.updatedOn = updatedOn;
	}

	public Long getId() {
		return id;
	}

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

	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 getEmailId() {
		return emailId;
	}

	public void setEmailId(String emailId) {
		this.emailId = emailId;
	}

	public String getPassword() {
		return password;
	}

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

	public String getMobileNumber() {
		return mobileNumber;
	}

	public void setMobileNumber(String mobileNumber) {
		this.mobileNumber = mobileNumber;
	}

	public Character getIsAdmin() {
		return isAdmin;
	}

	public void setIsAdmin(Character isAdmin) {
		this.isAdmin = isAdmin;
	}

	public Date getCreatedOn() {
		return createdOn;
	}

	public void setCreatedOn(Date createdOn) {
		this.createdOn = createdOn;
	}
```

```java
	public Date getUpdatedOn() {
		return updatedOn;
	}

	public void setUpdatedOn(Date updatedOn) {
		this.updatedOn = updatedOn;
	}

}
```

Entity - LoginSession.java

```java
package com.learn.quiz.entity;

import java.util.Date;

public class LoginSession {

	private Long id;

	private Long userId;

	private String accessToken;

	private Date lastAccess;

	public LoginSession() {
	}

	public LoginSession(Long id, Long userId, String accessToken, Date 
lastAccess) {
		super();
		this.id = id;
		this.userId = userId;
		this.accessToken = accessToken;
		this.lastAccess = lastAccess;
	}

	public Long getId() {
		return id;
	}

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

	public Long getUserId() {
		return userId;
	}

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

	public String getAccessToken() {
		return accessToken;
	}

	public void setAccessToken(String accessToken) {
		this.accessToken = accessToken;
	}

	public Date getLastAccess() {
		return lastAccess;
	}

	public void setLastAccess(Date lastAccess) {
		this.lastAccess = lastAccess;
	}

}
```

DTO - LoginDto

```java
package com.learn.quiz.dto;

public class LoginDto {

	private String userName;
	private String password;

	public LoginDto(String userName, String password) {
		super();
		this.userName = userName;
		this.password = password;
	}

	public LoginDto() {
		super();
	}

	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;
	}
```

DTO - CreateUserDto

```java
package com.learn.quiz.dto;

public class CreateUserDto {

	private String firstName;
	private String lastName;
	private String emailId;
	private String mobileNumber;
	private String password;

	public CreateUserDto() {
		super();
	}
	public CreateUserDto(String firstName, String lastName, String
emailId, String mobileNumber, String password) {
		super();
		this.firstName = firstName;
		this.lastName = lastName;
		this.emailId = emailId;
		this.mobileNumber = mobileNumber;
		this.password = password;
	}

	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 getEmailId() {
		return emailId;
	}

	public void setEmailId(String emailId) {
		this.emailId = emailId;
	}

	public String getMobileNumber() {
		return mobileNumber;
	}

	public void setMobileNumber(String mobileNumber) {
		this.mobileNumber = mobileNumber;
	}

	public String getPassword() {
		return password;
	}

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

}
```

DTO - ChangePasswordDto

```
package com.learn.quiz.dto;

public class ChangePasswordDto {
	private String currentPassword;
	private String newPassword;
	private String retypePassword;

	public ChangePasswordDto() {
		super();
	}

	public ChangePasswordDto(String currentPassword, String newPassword, 
String retypePassword) {
		super();
		this.currentPassword = currentPassword;
		this.newPassword = newPassword;
		this.retypePassword = retypePassword;
	}

	public String getCurrentPassword() {
		return currentPassword;
	}

	public void setCurrentPassword(String currentPassword) {
		this.currentPassword = currentPassword;
	}

	public String getNewPassword() {
		return newPassword;
	}

	public void setNewPassword(String newPassword) {
		this.newPassword = newPassword;
	}

	public String getRetypePassword() {
		return retypePassword;
	}

	public void setRetypePassword(String retypePassword) {
		this.retypePassword = retypePassword;
	}
}
```
DTO - Response
```
package com.learn.quiz.dto;

public class Response {

	private String message;

	public Response(String message) {
		super();
		this.message = message;
	}

	public String getMessage() {
		return message;
	}

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

}
```

DTO - LoginResponse

```
package com.learn.quiz.dto;

public class LoginResponse {

	private String accessToken;

	public LoginResponse() {
		super();
	}

	public LoginResponse(String accessToken) {
		super();
		this.accessToken = accessToken;
	}

	public String getAccessToken() {
		return accessToken;
	}

	public void setAccessToken(String accessToken) {
		this.accessToken = accessToken;
	}

}
```

Repository - UserDao.java

```
package com.learn.quiz.repository;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.entity.User;

@Repository
public class UserDao {

	@Autowired
	JdbcTemplate jdbcTemplate;

      public User getByEmailIdAndPassword(String userName, String password)
{
		List<User> rs = jdbcTemplate.query("SELECT u.* from user u
where u.email_id = ? and u.password = ?",
					new BeanPropertyRowMapper<User>(User.class), new
Object[] { userName, password });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

	public User findById(Long id) {
		List<User> rs = jdbcTemplate.query("SELECT u.* from user u 
where u.id = ?",
				new BeanPropertyRowMapper<User>(User.class), new 
Object[] { id });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

	public int save(User user) {
		return jdbcTemplate.update(
				"INSERT INTO `quiz`.`user` ( `first_name`, 
`last_name`, `email_id`, `password`, `mobile_number`) "
						+ " VALUES ( ?, ?, ?, ?, ?)",
				new Object[] { user.getFirstName(), 
user.getLastName(), user.getEmailId(), user.getPassword(),
						user.getMobileNumber() });
	}

	public int updatePassword(Long id, String newPassword) {
		return jdbcTemplate.update("UPDATE `quiz`.`user` SET 
`password` = ?, `updated_on` = now() WHERE `id` = ?",
				new Object[] { newPassword, id });
	}

}
```

Repository - LoginSessionDao.java

```
package com.learn.quiz.repository;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.entity.LoginSession;

@Repository
public class LoginSessionDao {

	@Autowired
	JdbcTemplate jdbcTemplate;

	public Integer deleteByAccessToken(String accessToken) {
		return jdbcTemplate.update("delete from login_session ls where 
ls.access_token = ?",
				new Object[] { accessToken });
	}

	public LoginSession findByAccessToken(String accessToken) {
		List<LoginSession> rs = jdbcTemplate.query("SELECT ls.* from 
login_session ls where ls.access_token = ?",
					new 
BeanPropertyRowMapper<LoginSession>(LoginSession.class), new Object[] 
{ accessToken });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

	public LoginSession getByUserId(Long userId) {
		List<LoginSession> rs = jdbcTemplate.query("SELECT ls.* from 
login_session ls where ls.user_id = ?",
					new 
BeanPropertyRowMapper<LoginSession>(LoginSession.class), new Object[] 
{ userId });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

	public LoginSession save(LoginSession loginSession) {
		jdbcTemplate.update("INSERT INTO `quiz`.`login_session` 
(`user_id`, `access_token`) VALUES (?, ?)",
					new Object[] { loginSession.getUserId(), 
loginSession.getAccessToken() });
		return getByUserId(loginSession.getUserId());
	}

}
```

Exception - CustomException

```
package com.learn.quiz.exceptionHandler;

import org.springframework.http.HttpStatus;

public class CustomException extends RuntimeException {

	/**
	 * 
	 */
	private static final long serialVersionUID = -6472915215894290670L;

	private final HttpStatus httpStatus;

	public CustomException(String message, HttpStatus httpStatus) {
		super(message);
		this.httpStatus = httpStatus;
	}

	public HttpStatus getHttpStatus() {
		return httpStatus;
	}
}
```

Exception - CustomExceptionHandler

```
package com.learn.quiz.exceptionHandler;

import java.io.IOException;

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

import com.learn.quiz.dto.Response;

@RestControllerAdvice
public class CustomExceptionHandler {

	@ExceptionHandler(IOException.class)
	public ResponseEntity<Response> ioExceptionHandler(IOException ex,
WebRequest request) {
		ex.printStackTrace();
		return new ResponseEntity<Response>(new Response("Unable to
access."), HttpStatus.BAD_REQUEST);
	}

	@ExceptionHandler(CustomException.class)
	public ResponseEntity<Response>
customExceptionHandler(CustomException ex, WebRequest request) {
		ex.printStackTrace();
		return new ResponseEntity<Response>(new
Response(ex.getLocalizedMessage()), ex.getHttpStatus());
	}

	@ExceptionHandler(MissingRequestHeaderException.class)
	public ResponseEntity<Response>
ioExceptionHandler(MissingRequestHeaderException ex, WebRequest request) {
		ex.printStackTrace();
		return new ResponseEntity<Response>(new
Response(ex.getLocalizedMessage()), HttpStatus.BAD_REQUEST);
	}
}
```

Service - UserService

```
package com.learn.quiz.service;

import java.util.Date;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.learn.quiz.dto.ChangePasswordDto;
import com.learn.quiz.dto.CreateUserDto;
import com.learn.quiz.dto.LoginDto;
import com.learn.quiz.dto.LoginResponse;
import com.learn.quiz.dto.Response;
import com.learn.quiz.entity.LoginSession;
import com.learn.quiz.entity.User;
import com.learn.quiz.exceptionHandler.CustomException;
import com.learn.quiz.repository.LoginSessionDao;
import com.learn.quiz.repository.UserDao;
import com.learn.quiz.utility.AES;

@Service
public class UserService {

	@Autowired
	private UserDao userDao;

	@Autowired
	private LoginSessionDao loginSessionDao;

	private User getUser(String authorization) {
		LoginSession loginSession =
loginSessionDao.findByAccessToken(authorization.split(" ")[1]);
		if (loginSession == null) {
			throw new CustomException("Invalid access.",
HttpStatus.UNAUTHORIZED);
		}
		User user = userDao.findById(loginSession.getUserId());
		if (user != null) {
			return user;
		} else {
			throw new CustomException("Invalid access.",
HttpStatus.UNAUTHORIZED);
		}
	}

	@Transactional(readOnly = false)
	public ResponseEntity<LoginResponse> authenticate(LoginDto loginDto)
throws CustomException {
		User user =
userDao.getByEmailIdAndPassword(loginDto.getUserName(),
loginDto.getPassword());
		if (user != null) {
			String accessToken = AES.encrypt(user.getId() +
user.getEmailId());
			LoginSession loginSession =
loginSessionDao.getByUserId(user.getId());
			if (loginSession == null) {
				loginSession = new LoginSession();
				loginSession.setUserId(user.getId());
			}
			loginSession.setAccessToken(accessToken);
			loginSession.setLastAccess(new Date());
			loginSessionDao.save(loginSession);
			return new ResponseEntity<LoginResponse>(new
LoginResponse(accessToken), HttpStatus.OK);
		} else {
			throw new CustomException("Invalid username or
password.", HttpStatus.BAD_REQUEST);
```

