Source code :

Displaying Feedback:

FeedbackApplication.java :

```
package com.example.feedbackapp;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;
@SpringBootApplication
public class FeedbackApplication implements WebMvcConfigurer {
public static void main(String[] args) {
SpringApplication.run(FeedbackApplication.class, args);
}
@Override
public void addResourceHandlers(ResourceHandlerRegistry registry) {
registry.addResourceHandler("/**")
.addResourceLocations("classpath:/static/");
}
}
```

ApplicationConfig.java:

```
package com.example.feedbackapp.config;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.EnableWebMvc;
@Configuration
@EnableWebMvc
@ComponentScan(basePackages = "com.example.feedbackapp.controller")
```

```java
public class ApplicationConfig {

}
```

Initializer.java:

```java
package com.example.feedbackapp.config;

import
org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class Initializer extends AbstractAnnotationConfigDispatcherServletInitializer
{

@Override

protected Class<?>[] getRootConfigClasses() {

// TODO Auto-generated method stub

return new Class[]{ApplicationConfig.class};

}

@Override

protected Class<?>[] getServletConfigClasses() {

// TODO Auto-generated method stub

return null;

}

@Override

protected String[] getServletMappings() {

// TODO Auto-generated method stub

 return new String[]{"/"};

}

}
```

CustomErrorController.java:

```java
package com.example.feedbackapp.controller;

import org.springframework.boot.web.servlet.error.ErrorController;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
@Controller
public class CustomErrorController implements ErrorController {
@RequestMapping("/error")
public String handleError() {
return "error";
}
public String getErrorPath() {
return null;
}


}
```

FeedbackController.java:

```
package com.example.feedbackapp.controller;
import com.example.feedbackapp.exception.FeedbackNotFoundException;
import com.example.feedbackapp.model.Feedback;
import com.example.feedbackapp.repository.FeedbackRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.servlet.ModelAndView;


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

```
@Controller
@RequestMapping("/feedback")
public class FeedbackController {


private final FeedbackRepository feedbackRepository;


@Autowired
public FeedbackController(FeedbackRepository feedbackRepository) {
this.feedbackRepository = feedbackRepository;
}


@GetMapping("/")
public String showIndexPage() {
return "index.html";
}


@GetMapping("/all")
public ResponseEntity<List<Feedback>> getAllFeedback() {
List<Feedback> feedbackList = feedbackRepository.findAll();
return ResponseEntity.ok(feedbackList);
}


@PostMapping
public ResponseEntity<Feedback> createFeedback(@RequestBody Feedback feedback) {
Feedback savedFeedback = feedbackRepository.save(feedback);
return ResponseEntity.status(HttpStatus.CREATED).body(savedFeedback);
}
@ExceptionHandler(FeedbackNotFoundException.class)
public ResponseEntity<String>
handleFeedbackNotFoundException(FeedbackNotFoundException ex) {
return ResponseEntity.status(HttpStatus.NOT_FOUND).body(ex.getMessage());
}


@RequestMapping("/error")
public String handleError() {
// Return the name of your custom error page HTML file
return "error.html";
}
}
```

FeedbackNotFoundException:

```
package com.example.feedbackapp.exception;


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


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


public FeedbackNotFoundException(String message) {
super(message);
}
}
```

Feedback.java:

```java
package com.example.feedbackapp.model;


import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.Table;
import jakarta.validation.constraints.NotBlank;


@Entity
@Table(name = "feedback")
public class Feedback {


@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;


@NotBlank
private String username;


@NotBlank
private String message;
```

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



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



public String getUsername() {
return username;
}



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



public String getMessage() {
return message;
}



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

// Getters and Setters go here

// Optional: Add constructors and other methods as needed
}
```

FeedbackRepository.java:

```
package com.example.feedbackapp.repository;

import com.example.feedbackapp.model.Feedback;

import java.util.Optional;

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

@Repository
public interface FeedbackRepository extends JpaRepository<Feedback, Long> {
 Optional<Feedback> findByIdAndUsername(Long id, String username);
Feedback save(Feedback any);
}
```

Index.html:

```html
<!DOCTYPE html>
<html>
<head>
<title>Feedback Form</title>
</head>
<body>
<h1>Feedback Form</h1>
<form id="feedbackForm">
<label for="username">Username:</label>
<input type="text" id="username" name="username" required><br><br>
<label for="message">Message:</label><br>
<textarea id="message" name="message" rows="5" cols="50" required></textarea><br><br>
<input type="submit" value="Submit">
</form>

<script>
document.getElementById("feedbackForm").addEventListener("submit", function(event) {
event.preventDefault();
var username = document.getElementById("username").value;
var message = document.getElementById("message").value;

var feedback = {
username: username,
message: message
};
fetch("/feedback", {
method: "POST",
headers: {
"Content-Type": "application/json"
},
body: JSON.stringify(feedback)
})
.then(function(response) {
if (response.ok) {
alert("Feedback submitted successfully!");
} else {
alert("Error submitting feedback. Please try again.");
}
})
.catch(function(error) {
console.log(error);
alert("An error occurred while submitting the feedback.");
});
});
</script>
</body>
</html>
```

Error.html:

```
<!DOCTYPE html>
<html>
<head>
<title>Error</title>
</head>
<body>
<h1>Oops! An error occurred.</h1>
```

```
<p>There was an unexpected error.</p>
<!-- Additional error information or custom error message can be added here -->
</body>
</html>
```

Application.properties :

```
spring.datasource.url=jdbc:mysql://localhost:3306/feedback_db
spring.datasource.username=root
spring.datasource.password=Deepika@1234
spring.jpa.hibernate.ddl-auto=create


spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
spring.jpa.open-in-view=false


server.port=8088
```

Pom.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>
<parent>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-parent</artifactId>
<version>3.1.1-SNAPSHOT</version>
<relativePath/> <!-- lookup parent from repository -->
```

```xml
</parent>
<groupId>com.example.feedbackapp</groupId>
<artifactId>Feedback</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>Feedback</name>
<description>Demo project for Spring Boot</description>
<properties>
<java.version>17</java.version>
</properties>
<dependencies>
<!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->
<!-- Spring Boot Web Starter -->
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
</dependency>


<!-- Spring Boot Data JPA Starter -->
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>


<!-- H2 Database -->
<dependency>
<groupId>com.h2database</groupId>
<artifactId>h2</artifactId>
<scope>runtime</scope>
</dependency>
```

```xml
<!-- MySQL Connector -->
<dependency>
<groupId>mysql</groupId>
<artifactId>mysql-connector-java</artifactId>
<version>8.0.28</version>
</dependency>

<!-- Spring Boot Validation -->
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-validation</artifactId>
</dependency>

<!-- Hibernate Validator -->
<dependency>
<groupId>org.hibernate.validator</groupId>
<artifactId>hibernate-validator</artifactId>
</dependency>

<!-- Spring Boot Test Starter -->
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-test</artifactId>
<scope>test</scope>
</dependency>
</dependencies>
```

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

My sql:

```sql
Create database feedback_db;
use feedback_db;
CREATE TABLE feedback (
  id INT PRIMARY KEY AUTO_INCREMENT,
  username VARCHAR(255) NOT NULL,
  message VARCHAR(1000) NOT NULL
);
select * from feedback;
```