**Source Code**

MyMoviePlanApplication.Java

**package com.MyMoviePlan;**

**import org.springframework.boot.SpringApplication;**

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

**@SpringBootApplication**

**public class MyMoviePlanApplication {**

**public static void main(String[] args) {**

**SpringApplication.run(MyMoviePlanApplication.class, args);**

**}**

**}**

**2. ServletIntializer.java**

**package com.MyMoviePlan;**

**import org.springframework.boot.builder.SpringApplicationBuilder;**

**import org.springframework.boot.web.servlet.support.SpringBootServletInitializer;**

**public class ServletInitializer extends SpringBootServletInitializer {**

**@Override**

**protected SpringApplicationBuilder configure(SpringApplicationBuilder application) {**

**return application.sources(MyMoviePlanApplication.class);**

**}**

**}**

**3. ActorEntity**

**package com.MyMoviePlan.entity;**

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

**import lombok.\*;**

**import javax.persistence.\*;**

**import java.io.Serializable;**

**@Entity**

**@Data**

**@AllArgsConstructor**

**@NoArgsConstructor**

**@EqualsAndHashCode**

**@Table(name = "actors")**

**public class ActorEntity implements Serializable {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private int id;**

**@Column(name = "is\_cast")**

**private String isCast;**

**private String name;**

**private String role;**

**@Column(length = Integer.MAX\_VALUE, columnDefinition="TEXT")**

**private String image;**

**@JsonIgnore**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@ManyToOne(targetEntity = MovieEntity.class)**

**private MovieEntity movie;**

**public ActorEntity(String name, String role, String image) {**

**this.name = name;**

**this.role = role;**

**this.image = image;**

**}**

**}**

**Auditorium Entity**

**package com.MyMoviePlan.entity;**

**import lombok.\*;**

**import javax.persistence.\*;**

**import java.io.Serializable;**

**import java.util.List;**

**//@JsonIdentityInfo(generator = ObjectIdGenerators.PropertyGenerator.class,**

**// property = "id", scope = ShowEntity.class)**

**@Entity**

**@Data**

**@NoArgsConstructor**

**@AllArgsConstructor**

**@EqualsAndHashCode**

**@Table(name = "auditoriums")**

**public class AuditoriumEntity implements Serializable {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private int id;**

**private String name;**

**@Column(length = Integer.MAX\_VALUE, columnDefinition="TEXT")**

**private String image;**

**private String email;**

**@Column(name = "customer\_care\_no")**

**private String customerCareNo;**

**private String address;**

**@Column(name = "seat\_capacity")**

**private int seatCapacity;**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@ElementCollection**

**@CollectionTable(name = "auditorium\_facilities", joinColumns = @JoinColumn(name = "auditorium\_id"))**

**@Column(name = "facility")**

**private List<String> facilities;**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@ElementCollection**

**@CollectionTable(name = "auditorium\_safeties", joinColumns = @JoinColumn(name = "auditorium\_id"))**

**@Column(name = "safety")**

**private List<String> safeties;**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@JoinColumn(name = "auditorium\_id", referencedColumnName = "id")**