```
		}
	}

	public ResponseEntity<Response> changePassword(String authorization,
ChangePasswordDto changePasswordDto) {
		User user = getUser(authorization);
		if
(!changePasswordDto.getNewPassword().equals(changePasswordDto.getRetypePass
word())) {
			return new ResponseEntity<Response>(new Response("New
Password and Retype Password should be same."),
					HttpStatus.BAD_REQUEST);
		} else {
			if (changePasswordDto.getCurrentPassword() == null) {
				return new ResponseEntity<Response>(new
Response("User not exists, Please login again."),
						HttpStatus.BAD_REQUEST);
			} else if
(!user.getPassword().equals(changePasswordDto.getCurrentPassword())) {
				return new ResponseEntity<Response>(new
Response("Invalid Password."), HttpStatus.BAD_REQUEST);
			} else {
				if (userDao.updatePassword(user.getId(),
changePasswordDto.getNewPassword()) > 0) {
					return new ResponseEntity<Response>(
							new Response("Password has
changed successfully, You need to login again."), HttpStatus.OK);
				} else {
					return new ResponseEntity<Response>(new
Response("Internal Server Error."),

	HttpStatus.INTERNAL_SERVER_ERROR);
				}
			}
		}
	}

	public ResponseEntity<Response> createUser(CreateUserDto
createUserDto) {
		User user = new User(null, createUserDto.getFirstName(),
createUserDto.getLastName(),
				createUserDto.getEmailId(),
createUserDto.getPassword(), createUserDto.getMobileNumber(), 'N',
				new Date(), new Date());
		if (userDao.save(user) > 0) {
			return new ResponseEntity<Response>(new Response("User
created successfully."), HttpStatus.OK);
		} else {
			return new ResponseEntity<Response>(new
Response("Unable to create user."),
					HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}
}
```

Service - LoginSessionService

```
package com.learn.quiz.service;
```

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

import com.learn.quiz.dto.Response;
import com.learn.quiz.repository.LoginSessionDao;

@Service
public class LoginSessionService {

	@Autowired
	private LoginSessionDao loginSessionDao;

	public ResponseEntity<Response> logout(String authToken) {
		if (loginSessionDao.deleteByAccessToken(authToken.split("
")[1]) >= 1) {
			return new ResponseEntity<Response>(new
Response("Logout successfully"), HttpStatus.OK);
		}
		return new ResponseEntity<Response>(new Response("Logout
error"), HttpStatus.INTERNAL_SERVER_ERROR);
	}

}
```

Utility - AES.java

```
package com.learn.quiz.utility;

import java.nio.charset.StandardCharsets;
import java.security.spec.KeySpec;
import java.util.Base64;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.PBEKeySpec;
import javax.crypto.spec.SecretKeySpec;

public class AES {

	private static final String SECRET_KEY =
"my_super_secret_key_ho_ho_ho";

	private static final String SALT = "ssshhhhhhhhhhh!!!!";

	public static String encrypt(String strToEncrypt) {
		try {
			// Create default byte array
			byte[] iv = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0 };
			IvParameterSpec ivspec = new IvParameterSpec(iv);

			// Create SecretKeyFactory object
			SecretKeyFactory factory =
SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");

			// Create KeySpec object and assign with
			// constructor
```

```
                    KeySpec spec = new PBEKeySpec(SECRET_KEY.toCharArray(),
SALT.getBytes(), 65536, 256);
                    SecretKey tmp = factory.generateSecret(spec);
                    SecretKeySpec secretKey = new
SecretKeySpec(tmp.getEncoded(), "AES");

                    Cipher cipher =
Cipher.getInstance("AES/CBC/PKCS5Padding");
                    cipher.init(Cipher.ENCRYPT_MODE, secretKey, ivspec);
                    // Return encrypted string
                    return
Base64.getEncoder().encodeToString(cipher.doFinal(strToEncrypt.getBytes(Sta
ndardCharsets.UTF_8)));
             } catch (Exception e) {
                    System.out.println("Error while encrypting: " +
e.toString());
             }
             return null;
       }

       // This method use to decrypt to string
       public static String decrypt(String strToDecrypt) {
             try {

                    // Default byte array
                    byte[] iv = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0 };
                    // Create IvParameterSpec object and assign with
                    // constructor
                    IvParameterSpec ivspec = new IvParameterSpec(iv);

                    // Create SecretKeyFactory Object
                    SecretKeyFactory factory =
SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");

                    // Create KeySpec object and assign with
                    // constructor
                    KeySpec spec = new PBEKeySpec(SECRET_KEY.toCharArray(),
SALT.getBytes(), 65536, 256);
                    SecretKey tmp = factory.generateSecret(spec);
                    SecretKeySpec secretKey = new
SecretKeySpec(tmp.getEncoded(), "AES");

                    Cipher cipher =
Cipher.getInstance("AES/CBC/PKCS5PADDING");
                    cipher.init(Cipher.DECRYPT_MODE, secretKey, ivspec);
                    // Return decrypted string
                    return new
String(cipher.doFinal(Base64.getDecoder().decode(strToDecrypt)));
             } catch (Exception e) {
                    System.out.println("Error while decrypting: " +
e.toString());
             }
             return null;
       }
}
```

Controller - UserController

```
package com.learn.quiz.controller;
```

```java
import org.springframework.beans.factory.annotation.Autowired;
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.RequestHeader;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.learn.quiz.dto.ChangePasswordDto;
import com.learn.quiz.dto.CreateUserDto;
import com.learn.quiz.dto.LoginDto;
import com.learn.quiz.dto.LoginResponse;
import com.learn.quiz.dto.Response;
import com.learn.quiz.service.LoginSessionService;
import com.learn.quiz.service.UserService;

@RestController
@RequestMapping("/api/v1")
public class UserController {

	@Autowired
	private UserService userService;

	@Autowired
	private LoginSessionService loginSessionService;

	@PostMapping("/login")
	public ResponseEntity<LoginResponse> login(@RequestBody LoginDto
loginDto) {
		return userService.authenticate(loginDto);
	}

	@PostMapping("/logout")
	public ResponseEntity<Response>
logout(@RequestHeader("Authorization") String authorization) {
		return loginSessionService.logout(authorization);
	}

	@PostMapping("/change-password")
	public ResponseEntity<Response>
changePassword(@RequestHeader("Authorization") String authorization,
			@RequestBody ChangePasswordDto changePasswordDto) {
		return userService.changePassword(authorization,
changePasswordDto);
	}

	@PostMapping("/create-user")
	public ResponseEntity<Response> createUser(@RequestBody
CreateUserDto createUserDto) {
		return userService.createUser(createUserDto);
	}
}
```

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"
```

```xml
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>2.5.6</version>
		<relativePath /> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.learn</groupId>
	<artifactId>auth-service</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>auth-service</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-data-jdbc</artifactId>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-web</artifactId>
		</dependency>
		<dependency>
			<groupId>io.springfox</groupId>
			<artifactId>springfox-boot-starter</artifactId>
			<version>3.0.0</version>
		</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-test</artifactId>
			<scope>test</scope>
		</dependency>

		<dependency>
			<groupId>mysql</groupId>
			<artifactId>mysql-connector-java</artifactId>
			<version>8.0.29</version>
		</dependency>

	</dependencies>

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

</project>
```

2) Admin-service

application.properties

```
spring.application.name=admin-service
server.port = 9951
spring.datasource.url= jdbc:mysql://localhost:3306/quiz
spring.datasource.driverClassName=com.mysql.cj.jdbc.Driver
spring.datasource.username=root
spring.datasource.password=Ponmalai11
```

Main method - AdminServiceApplication

```
package com.learn.quiz.adminService;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication
@ComponentScan("com.learn.quiz.*")
public class AdminServiceApplication {

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

}
```

SpringConfig

```
package com.learn.quiz.adminService;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

import springfox.documentation.builders.PathSelectors;
import springfox.documentation.builders.RequestHandlerSelectors;
import springfox.documentation.spi.DocumentationType;
import springfox.documentation.spring.web.plugins.Docket;
import springfox.documentation.swagger2.annotations.EnableSwagger2;

@Configuration
@EnableSwagger2
public class SpringFoxConfig {
	@Bean
	public Docket api() {
		return new
Docket(DocumentationType.SWAGGER_2).select().apis(RequestHandlerSelectors.b
asePackage("com.learn.quiz.controller"))
```

```
                                .paths(PathSelectors.any()).build();
        }
}
```

Entity - User.java

```
package com.learn.quiz.entity;

import java.util.Date;

import com.fasterxml.jackson.annotation.JsonIgnore;

public class User {

        private Long id;

        private String firstName;

        private String lastName;

        private String emailId;

        @JsonIgnore
        private String password;

        private String mobileNumber;

        @JsonIgnore
        private Character isAdmin;

        @JsonIgnore
        private Date createdOn;

        @JsonIgnore
        private Date updatedOn;

        public User() {
                super();
        }

        public User(Long id, String firstName, String lastName, String
emailId, String password, String mobileNumber,
                        Character isAdmin, Date createdOn, Date updatedOn) {
                super();
                this.id = id;
                this.firstName = firstName;
                this.lastName = lastName;
                this.emailId = emailId;
                this.password = password;
                this.mobileNumber = mobileNumber;
                this.isAdmin = isAdmin;
                this.createdOn = createdOn;
                this.updatedOn = updatedOn;
        }

        public Long getId() {
                return id;
        }
```

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

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 getEmailId() {
	return emailId;
}

public void setEmailId(String emailId) {
	this.emailId = emailId;
}

public String getPassword() {
	return password;
}

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

public String getMobileNumber() {
	return mobileNumber;
}

public void setMobileNumber(String mobileNumber) {
	this.mobileNumber = mobileNumber;
}

public Character getIsAdmin() {
	return isAdmin;
}

public void setIsAdmin(Character isAdmin) {
	this.isAdmin = isAdmin;
}

public Date getCreatedOn() {
	return createdOn;
}

public void setCreatedOn(Date createdOn) {
	this.createdOn = createdOn;
}
```

```
	public Date getUpdatedOn() {
		return updatedOn;
	}

	public void setUpdatedOn(Date updatedOn) {
		this.updatedOn = updatedOn;
	}

}
```

Entity - Quiz.java

```
package com.learn.quiz.entity;

import java.util.Date;

import com.fasterxml.jackson.annotation.JsonIgnore;

public class Quiz {

	private Long id;

	private String title;

	private String category;

	@JsonIgnore
	private Character deleted = 'N';

	@JsonIgnore
	private Date createdOn = new Date();

	@JsonIgnore
	private Date updatedOn = new Date();

	public Quiz() {
		super();
	}

	public Quiz(Long id, String title, String category, Character
deleted, Date createdOn, Date updatedOn) {
		super();
		this.id = id;
		this.title = title;
		this.category = category;
		this.deleted = deleted;
		this.createdOn = createdOn;
		this.updatedOn = updatedOn;
	}

	public Long getId() {
		return id;
	}

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

	public String getTitle() {
```

```java
		return title;
	}

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

	public String getCategory() {
		return category;
	}

	public void setCategory(String category) {
		this.category = category;
	}

	public Character getDeleted() {
		return deleted;
	}

	public void setDeleted(Character deleted) {
		this.deleted = deleted;
	}

	public Date getCreatedOn() {
		return createdOn;
	}

	public void setCreatedOn(Date createdOn) {
		this.createdOn = createdOn;
	}

	public Date getUpdatedOn() {
		return updatedOn;
	}

	public void setUpdatedOn(Date updatedOn) {
		this.updatedOn = updatedOn;
	}

}
```

