Handling USer Authentication--->

main --->

```
package com.example.Auth;

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

@SpringBootApplication


public class HandlingUserAuthenticationApplication {

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

}
```

Entity class---->

```
package com.example.Auth.entity;

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

@Entity
@Table(name="Auth")
public class UserEntity {

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

        private String name;
        private String email;
        private String password;
        public int getId() {
                return id;
        }
        public void setId(int 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 "UserEntity [id=" + id + ", name=" + name + ", email=" +
email + ", password=" + password + "]";
        }



}
```

Repository----->

```
package com.example.Auth.Repository;

import java.util.Optional;

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

import com.example.Auth.entity.UserEntity;
```

```
@Repository
public interface AuthRepo extends CrudRepository<UserEntity, Integer> {

         Optional<UserEntity> findUserByName(String name);
}
```

Sevice class--->

```
package com.example.Auth.Service;

import java.util.Optional;

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

import com.example.Auth.Repository.AuthRepo;
import com.example.Auth.entity.UserEntity;
import com.example.Auth.exception.UserNotFoundExceptions;

@Service
public class AuthenticationService {

        @Autowired
        private AuthRepo repo;

        public UserEntity GetUserByName(String name) {
                Optional<UserEntity> found = repo.findUserByName(name);
                if (found.isPresent())
                        return found.get();
                else
                        throw new UserNotFoundExceptions();
        }

        public Boolean isValidPassword(String cmp, String actual) {
                return ((cmp.equals(actual)) ? true : false);
        }

}
```

Controller class--->

```
package com.example.Auth.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;

import com.example.Auth.Service.AuthenticationService;
import com.example.Auth.entity.UserEntity;



@Controller
@RequestMapping("/User")
public class UserController {


        @Autowired
        AuthenticationService authService;

        @GetMapping("/Auth")
        public String showLogin() {
                return "auth";
        }

        @PostMapping("/Auth")
        @ResponseBody
        public String authenticateUser(@RequestParam("username") String username,
@RequestParam("password") String password) {
                UserEntity user = authService.GetUserByName(username);
                String smt = (authService.isValidPassword(password,
user.getPassword())) ? "success" : "failure";
                return smt;
        }
}
```

Exception --->

```
package com.example.Auth.exception;

public class UserNotFoundExceptions extends  RuntimeException  {

        /**
```

```
         *
         */
        private static final long serialVersionUID = 1L;

}
```

properties file--->

```
spring.jpa.hibernate.ddl-auto=create
spring.datasource.url=jdbc:mysql://localhost:3306/mydb
spring.datasource.username=root
spring.datasource.password=Veda@1202189

logging.level.org.springframework.web: DEBUG
spring.mvc.view.prefix=/WEB-INF/views/
spring.mvc.view.suffix=.jsp
server.port=8087


spring.jpa.properties.hibernate.show_sql=true
spring.jpa.properties.hibernate.format_sql=true


logging.level.org.hibernate=INFO
logging.level.org.hibernate.SQL=INFO
logging.level.org.hibernate.type.descriptor.sql.BasicBinder=TRACE
logging.level.org.springframework=TRACE
#logging.level.org.apache=ERROR
#logging.level.org.springframework.web=TRACE
```