

AppErrorController.java:-

```
package com.example.UserManager.controller;

import org.springframework.boot.web.servlet.error.ErrorController;
import org.springframework.web.bind.annotation.RequestMapping;

public class AppErrorController implements ErrorController {

    @RequestMapping("/error")
    public String handleError() {
        //do something like logging
        return "error";
    }

    @Override
    public String getErrorPath() {
        return null;
    }
}
```

MainController.java:-

```
package com.example.UserManager.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestParam;

@Controller
public class MainController {

    @GetMapping(value = "/")
    public String showIndexPage(ModelMap model,
                                @RequestParam(value = "name", required = false, defaultValue
= "World") String name) {
        model.addAttribute("name", name);
        return "index";
    }
}
```

UserController.java:-

```
package com.example.UserManager.controller;
```

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
```

```
import com.example.UserManager.entities.User;
import com.example.UserManager.services.UserService;
```

@Controller

```
public class UserController {
//controls the functionality of the user entity
```

 @Autowired

```
    private UserService userService;
```

```
    Logger logger = LoggerFactory.getLogger(UserController.class);
```

 @GetMapping("/users")

```
    public String showUsers(ModelMap model) {
        logger.info("Getting all users");
        Iterable<User> users = userService.GetAllUsers();
        logger.info("Passing users to view");
        model.addAttribute("users", users );
        return "users";
    }
```

 @RequestMapping(value = "/search/{id}", method = RequestMethod.POST)

```
    public String searchUser(ModelMap model, @RequestParam("id") int id) {
        logger.info("Searching for a user");
        User user = userService.GetUserById(id);
        logger.info("Passing Searched User to View");
        model.addAttribute("userSearch", user);
        return "search";
    }
```

 @PostMapping("search/update")

```

        public String updateUser(ModelMap model, @ModelAttribute("update") User
user) {
            logger.info("Updating a User");
            userService.UpdateUser(user);
            model.addAttribute("updatedUser", user);
            return "update";
        }
    }
}

```

UserExceptionHandler.java:-

```

package com.example.UserManager.controller;

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

import com.example.UserManager.exceptions.UserNotFoundException;

@ControllerAdvice
public class UserExceptionHandler {

    @ExceptionHandler(value=UserNotFoundException.class)
    public ResponseEntity<Object> exception(UserNotFoundException ex) {
        return new ResponseEntity<>("Product not found",
HttpStatus.NOT_FOUND);
    }

}

```

User.java:-

```

package com.example.UserManager.entities;

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

@Entity
public class User { //The Entity of a User; What it is.

    @Id

```

```
@GeneratedValue(strategy=GenerationType.AUTO)
private Integer id;

private String name;

private String email;

private String password;

public User() {
    super();
}

public User(Integer id, String name, String email, String password) {
    super();
    this.id = id;
    this.name = name;
    this.email = email;
    this.password = password;
}

public Integer getId() {
    return id;
}

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

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getEmail() {
    return email;
}

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

```

    public String getPassword() {
        return password;
    }

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

    @Override
    public String toString() {
        return (id.toString() + " " + name + " " + email + " " + password);
    }

}

```

UserNotFoundException.java:-

```

package com.example.UserManager.exceptions;

public class UserNotFoundException extends RuntimeException {

    private static final long serialVersionUID = 1L;

}

```

UserRepository.java:-

```

package com.example.UserManager.repositories;

import org.springframework.data.repository.CrudRepository;
import org.springframework.stereotype.Repository;

import com.example.UserManager.entities.User;

@Repository
public interface UserRepository extends CrudRepository<User, Integer> {

    public User findByName(String name);

}

```

UserService.java:-

```
package com.example.UserManager.services;

import java.util.Optional;

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

import com.example.UserManager.entities.User;
import com.example.UserManager.exceptions.UserNotFoundException;
import com.example.UserManager.repositories.UserRepository;

@Service
public class UserService {

    @Autowired
    private UserRepository userRepository;

    public Iterable<User> GetAllUsers() {
        return userRepository.findAll();
    }

    public User GetUserByName(String name) {
        return userRepository.findByName(name);
    }

    public User GetUserById(Integer id) {
        Optional<User> foundUser = userRepository.findById(id);
        if(!foundUser.isPresent()) throw new UserNotFoundException();
        return foundUser.get();
    }

    public User UpdateUser(User userToUpdate) {
        return userRepository.save(userToUpdate);
    }
}
```

UserManagerApplication.java:-

```
package com.example.UserManager;

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

@SpringBootApplication
public class UserManagerApplication {

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

}
```

Pom.xml:-

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.4.3</version>
        <relativePath /> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.example</groupId>
    <artifactId>UserManager</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>UserManager</name>
    <description>Searching for a Specific User and Updating the User
Information.</description>
    <properties>
        <java.version>1.8</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
```

```
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>

<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <scope>runtime</scope>
</dependency>
<dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <optional>true</optional>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
</dependency>

<dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>jstl</artifactId>
    <version>1.2</version>
</dependency>

<dependency>
    <groupId>org.apache.tomcat.embed</groupId>
    <artifactId>tomcat-embed-jasper</artifactId>
    <scope>provided</scope>
</dependency>

<dependency>
    <groupId>javax.xml.bind</groupId>
    <artifactId>jaxb-api</artifactId>
</dependency>

<dependency>
    <groupId>org.javassist</groupId>
    <artifactId>javassist</artifactId>
    <version>3.25.0-GA</version>
</dependency>
```



```

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-devtools</artifactId>
            <scope>runtime</scope>
            <optional>true</optional>
        </dependency>
    </dependencies>

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

<groupId>org.projectlombok</groupId>
                                <artifactId>lombok</artifactId>
                        </exclude>
                    </excludes>
                </configuration>
            </plugin>
        </plugins>
    </build>

</project>

```