Entity - Question.java

```java
package com.learn.quiz.entity;

import java.util.Date;

import com.fasterxml.jackson.annotation.JsonIgnore;

public class Question {

	private Long id;

	private String question;

	private String optionA;

	private String optionB;
	private String optionC;

	private String optionD;

	private Character rightOption;

	@JsonIgnore
	private Character deleted = 'N';

	@JsonIgnore
	private Date createdOn = new Date();

	@JsonIgnore
	private Date updatedOn = new Date();

	public Question() {
		super();
		// TODO Auto-generated constructor stub
	}

	public Question(Long id, String question, String optionA, String
optionB, String optionC, String optionD,
			Character rightOption, Character deleted, Date
createdOn, Date updatedOn) {
		super();
		this.id = id;
		this.question = question;
		this.optionA = optionA;
		this.optionB = optionB;
		this.optionC = optionC;
		this.optionD = optionD;
		this.rightOption = rightOption;
		this.deleted = deleted;
		this.createdOn = createdOn;
		this.updatedOn = updatedOn;
	}

	public Long getId() {
		return id;
	}

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

	public String getQuestion() {
		return question;
	}

	public void setQuestion(String question) {
		this.question = question;
	}

	public String getOptionA() {
		return optionA;
	}

	public void setOptionA(String optionA) {
		this.optionA = optionA;
	}

	public String getOptionB() {
		return optionB;
	}

	public void setOptionB(String optionB) {
		this.optionB = optionB;
	}

	public String getOptionC() {
		return optionC;
	}

	public void setOptionC(String optionC) {
		this.optionC = optionC;
	}

	public String getOptionD() {
		return optionD;
	}

	public void setOptionD(String optionD) {
		this.optionD = optionD;
	}

	public Character getRightOption() {
		return rightOption;
	}

	public void setRightOption(Character rightOption) {
		this.rightOption = rightOption;
	}

	public Character getDeleted() {
		return deleted;
	}

	public void setDeleted(Character deleted) {
		this.deleted = deleted;
	}

	public Date getCreatedOn() {
		return createdOn;
	}

	public void setCreatedOn(Date createdOn) {
		this.createdOn = createdOn;
	}

	public Date getUpdatedOn() {
		return updatedOn;
	}

	public void setUpdatedOn(Date updatedOn) {
		this.updatedOn = updatedOn;
	}

}
```

Entity - QuizQuestion.java

```
package com.learn.quiz.entity;

import java.util.Date;

import com.fasterxml.jackson.annotation.JsonIgnore;

public class QuizQuestion {

	private Long id;

	private Long quizId;

	private Long questionId;

	@JsonIgnore
	private Character deleted = 'N';

	@JsonIgnore
	private Date createdOn = new Date();

	@JsonIgnore
	private Date updatedOn = new Date();

	public QuizQuestion(Long id, Long quizId, Long questionId, Character 
deleted, Date createdOn, Date updatedOn) {
		super();
		this.id = id;
		this.quizId = quizId;
		this.questionId = questionId;
		this.deleted = deleted;
		this.createdOn = createdOn;
		this.updatedOn = updatedOn;
	}

	public QuizQuestion() {
		super();
	}

	public Long getId() {
		return id;
	}

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

	public Long getQuizId() {
		return quizId;
	}

	public void setQuizId(Long quizId) {
		this.quizId = quizId;
	}

	public Long getQuestionId() {
		return questionId;
```

```java
	}

	public void setQuestionId(Long questionId) {
		this.questionId = questionId;
	}

	public Character getDeleted() {
		return deleted;
	}

	public void setDeleted(Character deleted) {
		this.deleted = deleted;
	}

	public Date getCreatedOn() {
		return createdOn;
	}

	public void setCreatedOn(Date createdOn) {
		this.createdOn = createdOn;
	}

	public Date getUpdatedOn() {
		return updatedOn;
	}

	public void setUpdatedOn(Date updatedOn) {
		this.updatedOn = updatedOn;
	}

}
```

Entity - LoginSession.java

```java
package com.learn.quiz.entity;

import java.util.Date;

public class LoginSession {

	private Long id;

	private Long userId;

	private String accessToken;

	private Date lastAccess;

	public LoginSession() {
	}

	public LoginSession(Long id, Long userId, String accessToken, Date 
lastAccess) {
		super();
		this.id = id;
		this.userId = userId;
		this.accessToken = accessToken;
		this.lastAccess = lastAccess;
	}
	public Long getId() {
		return id;
	}

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

	public Long getUserId() {
		return userId;
	}

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

	public String getAccessToken() {
		return accessToken;
	}

	public void setAccessToken(String accessToken) {
		this.accessToken = accessToken;
	}

	public Date getLastAccess() {
		return lastAccess;
	}

	public void setLastAccess(Date lastAccess) {
		this.lastAccess = lastAccess;
	}

}
```

DTO - QuestionDto

```java
package com.learn.quiz.dto;

public class QuestionDto {

	private String question;
	private String optionA;
	private String optionB;
	private String optionC;
	private String optionD;
	private Character rightOption;

	public QuestionDto(String question, String optionA, String optionB,
String optionC, String optionD,
			String rightOption) {
		super();
		this.question = question;
		this.optionA = optionA;
		this.optionB = optionB;
		this.optionC = optionC;
		this.optionD = optionD;
		if (rightOption != null && rightOption.length() > 0)
			this.rightOption = rightOption.charAt(0);
	}

	public QuestionDto() {
		super();
	}

	public String getQuestion() {
		return question;
	}

	public void setQuestion(String question) {
		this.question = question;
	}

	public String getOptionA() {
		return optionA;
	}

	public void setOptionA(String optionA) {
		this.optionA = optionA;
	}

	public String getOptionB() {
		return optionB;
	}

	public void setOptionB(String optionB) {
		this.optionB = optionB;
	}

	public String getOptionC() {
		return optionC;
	}

	public void setOptionC(String optionC) {
		this.optionC = optionC;
	}

	public String getOptionD() {
		return optionD;
	}

	public void setOptionD(String optionD) {
		this.optionD = optionD;
	}

	public Character getRightOption() {
		return rightOption;
	}

	public void setRightOption(Character rightOption) {
		this.rightOption = rightOption;
	}

}
```

DTO - QuestionRequestDto

```
package com.learn.quiz.dto;

public class QuestionRequestDto {

	private String question;

	private String optionA;

	private String optionB;

	private String optionC;

	private String optionD;

	private Character rightOption;

	public QuestionRequestDto() {
		super();
	}

	public QuestionRequestDto(String question, String optionA, String
optionB, String optionC, String optionD,
			Character rightOption) {
		super();
		this.question = question;
		this.optionA = optionA;
		this.optionB = optionB;
		this.optionC = optionC;
		this.optionD = optionD;
		this.rightOption = rightOption;
	}

	public String getQuestion() {
		return question;
	}

	public void setQuestion(String question) {
		this.question = question;
	}

	public String getOptionA() {
		return optionA;
	}

	public void setOptionA(String optionA) {
		this.optionA = optionA;
	}

	public String getOptionB() {
		return optionB;
	}

	public void setOptionB(String optionB) {
		this.optionB = optionB;
	}

	public String getOptionC() {
		return optionC;
	}

	public void setOptionC(String optionC) {
		this.optionC = optionC;
	}

	public String getOptionD() {
		return optionD;
	}

	public void setOptionD(String optionD) {
		this.optionD = optionD;
	}

	public Character getRightOption() {
		return rightOption;
	}

	public void setRightOption(Character rightOption) {
		this.rightOption = rightOption;
	}

}
```

DTO - QuizQuestionRequestDto

```
package com.learn.quiz.dto;

public class QuizQuestionRequestDto {

	private Long quizId;

	private Long questionId;

	public QuizQuestionRequestDto(Long quizId, Long questionId) {
		super();
		this.quizId = quizId;
		this.questionId = questionId;
	}

	public QuizQuestionRequestDto() {
		super();
	}

	public Long getQuizId() {
		return quizId;
	}

	public void setQuizId(Long quizId) {
		this.quizId = quizId;
	}

	public Long getQuestionId() {
		return questionId;
	}

	public void setQuestionId(Long questionId) {
		this.questionId = questionId;
	}

}
```

DTO - QuizRequestDto

```
package com.learn.quiz.dto;

public class QuizRequestDto {

	private String title;

	private String category;

	public QuizRequestDto() {
		super();
	}

	public QuizRequestDto(String title, String category) {
		super();
		this.title = title;
		this.category = category;
	}

	public String getTitle() {
		return title;
	}

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

	public String getCategory() {
		return category;
	}

	public void setCategory(String category) {
		this.category = category;
	}
}
```

DTO - ResponseDto

```
package com.learn.quiz.dto;

public class Response {

	private String message;

	public Response(String message) {
		super();
		this.message = message;
	}

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

}
```

DTO - UpdateQuestionRequestDto

```
package com.learn.quiz.dto;

public class UpdateQuestionRequestDto extends QuestionRequestDto {

	private Long id;

	public UpdateQuestionRequestDto() {
		super();
	}

	public UpdateQuestionRequestDto(String question, String optionA, 
String optionB, String optionC, String optionD,
			Character rightOption, Long id) {
		super(question, optionA, optionB, optionC, optionD, 
rightOption);
		this.id = id;
	}

	public Long getId() {
		return id;
	}

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

}
```

DTO - UpdateQuizRequestDto

```
package com.learn.quiz.dto;

public class UpdateQuizRequestDto extends QuizRequestDto {

	private Long id;

	public UpdateQuizRequestDto() {
		super();
	}

	public UpdateQuizRequestDto(Long id, String title, String category) 
{
		super(title, category);
		this.id = id;
	}

	public Long getId() {
		return id;
	}

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

}
```

DTO - UpdateUserDto

```
package com.learn.quiz.dto;

public class UpdateUserDto {

	private String firstName;
	private String lastName;
	private String mobileNumber;

	public UpdateUserDto() {
		super();
	}

	public UpdateUserDto(String firstName, String lastName, String
mobileNumber) {
		super();
		this.firstName = firstName;
		this.lastName = lastName;
		this.mobileNumber = mobileNumber;
	}

	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 getMobileNumber() {
		return mobileNumber;
	}

	public void setMobileNumber(String mobileNumber) {
		this.mobileNumber = mobileNumber;
	}

}
```

Repository - LoginSessionDao.java

```
package com.learn.quiz.repository;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.entity.LoginSession;

@Repository
public class LoginSessionDao {

	@Autowired
	JdbcTemplate jdbcTemplate;

	public LoginSession findByAccessToken(String accessToken) {
		System.out.println("accessToken:"+accessToken);
		List<LoginSession> rs = jdbcTemplate.query("SELECT ls.* from 
login_session ls where ls.access_token = ?",
					new 
BeanPropertyRowMapper<LoginSession>(LoginSession.class), new Object[] 
{ accessToken });
		LoginSession loginSession = null;
		for(int i=0;i<rs.size();i++)
		{
			System.out.print(i);
			System.out.print(" id:"+rs.get(i).getId());
			System.out.print(" userId:"+rs.get(i).getUserId());
			System.out.print(" 
accessToken:"+rs.get(i).getAccessToken());
			System.out.println(" 
lastAccess:"+rs.get(i).getLastAccess());
			if(i==0)
			{
				loginSession = new LoginSession();
				
	loginSession.setAccessToken(rs.get(i).getAccessToken());
				loginSession.setId(rs.get(i).getId());
				loginSession.setUserId(rs.get(i).getUserId());;
				
	loginSession.setLastAccess(rs.get(i).getLastAccess());
			}
		}

		//if (rs != null && rs.size() > 0) {
			//loginSession=rs.get(0);
			return loginSession;
		//} else {
		//	return null;
		//}
	}

}
```

Repository - QuestionDao.java

```
package com.learn.quiz.repository;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.entity.Question;

@Repository
public class QuestionDao {

	@Autowired
	JdbcTemplate jdbcTemplate;

	public List<Question> findAllByDeleted(Character deleted) {
		return jdbcTemplate.query("SELECT q.* from question q where
q.deleted = 'N'",
				new
BeanPropertyRowMapper<Question>(Question.class));
	}

	public Question findById(Long questionId) {
		List<Question> rs = jdbcTemplate.query("SELECT q.* from
question q where q.id = ? and q.deleted = 'N'",
				new
BeanPropertyRowMapper<Question>(Question.class), new Object[]
{ questionId });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

	public int save(Question question) {
		return jdbcTemplate.update(
				"INSERT INTO quiz.question "
						+ "( question, option_a, option_b,
option_c, option_d, right_option, deleted)"
						+ " VALUES ( ?, ?, ?, ?, ?, ?,
'N' );",
				new Object[] { question.getQuestion(),
question.getOptionA(), question.getOptionB(),
						question.getOptionC(),
question.getOptionD(), question.getRightOption().toString() });
	}

	public int update(Question question) {
		return jdbcTemplate.update(
				"UPDATE quiz.question SET question = ?, option_a
= ?, option_b = ?, option_c = ?, option_d = ?, "
						+ "right_option = ?, deleted = 'N',
updated_on = now() WHERE id = ? ",
				new Object[] { question.getQuestion(),
question.getOptionA(), question.getOptionB(),
						question.getOptionC(),
question.getOptionD(), question.getRightOption().toString(),
						question.getId() });
	}

	public int delete(Long questionId) {
		return jdbcTemplate.update("UPDATE quiz.question SET deleted = 
'Y', updated_on = now() WHERE id = ? ",
				new Object[] { questionId });
	}

}
```

Repository - QuizDao.java

```
package com.learn.quiz.repository;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.entity.Quiz;

@Repository
public class QuizDao {

	@Autowired
	JdbcTemplate jdbcTemplate;

	public List<Quiz> findAllByDeleted(Character deleted) {
		return jdbcTemplate.query("SELECT q.* from quiz q where 
q.deleted = 'N'",
				new BeanPropertyRowMapper<Quiz>(Quiz.class));
	}

	public Quiz findById(Long quizId) {
		List<Quiz> rs = jdbcTemplate.query("SELECT q.* from quiz q 
where q.deleted = 'N' and q.id = ?",
				new BeanPropertyRowMapper<Quiz>(Quiz.class), new 
Object[] { quizId });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

	public int save(Quiz quiz) {
		if (quiz.getId() == null) {
			return persist(quiz);
		} else {
			return merge(quiz);
		}
	}

	private int merge(Quiz quiz) {
		return jdbcTemplate.update(
				"UPDATE `quiz`.`quiz` SET `title` =     ?, 
`category` =?, `deleted` = ?, `updated_on` = now() WHERE `id` = ?",
				new Object[] { quiz.getTitle(), 
quiz.getCategory(), quiz.getDeleted().toString(), quiz.getId() });
	}
	private int persist(Quiz quiz) {
		return jdbcTemplate.update("INSERT INTO `quiz`.`quiz`
( `title`, `category`, `deleted`) VALUES (?, ?, 'N' )",
				new Object[] { quiz.getTitle(),
quiz.getCategory() });
	}

}
```

Repository - QuizQuestionDao.java

```
package com.learn.quiz.repository;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.entity.QuizQuestion;

@Repository
public class QuizQuestionDao {

	@Autowired
	JdbcTemplate jdbcTemplate;

	public List<QuizQuestion> findByQuizId(Long quizId) {
		return jdbcTemplate.query("SELECT q.* from quiz_question q
where q.quiz_id = ? and q.deleted = 'N'",
				new
BeanPropertyRowMapper<QuizQuestion>(QuizQuestion.class), new Object[]
{ quizId });
	}

	public int save(QuizQuestion quizQuestion) {
		return jdbcTemplate.update(
				"INSERT INTO `quiz`.`quiz_question` ( `quiz_id`,
`question_id`, `deleted`) VALUES (?, ?, 'N' )",
				new Object[] { quizQuestion.getQuizId(),
quizQuestion.getQuestionId() });
	}

	public QuizQuestion findById(Long quizQuestionId) {
		List<QuizQuestion> rs = jdbcTemplate.query("SELECT q.* from
quiz_question q where q.id = ? and q.deleted = 'N'",
				new
BeanPropertyRowMapper<QuizQuestion>(QuizQuestion.class), new Object[]
{ quizQuestionId });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

	public int delete(Long quizQuestionId) {
		return jdbcTemplate.update(
				"UPDATE `quiz`.`quiz_question` SET `deleted` =
'Y', `updated_on` = now() WHERE `id` = ?",
				new Object[] { quizQuestionId });
	}

}
```

Repository - UserDao.java

```
package com.learn.quiz.repository;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.entity.User;

@Repository
public class UserDao {

	@Autowired
	JdbcTemplate jdbcTemplate;

	public User findById(Long id) {
		List<User> rs = jdbcTemplate.query("SELECT u.* from user u
where u.id = ?",
				new BeanPropertyRowMapper<User>(User.class), new
Object[] { id });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

	public int update(User user) {
		return jdbcTemplate.update(
				"UPDATE `quiz`.`user` SET `first_name` = ?,
`last_name` = ?, `mobile_number` = ?, `updated_on` = now()"
						+ " WHERE `id` = ?;",
				new Object[] { user.getFirstName(),
user.getLastName(), user.getMobileNumber(), user.getId() });
	}

}
```

Exception - CustomException

```
package com.learn.quiz.exceptionHandler;

import org.springframework.http.HttpStatus;

public class CustomException extends RuntimeException {
	/**
	 * 
	 */
	private static final long serialVersionUID = -6971346490685985372L;

	private final HttpStatus httpStatus;

	public CustomException(String message, HttpStatus httpStatus) {
		super(message);
		this.httpStatus = httpStatus;
	}

	public HttpStatus getHttpStatus() {
		return httpStatus;
	}

}
```

Service - QuestionService

```
package com.learn.quiz.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Service;

import com.learn.quiz.dto.QuestionRequestDto;
import com.learn.quiz.dto.Response;
import com.learn.quiz.dto.UpdateQuestionRequestDto;
import com.learn.quiz.entity.Question;
import com.learn.quiz.exceptionHandler.CustomException;
import com.learn.quiz.repository.QuestionDao;

@Service
public class QuestionService {

	@Autowired
	private QuestionDao questionDao;

	@Autowired
	private UserService userService;

	public List<Question> findAll(String authorization) {
		userService.getUser(authorization);
		return questionDao.findAllByDeleted('N');
	}

	public Question findById(Long questionId, String authorization) {
		userService.getUser(authorization);
		Question question = questionDao.findById(questionId);
		if (question == null) {
			throw new CustomException("Question not found.", 
HttpStatus.BAD_REQUEST);
		} else {
			return question;
		}
	}

	public ResponseEntity<Response> deleteQuestion(Long questionId,
String authorization) {

		if (questionDao.delete(questionId) > 0) {
			return new ResponseEntity<Response>(new
Response("Question deleted successfully."), HttpStatus.OK);
		} else {
			return new ResponseEntity<Response>(new
Response("Question not able to delete."),
					HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}

	public ResponseEntity<Response>
updateQuestion(UpdateQuestionRequestDto updateQuestionRequestDto,
			String authorization) {
		Question question = findById(updateQuestionRequestDto.getId(),
authorization);
		question.setQuestion(updateQuestionRequestDto.getQuestion());
		question.setOptionA(updateQuestionRequestDto.getOptionA());
		question.setOptionB(updateQuestionRequestDto.getOptionB());
		question.setOptionC(updateQuestionRequestDto.getOptionC());
		question.setOptionD(updateQuestionRequestDto.getOptionD());

	question.setRightOption(updateQuestionRequestDto.getRightOption());
		if (questionDao.update(question) > 0) {
			return new ResponseEntity<Response>(new
Response("Question updated successfully."), HttpStatus.OK);
		} else {
			return new ResponseEntity<Response>(new
Response("Question not able to update."),
					HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}

	public ResponseEntity<Response> createQuestion(QuestionRequestDto
questionRequestDto, String authorization) {
		Question question = new Question();
		question.setQuestion(questionRequestDto.getQuestion());
		question.setOptionA(questionRequestDto.getOptionA());
		question.setOptionB(questionRequestDto.getOptionB());
		question.setOptionC(questionRequestDto.getOptionC());
		question.setOptionD(questionRequestDto.getOptionD());
		question.setRightOption(questionRequestDto.getRightOption());
		if (questionDao.save(question) > 0) {
			return new ResponseEntity<Response>(new
Response("Question created successfully."), HttpStatus.OK);
		} else {
			return new ResponseEntity<Response>(new
Response("Question not able to create."),
					HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}

}
```

Service - QuizQuestionService

```
package com.learn.quiz.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Service;

import com.learn.quiz.dto.QuizQuestionRequestDto;
import com.learn.quiz.dto.Response;
import com.learn.quiz.entity.QuizQuestion;
import com.learn.quiz.exceptionHandler.CustomException;
import com.learn.quiz.repository.QuizQuestionDao;

@Service
public class QuizQuestionService {

	@Autowired
	private QuizQuestionDao questionDao;

	@Autowired
	private UserService userService;

	public QuizQuestion findById(Long quizQuestionId, String 
authorization) {
		userService.getUser(authorization);
		QuizQuestion qq = questionDao.findById(quizQuestionId);
		if (qq == null)
			throw new CustomException("QuizQuestion not found.", 
HttpStatus.BAD_REQUEST);
		else
			return qq;
	}

	public List<QuizQuestion> findByQuizId(Long quizId, String 
authorization) {
		userService.getUser(authorization);
		return questionDao.findByQuizId(quizId);
	}

	public ResponseEntity<Response> deleteQuizQuestion(Long 
quizQuestionId, String authorization) {
		userService.getUser(authorization);
		if (questionDao.delete(quizQuestionId) > 0) {
			return new ResponseEntity<Response>(new 
Response("QuizQuestion deleted successfully."), HttpStatus.OK);
		} else {
			return new ResponseEntity<Response>(new 
Response("QuizQuestion not able to delete."),
					HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}

	public ResponseEntity<Response> 
addQuizQuestion(QuizQuestionRequestDto quizQuestionRequestDto,
			String authorization) {
```

```
		userService.getUser(authorization);
		QuizQuestion quizQuestion = new QuizQuestion();
		quizQuestion.setQuizId(quizQuestionRequestDto.getQuizId());
		
	quizQuestion.setQuestionId(quizQuestionRequestDto.getQuestionId());
		if (questionDao.save(quizQuestion) > 0) {
			return new ResponseEntity<Response>(new 
Response("Question added successfully."), HttpStatus.OK);
		} else {
			return new ResponseEntity<Response>(new 
Response("Question not able to add."),
					HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}

}
```

Service - QuizService

```
package com.learn.quiz.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Service;

import com.learn.quiz.dto.QuizRequestDto;
import com.learn.quiz.dto.Response;
import com.learn.quiz.dto.UpdateQuizRequestDto;
import com.learn.quiz.entity.Quiz;
import com.learn.quiz.exceptionHandler.CustomException;
import com.learn.quiz.repository.QuizDao;

@Service
public class QuizService {

	@Autowired
	private QuizDao quizDao;

	@Autowired
	private UserService userService;

	public List<Quiz> findAll(String authorization) {
		userService.getUser(authorization);
		return quizDao.findAllByDeleted('N');
	}

	public Quiz findById(Long quizId, String authorization) {
		userService.getUser(authorization);
		Quiz quiz = quizDao.findById(quizId);
		if (quiz == null)
			throw new CustomException("Quiz not found.", 
HttpStatus.BAD_REQUEST);
		else
			return quiz;
	}
	public ResponseEntity<Response> deleteQuiz(Long quizId, String 
authorization) {
		Quiz quiz = findById(quizId, authorization);
		quiz.setDeleted('Y');
		if (quizDao.save(quiz) > 0) {
			return new ResponseEntity<Response>(new Response("Quiz 
deleted successfully."), HttpStatus.OK);
		} else {
			return new ResponseEntity<Response>(new Response("Quiz 
not able to delete."),
					HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}

	public ResponseEntity<Response> updateQuiz(UpdateQuizRequestDto 
updateQuizRequestDto, String authorization) {
		Quiz quiz = findById(updateQuizRequestDto.getId(), 
authorization);
		quiz.setTitle(updateQuizRequestDto.getTitle());
		quiz.setCategory(updateQuizRequestDto.getCategory());
		if (quizDao.save(quiz) > 0) {
			return new ResponseEntity<Response>(new Response("Quiz 
updated successfully."), HttpStatus.OK);
		} else {
			return new ResponseEntity<Response>(new Response("Quiz 
not able to update."),
					HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}

	public ResponseEntity<Response> createQuiz(QuizRequestDto 
quizRequestDto, String authorization) {
		userService.getUser(authorization);
		Quiz quiz = new Quiz();
		quiz.setTitle(quizRequestDto.getTitle());
		quiz.setCategory(quizRequestDto.getCategory());
		if (quizDao.save(quiz) > 0) {
			return new ResponseEntity<Response>(new Response("Quiz 
created successfully."), HttpStatus.OK);
		} else {
			return new ResponseEntity<Response>(new Response("Quiz 
not able to create."),
					HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}

}
```

Service - UserService

```
package com.learn.quiz.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Service;

import com.learn.quiz.dto.Response;
import com.learn.quiz.dto.UpdateUserDto;
```

```java
import com.learn.quiz.entity.LoginSession;
import com.learn.quiz.entity.User;
import com.learn.quiz.exceptionHandler.CustomException;
import com.learn.quiz.repository.LoginSessionDao;
import com.learn.quiz.repository.UserDao;

@Service
public class UserService {

	@Autowired
	private UserDao userDao;

	@Autowired
	private LoginSessionDao loginSessionDao;

	public User getUser(String authorization) {
		System.out.println("authorization:"+authorization);
		System.out.println("authorization.split(\"
\")[0]:"+authorization.split(" ")[0]);
		LoginSession loginSession =
loginSessionDao.findByAccessToken(authorization.split(" ")[0]);
		if (loginSession == null) {
			throw new CustomException("Invalid access
loginSession.", HttpStatus.UNAUTHORIZED);
		}
		User user = userDao.findById(loginSession.getUserId());
		if (user == null || user.getIsAdmin() == 'N') {
			throw new CustomException("Invalid access user id.",
HttpStatus.UNAUTHORIZED);
		}
		return user;
	}

	public ResponseEntity<Response> updateProfile(UpdateUserDto
updateUserDto, String authorization) {
		User user = getUser(authorization);
		user.setFirstName(updateUserDto.getFirstName());
		user.setLastName(updateUserDto.getFirstName());
		user.setMobileNumber(updateUserDto.getMobileNumber());
		if (userDao.update(user) > 0) {
			return new ResponseEntity<Response>(new
Response("Profile updated successfully."), HttpStatus.OK);
		} else {
			return new ResponseEntity<Response>(new
Response("Unable to update profile."),
					HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}

}
```

Utility - ExcelReader.java

```java
package com.learn.quiz.utility;

import java.io.IOException;
import java.io.InputStream;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.CellType;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import com.learn.quiz.dto.QuestionDto;

public class ExcelReader {
	public static List<QuestionDto> getQuestion(InputStream inputStream)
{
		List<QuestionDto> questionSets = new ArrayList<QuestionDto>();
		try (XSSFWorkbook wb = new XSSFWorkbook(inputStream)) {
			XSSFSheet sheet = wb.getSheetAt(0);
			Iterator<Row> itr = sheet.iterator();
			if (itr.hasNext()) {
				itr.next();
			}
			while (itr.hasNext()) {
				Row row = itr.next();
				String question = null;
				String optionA = null;
				String optionB = null;
				String optionC = null;
				String optionD = null;
				String correctOption = null;
				int i = 0;
				Iterator<Cell> cellIterator = row.cellIterator();
				while (cellIterator.hasNext()) {
					Cell cell = cellIterator.next();
					cell.~~setCellType~~(CellType.STRING);
					switch (i) {
					case 0:
					      question = cell.getStringCellValue();
						break;
					case 1:
						optionA = cell.getStringCellValue();
						break;
					case 2:
						optionB = cell.getStringCellValue();
						break;
					case 3:
						optionC = cell.getStringCellValue();
						break;
					case 4:
						optionD = cell.getStringCellValue();
						break;
					case 5:
						correctOption =
cell.getStringCellValue();
						break;
					default:
					}
					i++;
				}
				questionSets.add(new QuestionDto(question,
optionA, optionB, optionC, optionD, correctOption));
```

```
			}
			return questionSets;
		} catch (IOException e) {
			e.printStackTrace();
		} finally {
			if (inputStream != null) {
				try {
					inputStream.close();
				} catch (IOException e) {
					e.printStackTrace();
				}
			}
		}
		return null;
	}
}
```

Controller - CustomExceptionHandler

```
package com.learn.quiz.controller;

import java.io.IOException;

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

import com.learn.quiz.dto.Response;
import com.learn.quiz.exceptionHandler.CustomException;

@RestControllerAdvice
public class CustomExceptionHandler {

	@ExceptionHandler(IOException.class)
	public ResponseEntity<Response> ioExceptionHandler(IOException ex,
WebRequest request) {
		ex.printStackTrace();
		return new ResponseEntity<Response>(new Response("Unable to
access."), HttpStatus.BAD_REQUEST);
	}

	@ExceptionHandler(CustomException.class)
	public ResponseEntity<Response>
customExceptionHandler(CustomException ex, WebRequest request) {
		ex.printStackTrace();
		return new ResponseEntity<Response>(new
Response(ex.getLocalizedMessage()), ex.getHttpStatus());
	}
}
```

Controller - QuestionController

```
package com.learn.quiz.controller;
```

```java
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
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.RequestHeader;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.learn.quiz.dto.QuestionRequestDto;
import com.learn.quiz.dto.Response;
import com.learn.quiz.dto.UpdateQuestionRequestDto;
import com.learn.quiz.entity.Question;
import com.learn.quiz.service.QuestionService;

@RestController
@RequestMapping("/api/v1")
public class QuestionController {

	@Autowired
	private QuestionService questionService;

	@GetMapping("/getAllQuestion")
	public List<Question> getAllQuestion(@RequestHeader("Authorization")
String authorization) {
		return questionService.findAll(authorization);
	}

	@GetMapping("/get/question/{questionId}")
	public Question getQuestion(@PathVariable Long questionId,
@RequestHeader("Authorization") String authorization) {
		return questionService.findById(questionId, authorization);
	}

	@GetMapping("/delete/question/{questionId}")
	public ResponseEntity<Response> deleteQuestion(@PathVariable Long
questionId,
			@RequestHeader("Authorization") String authorization) {
		return questionService.deleteQuestion(questionId,
authorization);
	}

	@PostMapping("/update/question")
	public ResponseEntity<Response> updateQuestion(@RequestBody
UpdateQuestionRequestDto updateQuestionRequestDto,
			@RequestHeader("Authorization") String authorization) {
		return  questionService.updateQuestion(updateQuestionRequestDto,
authorization);
	}

	@PostMapping("/add/question")
	public ResponseEntity<Response> addQuestion(@RequestBody
QuestionRequestDto questionRequestDto,
			@RequestHeader("Authorization") String authorization) {
```

```
		return questionService.createQuestion(questionRequestDto,
authorization);
	}

}
```

Controller - QuizController

```
package com.learn.quiz.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
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.RequestHeader;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.learn.quiz.dto.QuizRequestDto;
import com.learn.quiz.dto.Response;
import com.learn.quiz.dto.UpdateQuizRequestDto;
import com.learn.quiz.entity.Quiz;
import com.learn.quiz.service.QuizService;

@RestController
@RequestMapping("/api/v1")
public class QuizController {

	@Autowired
	private QuizService quizService;

	@GetMapping("/getAllQuiz")
	public List<Quiz> getAllQuiz(@RequestHeader("Authorization") String
authorization) {
		return quizService.findAll(authorization);
	}

	@GetMapping("/get/quiz/{quizId}")
	public Quiz getQuiz(@PathVariable Long quizId,
@RequestHeader("Authorization") String authorization) {
		return quizService.findById(quizId, authorization);
	}

	@GetMapping("/delete/quiz/{quizId}")
      public ResponseEntity<Response> deleteQuiz(@PathVariable Long quizId,
@RequestHeader("Authorization") String authorization) {
		return quizService.deleteQuiz(quizId, authorization);
	}

	@PostMapping("/update/quiz")
	public ResponseEntity<Response> updateQuiz(@RequestBody
UpdateQuizRequestDto updateQuizRequestDto, @RequestHeader("Authorization")
String authorization) {
		return quizService.updateQuiz(updateQuizRequestDto,
authorization);
```

```
	}

	@PostMapping("/create/quiz")
	public ResponseEntity<Response> createQuiz(@RequestBody
QuizRequestDto quizRequestDto, @RequestHeader("Authorization") String
authorization) {
		return quizService.createQuiz(quizRequestDto, authorization);
	}

}
```

Controller - QuizQuestionController

```
package com.learn.quiz.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
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.RequestHeader;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.learn.quiz.dto.QuizQuestionRequestDto;
import com.learn.quiz.dto.Response;
import com.learn.quiz.entity.QuizQuestion;
import com.learn.quiz.service.QuizQuestionService;

@RestController
@RequestMapping("/api/v1")
public class QuizQuestionController {

	@Autowired
	private QuizQuestionService quizQuestionService;

	@GetMapping("/get/quiz/question/{quizId}")
	public List<QuizQuestion> getQuizQuestion(@PathVariable Long quizId,
			@RequestHeader("Authorization") String authorization) {
		return  quizQuestionService.findByQuizId(quizId,  authorization);
	}

	@GetMapping("/delete/quiz/question/{quizQuestionId}")
	public ResponseEntity<Response> deleteQuizQuestion(@PathVariable
Long quizQuestionId,
			@RequestHeader("Authorization") String authorization) {
		return quizQuestionService.deleteQuizQuestion(quizQuestionId,
authorization);
	}

	@PostMapping("/add/quiz/question")
	public ResponseEntity<Response> addQuizQuestion(@RequestBody
QuizQuestionRequestDto quizQuestionRequestDto,
			@RequestHeader("Authorization") String authorization) {
		return
quizQuestionService.addQuizQuestion(quizQuestionRequestDto, authorization);
```

```
	}

}
```

Controller - UserController

```java
package com.learn.quiz.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
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.RequestHeader;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.learn.quiz.dto.Response;
import com.learn.quiz.dto.UpdateUserDto;
import com.learn.quiz.entity.User;
import com.learn.quiz.service.UserService;

@RestController
@RequestMapping("/api/v1")
public class UserController {

	@Autowired
	private UserService userService;

	@PostMapping("/update-profile")
	public ResponseEntity<Response> createUser(@RequestBody
UpdateUserDto updateUserDto,
				@RequestHeader("Authorization") String authorization) {
		return userService.updateProfile(updateUserDto, authorization);
	}

	@GetMapping("/get/profile")
	public User getUser(@RequestHeader("Authorization") String
authorization) {
		 return userService.getUser(authorization);
	}
}
```

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>2.5.6</version>
		<relativePath /> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.learn.quiz</groupId>
	<artifactId>admin-service</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>admin-service</name>
	<description>Demo project for Spring Boot</description>
	<properties>
		<java.version>17</java.version>
	</properties>
	<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-jdbc</artifactId>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-web</artifactId>
		</dependency>
		<dependency>
			<groupId>io.springfox</groupId>
			<artifactId>springfox-boot-starter</artifactId>
			<version>3.0.0</version>
		</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-test</artifactId>
			<scope>test</scope>
		</dependency>

		<dependency>
			<groupId>mysql</groupId>
			<artifactId>mysql-connector-java</artifactId>
			<version>8.0.29</version>
		</dependency>

		<!-- https://mvnrepository.com/artifact/org.dom4j/dom4j -->
		<dependency>
			<groupId>org.dom4j</groupId>
			<artifactId>dom4j</artifactId>
			<version>2.1.3</version>
		</dependency>
		<!-- https://mvnrepository.com/artifact/org.apache.poi/poi -->
		<dependency>
			<groupId>org.apache.poi</groupId>
			<artifactId>poi</artifactId>
			<version>5.2.2</version>
		</dependency>
		<!-- https://mvnrepository.com/artifact/org.apache.poi/poi-
ooxml-schemas -->
		<dependency>
			<groupId>org.apache.poi</groupId>
			<artifactId>poi-ooxml-schemas</artifactId>
			<version>4.1.2</version>
		</dependency>
		<!-- https://mvnrepository.com/artifact/org.apache.poi/poi-
ooxml -->
```

```
		<dependency>
			<groupId>org.apache.poi</groupId>
			<artifactId>poi-ooxml</artifactId>
			<version>5.2.2</version>
		</dependency>

	</dependencies>

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

</project>
```

3) Quiz-service

Application.properties

```
spring.application.name=quiz-service
server.port = 9952
spring.datasource.url= jdbc:mysql://localhost:3306/quiz
spring.datasource.driverClassName=com.mysql.cj.jdbc.Driver
spring.datasource.username=root
spring.datasource.password=Ponmalai11
```

Mainmethod - QuizServiceApplication

```
package com.learn.quiz.quizService;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication
@ComponentScan("com.learn.quiz.*")
public class QuizServiceApplication {

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

}
```

SpringConfig

```
package com.learn.quiz.quizService;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

import springfox.documentation.builders.PathSelectors;
```

```java
import springfox.documentation.builders.RequestHandlerSelectors;
import springfox.documentation.spi.DocumentationType;
import springfox.documentation.spring.web.plugins.Docket;
import springfox.documentation.swagger2.annotations.EnableSwagger2;

@Configuration
@EnableSwagger2
public class SpringFoxConfig {
	@Bean
	public Docket api() {
		return new Docket(DocumentationType.SWAGGER_2).select()

	.apis(RequestHandlerSelectors.basePackage("com.learn.quiz.controller
")).paths(PathSelectors.any())
				.build();
	}
}
```

Entity - LoginSession.java

```java
package com.learn.quiz.entity;

import java.util.Date;

public class LoginSession {

	private Long id;

	private Long userId;

	private String accessToken;

	private Date lastAccess;

	public LoginSession() {
	}

	public LoginSession(Long id, Long userId, String accessToken, Date
lastAccess) {
		super();
		this.id = id;
		this.userId = userId;
		this.accessToken = accessToken;
		this.lastAccess = lastAccess;
	}

	public Long getId() {
		return id;
	}

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

	public Long getUserId() {
		return userId;
	}

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

	public String getAccessToken() {
		return accessToken;
	}

	public void setAccessToken(String accessToken) {
		this.accessToken = accessToken;
	}

	public Date getLastAccess() {
		return lastAccess;
	}

	public void setLastAccess(Date lastAccess) {
		this.lastAccess = lastAccess;
	}

}
```

Entity - Question.java

```java
package com.learn.quiz.entity;

import java.util.Date;

import com.fasterxml.jackson.annotation.JsonIgnore;

public class Question {

	private Long id;

	private String question;

	private String optionA;

	private String optionB;

	private String optionC;

	private String optionD;

	private Character rightOption;

	@JsonIgnore
	private Character deleted = 'N';

	@JsonIgnore
	private Date createdOn = new Date();

	@JsonIgnore
	private Date updatedOn = new Date();

	public Question() {
		super();
		// TODO Auto-generated constructor stub
	}
	public Question(Long id, String question, String optionA, String
optionB, String optionC, String optionD,
			Character rightOption, Character deleted, Date
createdOn, Date updatedOn) {
		super();
		this.id = id;
		this.question = question;
		this.optionA = optionA;
		this.optionB = optionB;
		this.optionC = optionC;
		this.optionD = optionD;
		this.rightOption = rightOption;
		this.deleted = deleted;
		this.createdOn = createdOn;
		this.updatedOn = updatedOn;
	}

	public Long getId() {
		return id;
	}

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

	public String getQuestion() {
		return question;
	}

	public void setQuestion(String question) {
		this.question = question;
	}

	public String getOptionA() {
		return optionA;
	}

	public void setOptionA(String optionA) {
		this.optionA = optionA;
	}

	public String getOptionB() {
		return optionB;
	}

	public void setOptionB(String optionB) {
		this.optionB = optionB;
	}

	public String getOptionC() {
		return optionC;
	}

	public void setOptionC(String optionC) {
		this.optionC = optionC;
	}

	public String getOptionD() {
		return optionD;
	}

	public void setOptionD(String optionD) {
		this.optionD = optionD;
	}

	public Character getRightOption() {
		return rightOption;
	}

	public void setRightOption(Character rightOption) {
		this.rightOption = rightOption;
	}

	public Character getDeleted() {
		return deleted;
	}

	public void setDeleted(Character deleted) {
		this.deleted = deleted;
	}

	public Date getCreatedOn() {
		return createdOn;
	}

	public void setCreatedOn(Date createdOn) {
		this.createdOn = createdOn;
	}

	public Date getUpdatedOn() {
		return updatedOn;
	}

	public void setUpdatedOn(Date updatedOn) {
		this.updatedOn = updatedOn;
	}

}
```

Entity - Quiz.java

```
package com.learn.quiz.entity;

import java.util.Date;

import com.fasterxml.jackson.annotation.JsonIgnore;

public class Quiz {

	private Long id;

	private String title;

	private String category;

	@JsonIgnore
```

```java
	private Character deleted = 'N';

	@JsonIgnore
	private Date createdOn = new Date();

	@JsonIgnore
	private Date updatedOn = new Date();

	public Quiz() {
		super();
	}

	public Quiz(Long id, String title, String category, Character
deleted, Date createdOn, Date updatedOn) {
		super();
		this.id = id;
		this.title = title;
		this.category = category;
		this.deleted = deleted;
		this.createdOn = createdOn;
		this.updatedOn = updatedOn;
	}

	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 String getCategory() {
		return category;
	}

	public void setCategory(String category) {
		this.category = category;
	}

	public Character getDeleted() {
		return deleted;
	}

	public void setDeleted(Character deleted) {
		this.deleted = deleted;
	}

	public Date getCreatedOn() {
		return createdOn;
	}
	public void setCreatedOn(Date createdOn) {
		this.createdOn = createdOn;
	}

	public Date getUpdatedOn() {
		return updatedOn;
	}

	public void setUpdatedOn(Date updatedOn) {
		this.updatedOn = updatedOn;
	}

}
```

Entity - QuizQuestion.java

```
package com.learn.quiz.entity;

import java.util.Date;

import com.fasterxml.jackson.annotation.JsonIgnore;

public class QuizQuestion {

	private Long id;

	private Long quizId;

	private Long questionId;

	@JsonIgnore
	private Character deleted = 'N';

	@JsonIgnore
	private Date createdOn = new Date();

	@JsonIgnore
	private Date updatedOn = new Date();

	public QuizQuestion(Long id, Long quizId, Long questionId, Character
deleted, Date createdOn, Date updatedOn) {
		super();
		this.id = id;
		this.quizId = quizId;
		this.questionId = questionId;
		this.deleted = deleted;
		this.createdOn = createdOn;
		this.updatedOn = updatedOn;
	}

	public QuizQuestion() {
		super();
	}

	public Long getId() {
		return id;
	}

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

	public Long getQuizId() {
		return quizId;
	}

	public void setQuizId(Long quizId) {
		this.quizId = quizId;
	}

	public Long getQuestionId() {
		return questionId;
	}

	public void setQuestionId(Long questionId) {
		this.questionId = questionId;
	}

	public Character getDeleted() {
		return deleted;
	}

	public void setDeleted(Character deleted) {
		this.deleted = deleted;
	}

	public Date getCreatedOn() {
		return createdOn;
	}

	public void setCreatedOn(Date createdOn) {
		this.createdOn = createdOn;
	}

	public Date getUpdatedOn() {
		return updatedOn;
	}

	public void setUpdatedOn(Date updatedOn) {
		this.updatedOn = updatedOn;
	}

}
```

Entity - User.java

```
package com.learn.quiz.entity;

import java.util.Date;

import com.fasterxml.jackson.annotation.JsonIgnore;

public class User {

	private Long id;

	private String firstName;
	private String lastName;

	private String emailId;

	@JsonIgnore
	private String password;

	private String mobileNumber;

	@JsonIgnore
	private Character isAdmin;

	@JsonIgnore
	private Date createdOn;

	@JsonIgnore
	private Date updatedOn;

	public User() {
		super();
	}

	public User(Long id, String firstName, String lastName, String
emailId, String password, String mobileNumber,
			Character isAdmin, Date createdOn, Date updatedOn) {
		super();
		this.id = id;
		this.firstName = firstName;
		this.lastName = lastName;
		this.emailId = emailId;
		this.password = password;
		this.mobileNumber = mobileNumber;
		this.isAdmin = isAdmin;
		this.createdOn = createdOn;
		this.updatedOn = updatedOn;
	}

	public Long getId() {
		return id;
	}

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

	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;
```

```java
	}

	public String getEmailId() {
		return emailId;
	}

	public void setEmailId(String emailId) {
		this.emailId = emailId;
	}

	public String getPassword() {
		return password;
	}

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

	public String getMobileNumber() {
		return mobileNumber;
	}

	public void setMobileNumber(String mobileNumber) {
		this.mobileNumber = mobileNumber;
	}

	public Character getIsAdmin() {
		return isAdmin;
	}

	public void setIsAdmin(Character isAdmin) {
		this.isAdmin = isAdmin;
	}

	public Date getCreatedOn() {
		return createdOn;
	}

	public void setCreatedOn(Date createdOn) {
		this.createdOn = createdOn;
	}

	public Date getUpdatedOn() {
		return updatedOn;
	}

	public void setUpdatedOn(Date updatedOn) {
		this.updatedOn = updatedOn;
	}

}
```

Entity - UserQuizQuestionAnswer.java

```java
package com.learn.quiz.entity;

import java.util.Date;

import com.fasterxml.jackson.annotation.JsonIgnore;
```

```java
public class UserQuizQuestionAnswer {

	private Long id;

	private Long userId;

	private Long quizQuestionId;

	private Character selectedOption;

	@JsonIgnore
	private Date createdOn = new Date();

	public UserQuizQuestionAnswer(Long id, Long userId, Long
quizQuestionId, Character selectedOption, Date createdOn) {
		super();
		this.id = id;
		this.userId = userId;
		this.quizQuestionId = quizQuestionId;
		this.selectedOption = selectedOption;
		this.createdOn = createdOn;
	}

	public UserQuizQuestionAnswer() {
		super();
		// TODO Auto-generated constructor stub
	}

	public Long getId() {
		return id;
	}

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

	public Long getUserId() {
		return userId;
	}

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

	public Long getQuizQuestionId() {
		return quizQuestionId;
	}

	public void setQuizQuestionId(Long quizQuestionId) {
		this.quizQuestionId = quizQuestionId;
	}

	public Character getSelectedOption() {
		return selectedOption;
	}

	public void setSelectedOption(Character selectedOption) {
		this.selectedOption = selectedOption;
	}

	public Date getCreatedOn() {
		return createdOn;
	}

	public void setCreatedOn(Date createdOn) {
		this.createdOn = createdOn;
	}

}
```

DTO - Response.java

```
package com.learn.quiz.dto;

public class Response {

	private String message;

	public Response(String message) {
		super();
		this.message = message;
	}

	public String getMessage() {
		return message;
	}

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

}
```

DTO - SubmitQuestion.java

```
package com.learn.quiz.dto;

public class SubmitQuestion {

	private Long quizId;

	private Long questionId;

	private Character answer;

      public SubmitQuestion(Long quizId, Long questionId, Character answer)
{
		super();
		this.quizId = quizId;
		this.questionId = questionId;
		this.answer = answer;
	}

	public SubmitQuestion() {
		super();
		// TODO Auto-generated constructor stub
	}

	public Long getQuizId() {
		return quizId;
	}

	public void setQuizId(Long quizId) {
		this.quizId = quizId;
	}

	public Long getQuestionId() {
		return questionId;
	}

	public void setQuestionId(Long questionId) {
		this.questionId = questionId;
	}

	public Character getAnswer() {
		return answer;
	}

	public void setAnswer(Character answer) {
		this.answer = answer;
	}

}
```

DTO - UserQuizQuestionAnswerRequestDto.java

```
package com.learn.quiz.dto;

public class UserQuizQuestionAnswerRequestDto {

	private Long quizId;

	private Long quizQuestionId;

	private Character selectedOption;

	public UserQuizQuestionAnswerRequestDto(Long quizId, Long
quizQuestionId, Character selectedOption) {
		super();
		this.quizId = quizId;
		this.quizQuestionId = quizQuestionId;
		this.selectedOption = selectedOption;
	}

	public UserQuizQuestionAnswerRequestDto() {
		super();
	}

	public Long getQuizId() {
		return quizId;
	}

	public void setQuizId(Long quizId) {
		this.quizId = quizId;
	}
```

```java
	public Long getQuizQuestionId() {
		return quizQuestionId;
	}

	public void setQuizQuestionId(Long quizQuestionId) {
		this.quizQuestionId = quizQuestionId;
	}

	public Character getSelectedOption() {
		return selectedOption;
	}

	public void setSelectedOption(Character selectedOption) {
		this.selectedOption = selectedOption;
	}

}
```

DTO - UserScore.java

```java
package com.learn.quiz.dto;

public class UserScore {

	private String firstName;
	private String lastName;
	private Long score;

	public UserScore() {
		super();
	}

	public UserScore(String firstName, String lastName, Long score) {
		super();
		this.firstName = firstName;
		this.lastName = lastName;
		this.score = score;
	}

	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 Long getScore() {
		return score;
	}
	public void setScore(Long score) {
		this.score = score;
	}

}
```

Repoistory - LoginSessionDao

```java
package com.learn.quiz.repository;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.entity.LoginSession;

@Repository
public class LoginSessionDao {

	@Autowired
	JdbcTemplate jdbcTemplate;

	public LoginSession findByAccessToken(String accessToken) {
		List<LoginSession> rs = jdbcTemplate.query("SELECT ls.* from
login_session ls where ls.access_token = ?",
				new
BeanPropertyRowMapper<LoginSession>(LoginSession.class), new Object[]
{ accessToken });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

}
```

Repoistory - QuestionDao

```java
package com.learn.quiz.repository;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.entity.Question;

@Repository
public class QuestionDao {
	@Autowired
	JdbcTemplate jdbcTemplate;

	public Question findById(Long questionId) {
		List<Question> rs = jdbcTemplate.query("SELECT q.* from
question q where q.id = ? and q.deleted = 'N'",
				new
BeanPropertyRowMapper<Question>(Question.class), new Object[]
{ questionId });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

}
```

Repoistory - QuizDao

```
package com.learn.quiz.repository;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

@Repository
public class QuizDao {

	@Autowired
	JdbcTemplate jdbcTemplate;

}
```

Repoistory - QuizQuestionDao

```
package com.learn.quiz.repository;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.entity.QuizQuestion;

@Repository
public class QuizQuestionDao {

	@Autowired
	JdbcTemplate jdbcTemplate;

	public List<QuizQuestion> findByQuizId(Long quizId) {
		return jdbcTemplate.query("SELECT q.* from quiz_question q
where q.quiz_id = ? and q.deleted = 'N'",
                        new
BeanPropertyRowMapper<QuizQuestion>(QuizQuestion.class), new Object[]
{ quizId });
      }

      public QuizQuestion findByQuizIdAndQuestionId(Long quizId, Long
questionId) {
            List<QuizQuestion> rs = jdbcTemplate.query(
                        "SELECT q.* from quiz_question q where q.quiz_id
= ? and q.question_id = ? and q.deleted = 'N'",
                        new
BeanPropertyRowMapper<QuizQuestion>(QuizQuestion.class), new Object[]
{ quizId, questionId });
            if (rs != null && rs.size() > 0) {
                  return rs.get(0);
            } else {
                  return null;
            }
      }

      public QuizQuestion findById(Long quizQuestionId) {
            List<QuizQuestion> rs = jdbcTemplate.query("SELECT q.* from
quiz_question q where q.id = ? and q.deleted = 'N'",
                        new
BeanPropertyRowMapper<QuizQuestion>(QuizQuestion.class), new Object[]
{ quizQuestionId });
            if (rs != null && rs.size() > 0) {
                  return rs.get(0);
            } else {
                  return null;
            }
      }
}
```

Repoistory - UserDao

```
package com.learn.quiz.repository;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.entity.User;

@Repository
public class UserDao {

      @Autowired
      JdbcTemplate jdbcTemplate;

     public User getByEmailIdAndPassword(String userName, String password)
{
            List<User> rs = jdbcTemplate.query("SELECT u.* from user u
where u.email_id = ? and u.password = ?",
                        new BeanPropertyRowMapper<User>(User.class), new
Object[] { userName, password });
```

```
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

	public User findById(Long id) {
		List<User> rs = jdbcTemplate.query("SELECT u.* from user u 
where u.id = ?",
				new BeanPropertyRowMapper<User>(User.class), new 
Object[] { id });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

}
```

Repoistory - UserQuizQuestionAnswerDao

```
package com.learn.quiz.repository;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

import com.learn.quiz.dto.UserScore;
import com.learn.quiz.entity.UserQuizQuestionAnswer;

@Repository
public class UserQuizQuestionAnswerDao {

	@Autowired
	JdbcTemplate jdbcTemplate;

	public UserQuizQuestionAnswer findByQuizQuestionIdAndUserId(Long 
quizQuestionId, Long userId) {
		List<UserQuizQuestionAnswer> rs = jdbcTemplate.query(
				"SELECT q.* FROM quiz.user_quiz_ques_ans q where 
q.quiz_question_id = ? and q.user_id = ?",
				new 
BeanPropertyRowMapper<UserQuizQuestionAnswer>(UserQuizQuestionAnswer.class),
				new Object[] { quizQuestionId, userId });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

      public UserQuizQuestionAnswer findById(Long userQuizQuestionAnswerId) 
{
		List<UserQuizQuestionAnswer> rs = jdbcTemplate.query("SELECT
q.* FROM quiz.user_quiz_ques_ans q where q.id = ?",
					new
BeanPropertyRowMapper<UserQuizQuestionAnswer>(UserQuizQuestionAnswer.class),
					new Object[] { userQuizQuestionAnswerId });
		if (rs != null && rs.size() > 0) {
			return rs.get(0);
		} else {
			return null;
		}
	}

	public int save(UserQuizQuestionAnswer userQuizQuestionAnswer) {
		return jdbcTemplate.update(
					"INSERT INTO `quiz`.`user_quiz_ques_ans`
( `user_id`, `quiz_question_id`, `selected_option`) VALUES (?, ?, ? )",
				      new Object[] { userQuizQuestionAnswer.getUserId(),
userQuizQuestionAnswer.getQuizQuestionId(),

	userQuizQuestionAnswer.getSelectedOption().toString() });
	}

	public List<UserQuizQuestionAnswer> findAll() {
		return jdbcTemplate.query("SELECT q.* FROM quiz.cd  q",
					new
BeanPropertyRowMapper<UserQuizQuestionAnswer>(UserQuizQuestionAnswer.class)
);
	}

	public List<UserScore> getLeaderBoard(Long quizId) {
		return jdbcTemplate.query(
					"select u.first_name, u.last_name, count(uqqa.id)
score from quiz.user_quiz_ques_ans uqqa "
							+ " inner join quiz_question qq on
qq.id = uqqa.quiz_question_id "
							+ " inner join question q on q.id =
qq.question_id "
							+ " inner join user u on u.id =
uqqa.user_id "
							+ " where q.right_option =
uqqa.selected_option and qq.quiz_id = ? "
							+ " group by u.first_name,
u.last_name, uqqa.id order by count(uqqa.id) desc",
					new
BeanPropertyRowMapper<UserScore>(UserScore.class), quizId);
	}
}
```

Exception - InfoMessage

```
package com.learn.quiz.customException;

public class InfoMessage extends Exception {

	/**
	 *
	 */
```

```java
	private static final long serialVersionUID = -3772840913500662875L;

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

}
```

ExceptionHandler - CustomException

```java
package com.learn.quiz.exceptionHandler;

import org.springframework.http.HttpStatus;

public class CustomException extends RuntimeException {

	/**
	 * 
	 */
	private static final long serialVersionUID = -6971346490685985372L;

	private final HttpStatus httpStatus;

	public CustomException(String message, HttpStatus httpStatus) {
		super(message);
		this.httpStatus = httpStatus;
	}

	public HttpStatus getHttpStatus() {
		return httpStatus;
	}

}
```

Service - QuizQuestionService

```java
package com.learn.quiz.service;

import java.util.List;

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

import com.learn.quiz.customException.InfoMessage;
import com.learn.quiz.entity.Question;
import com.learn.quiz.entity.QuizQuestion;
import com.learn.quiz.entity.User;
import com.learn.quiz.exceptionHandler.CustomException;
import com.learn.quiz.repository.QuestionDao;
import com.learn.quiz.repository.QuizQuestionDao;
import com.learn.quiz.repository.UserQuizQuestionAnswerDao;

@Service
public class QuizQuestionService {

	@Autowired
	private QuizQuestionDao quizQuestionDao;
	@Autowired
	private UserQuizQuestionAnswerDao userQuizQuestionAnswerDao;

	@Autowired
	private QuestionDao questionDao;

	@Autowired
	private UserService userService;

	public QuizQuestion findById(Long questionId, String authorization)
{
		userService.getUser(authorization);
		QuizQuestion qq = quizQuestionDao.findById(questionId);
		if (qq == null) {
			throw new CustomException("QuizQuestion not found.",
HttpStatus.BAD_REQUEST);
		} else {
			return qq;
		}
	}

	public Question getNextQuestion(Long quizId, String authorization)
throws InfoMessage {

		User user = userService.getUser(authorization);
		List<QuizQuestion> quizQuestions =
quizQuestionDao.findByQuizId(quizId);
		for (QuizQuestion quizQuestion : quizQuestions) {
			if
(userQuizQuestionAnswerDao.findByQuizQuestionIdAndUserId(quizQuestion.getId
(), user.getId()) == null) {
				Question nextQuestion =
questionDao.findById(quizQuestion.getQuestionId());
				nextQuestion.setRightOption(null);
				return nextQuestion;
			}
		}
		throw new InfoMessage("No more question.");
	}

}
```

Service - UserQuizQuestionAnswerService

```
package com.learn.quiz.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Service;

import com.learn.quiz.dto.Response;
import com.learn.quiz.dto.SubmitQuestion;
import com.learn.quiz.dto.UserScore;
import com.learn.quiz.entity.QuizQuestion;
import com.learn.quiz.entity.User;
import com.learn.quiz.entity.UserQuizQuestionAnswer;
import com.learn.quiz.exceptionHandler.CustomException;
import com.learn.quiz.repository.QuizQuestionDao;
import com.learn.quiz.repository.UserQuizQuestionAnswerDao;

@Service
public class UserQuizQuestionAnswerService {

	@Autowired
	private UserQuizQuestionAnswerDao userQuizQuestionAnswerDao;

	@Autowired
	private QuizQuestionDao quizQuestionDao;

	@Autowired
	private UserService userService;

	public List<UserQuizQuestionAnswer> findAll() {
		return userQuizQuestionAnswerDao.findAll();
	}

      public  UserQuizQuestionAnswer  findById(Long  userQuizQuestionAnswerId,
String authorization) {
		userService.getUser(authorization);
		UserQuizQuestionAnswer answer =
userQuizQuestionAnswerDao.findById(userQuizQuestionAnswerId);
		if (answer == null) {
			throw new CustomException("UserQuizQuestionAnswer not
found.", HttpStatus.BAD_REQUEST);
		} else {
			return answer;
		}
	}

	public ResponseEntity<Response>
createUserQuizQuestionAnswer(SubmitQuestion submitQuestion, String
authorization) {
		User user = userService.getUser(authorization);
		QuizQuestion quizQuestion =
quizQuestionDao.findByQuizIdAndQuestionId(submitQuestion.getQuizId(),
				submitQuestion.getQuestionId());
		UserQuizQuestionAnswer userQuizQuestionAnswer = new
UserQuizQuestionAnswer();
		userQuizQuestionAnswer.setUserId(user.getId());
		userQuizQuestionAnswer.setQuizQuestionId(quizQuestion.getId());

	userQuizQuestionAnswer.setSelectedOption(submitQuestion.getAnswer());
		if (userQuizQuestionAnswerDao.save(userQuizQuestionAnswer) > 0)
{
			return new ResponseEntity<Response>(new
Response("Answer submitted successfully."), HttpStatus.OK);
		} else {
			return new ResponseEntity<Response>(new
Response("Answer not able to submit."),
					HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}

	public List<UserScore> getLeaderBoard(Long quizId) {
		return userQuizQuestionAnswerDao.getLeaderBoard(quizId);
```

```
        }

}
```

Service - UserService

```
package com.learn.quiz.service;

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

import com.learn.quiz.entity.LoginSession;
import com.learn.quiz.entity.User;
import com.learn.quiz.exceptionHandler.CustomException;
import com.learn.quiz.repository.LoginSessionDao;
import com.learn.quiz.repository.UserDao;

@Service
public class UserService {

        @Autowired
        private UserDao userDao;

        @Autowired
        private LoginSessionDao loginSessionDao;

        public User getUser(String authorization) {
                LoginSession loginSession =
loginSessionDao.findByAccessToken(authorization.split(" ")[1]);
                if (loginSession == null) {
                        throw new CustomException("Invalid access.",
HttpStatus.UNAUTHORIZED);
                }
                User user = userDao.findById(loginSession.getUserId());
                if (user == null | user.getIsAdmin() == 'Y') {
                        throw new CustomException("Invalid access.",
HttpStatus.UNAUTHORIZED);
                }
                return user;
        }

}
```

Controller -QuizController

```
package com.learn.quiz.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
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.RequestHeader;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.learn.quiz.customException.InfoMessage;
import com.learn.quiz.dto.Response;
import com.learn.quiz.dto.SubmitQuestion;
import com.learn.quiz.dto.UserScore;
import com.learn.quiz.entity.Question;
import com.learn.quiz.service.QuizQuestionService;
import com.learn.quiz.service.UserQuizQuestionAnswerService;

@RestController
@RequestMapping("/api/v1")
public class QuizController {

	@Autowired
	private QuizQuestionService quizQuestionService;

	@Autowired
	private UserQuizQuestionAnswerService userQuizQuestionAnswerService;

	@GetMapping("/get/quiz/next/question/{quizId}")
	public Question getNextQuizQuestion(@PathVariable Long quizId,
@RequestHeader("Authorization") String authorization)
			throws InfoMessage {
		return quizQuestionService.getNextQuestion(quizId,
authorization);
	}

	@PostMapping("/submit/question")
	public ResponseEntity<Response> submitQuestion(@RequestBody
SubmitQuestion submitQuestion,
			@RequestHeader("Authorization") String authorization) {
		return
userQuizQuestionAnswerService.createUserQuizQuestionAnswer(submitQuestion,
authorization);
	}

	@GetMapping("/leaderBoard/{quizId}")
	public List<UserScore> leaderBoard(@PathVariable Long quizId,
			@RequestHeader("Authorization") String authorization) {
		return userQuizQuestionAnswerService.getLeaderBoard(quizId);
	}
}
```

Controller -CustomExceptionHandler

```
package com.learn.quiz.controller;

import java.io.IOException;

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

import com.learn.quiz.customException.InfoMessage;
import com.learn.quiz.dto.Response;
import com.learn.quiz.exceptionHandler.CustomException;
```

```
@RestControllerAdvice
public class CustomExceptionHandler {

	@ExceptionHandler(IOException.class)
	public ResponseEntity<Response> ioExceptionHandler(IOException ex, 
WebRequest request) {
		ex.printStackTrace();
		return new ResponseEntity<Response>(new Response("Unable to 
access."), HttpStatus.BAD_REQUEST);
	}

	@ExceptionHandler(CustomException.class)
	public ResponseEntity<Response> 
customExceptionHandler(CustomException ex, WebRequest request) {
		ex.printStackTrace();
		return new ResponseEntity<Response>(new 
Response(ex.getLocalizedMessage()), ex.getHttpStatus());
	}

	@ExceptionHandler(InfoMessage.class)
	public ResponseEntity<Response> infoMessageHandler(InfoMessage ex, 
WebRequest request) {
		ex.printStackTrace();
		return new ResponseEntity<Response>(new 
Response(ex.getLocalizedMessage()), HttpStatus.OK);
	}
}
```

Controller - SpringFoxConfig

```
package com.learn.quiz.controller;

import org.springframework.context.annotation.Bean;

import springfox.documentation.builders.PathSelectors;
import springfox.documentation.builders.RequestHandlerSelectors;
import springfox.documentation.spi.DocumentationType;
import springfox.documentation.spring.web.plugins.Docket;

public class SpringFoxConfig {
	@Bean
	public Docket api() {
		return new 
Docket(DocumentationType.SWAGGER_2).select().apis(RequestHandlerSelectors.a
ny())
					.paths(PathSelectors.any()).build();
	}
}
```

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>
```

```xml
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-parent</artifactId>
			<version>2.5.6</version>
			<relativePath /> <!-- lookup parent from repository -->
		</parent>
		<groupId>com.learn.quiz</groupId>
		<artifactId>quiz-service</artifactId>
		<version>0.0.1-SNAPSHOT</version>
		<name>quiz-service</name>
		<description>Demo project for Spring Boot</description>
		<properties>
			<java.version>17</java.version>
		</properties>
		<dependencies>
			<dependency>
				<groupId>org.springframework.boot</groupId>
				<artifactId>spring-boot-starter-data-jdbc</artifactId>
			</dependency>
			<dependency>
				<groupId>org.springframework.boot</groupId>
				<artifactId>spring-boot-starter-web</artifactId>
			</dependency>
			<dependency>
				<groupId>io.springfox</groupId>
				<artifactId>springfox-boot-starter</artifactId>
				<version>3.0.0</version>
			</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-test</artifactId>
				<scope>test</scope>
			</dependency>

			<dependency>
				<groupId>mysql</groupId>
				<artifactId>mysql-connector-java</artifactId>
				<version>8.0.29</version>
			</dependency>
		</dependencies>

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

	</project>
```