**@OneToMany(targetEntity = ShowEntity.class, cascade = CascadeType.ALL)**

**// @JoinTable(name = "auditorium\_shows",**

**// joinColumns = @JoinColumn(name = "auditorium\_id", unique = false),**

**// inverseJoinColumns = @JoinColumn(name = "show\_id", unique = false))**

**private List<ShowEntity> shows;**

**public AuditoriumEntity(String name, String image, String email, String customerCareNo, String address,**

**int seatCapacity, List<String> facilities, List<String> safeties, List<ShowEntity> shows) {**

**this.name = name;**

**this.image = image;**

**this.email = email;**

**this.customerCareNo = customerCareNo;**

**this.address = address;**

**this.seatCapacity = seatCapacity;**

**this.facilities = facilities;**

**this.safeties = safeties;**

**this.shows = shows;**

**}**

**public AuditoriumEntity setId(int id) {**

**this.id = id;**

**return this;**

**}**

**public AuditoriumEntity setName(String name) {**

**this.name = name;**

**return this;**

**}**

**public AuditoriumEntity setImage(String image) {**

**this.image = image;**

**return this;**

**}**

**public AuditoriumEntity setEmail(String email) {**

**this.email = email;**

**return this;**

**}**

**public AuditoriumEntity setCustomerCare(String customerCareNo) {**

**this.customerCareNo = customerCareNo;**

**return this;**

**}**

**public AuditoriumEntity setAddress(String address) {**

**this.address = address;**

**return this;**

**}**

**public AuditoriumEntity setSeatCapacity(int seatCapacity) {**

**this.seatCapacity = seatCapacity;**

**return this;**

**}**

**public AuditoriumEntity setFacilities(List<String> facilities) {**

**this.facilities = facilities;**

**return this;**

**}**

**public AuditoriumEntity setSafeties(List<String> safeties) {**

**this.safeties = safeties;**

**return this;**

**}**

**public AuditoriumEntity setShows(List<ShowEntity> shows) {**

**this.shows = shows;**

**return this;**

**}**

**}**

**Booking Details entity**

**package com.MyMoviePlan.entity;**

**import lombok.AllArgsConstructor;**

**import lombok.Data;**

**import lombok.EqualsAndHashCode;**

**import lombok.NoArgsConstructor;**

**import javax.persistence.\*;**

**import java.io.Serializable;**

**@Entity**

**@Data**

**@AllArgsConstructor**

**@NoArgsConstructor**

**@EqualsAndHashCode**

**@Table(name = "booking\_details")**

**public class BookingDetailsEntity implements Serializable {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private int id;**

**@Column(name = "auditorium\_id")**

**private int auditoriumId;**

**@Column(name = "show\_id")**

**private int showId;**

**@Column(name = "movie\_show\_id")**

**private int movieShowId;**

**@Column(name = "movie\_id")**

**private int movieId;**

**public BookingDetailsEntity(int auditoriumId, int showId, int movieShowId, int movieId) {**

**this.auditoriumId = auditoriumId;**

**this.showId = showId;**

**this.movieShowId = movieShowId;**

**this.movieId = movieId;**

**}**

**}**

**Booking Entity**

**package com.MyMoviePlan.entity;**

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

**import lombok.\*;**

**import javax.persistence.\*;**

**import java.io.Serializable;**

**import java.util.Date;**

**import java.util.List;**

**@Entity**

**@Data**

**@AllArgsConstructor**

**@NoArgsConstructor**

**@EqualsAndHashCode**

**@Table(name = "bookings")**

**public class BookingEntity implements Serializable {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private int id;**

**private double amount;**

**@Column(name = "total\_seats")**

**private int totalSeats;**

**@Column(name = "booked\_on")**

**@Temporal(TemporalType.DATE)**

**private Date bookedOn;**

**@Column(name = "date\_of\_booking")**

**@Temporal(TemporalType.DATE)**

**private Date dateOfBooking;**

**@Column(name = "user\_id")**

**private String userId;**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@ElementCollection**

**@CollectionTable(name = "booked\_seats", joinColumns = @JoinColumn(name = "booking\_id"))**

**@Column(name = "seat\_numbers")**

**private List<String> seatNumbers;**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@OneToOne(targetEntity = PaymentEntity.class, cascade = CascadeType.ALL)**

**@JoinColumn(name = "payment\_id")**

**private PaymentEntity payment;**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@OneToOne(targetEntity = BookingDetailsEntity.class, cascade = CascadeType.ALL)**

**@JoinColumn(name = "booking\_details\_id")**

**private BookingDetailsEntity bookingDetails;**

**@JsonIgnore**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@ManyToOne(targetEntity = MovieShowsEntity.class)**

**private MovieShowsEntity movieShow;**

**public BookingEntity(double amount, int totalSeats, Date bookedOn, Date dateOfBooking, List<String> seatNumbers,**

**PaymentEntity payment, String userId, MovieShowsEntity movieShow) {**

**this.amount = amount;**

**this.totalSeats = totalSeats;**

**this.bookedOn = bookedOn;**

**this.dateOfBooking = dateOfBooking;**

**this.seatNumbers = seatNumbers;**

**this.payment = payment;**

**this.userId = userId;**

**this.movieShow = movieShow;**

**}**

**public BookingEntity setMovieShow(MovieShowsEntity movieShow) {**

**this.movieShow = movieShow;**

**return this;**

**}**

**public BookingEntity setId(int id) {**

**this.id = id;**

**return this;**

**}**

**public BookingEntity setAmount(double amount) {**

**this.amount = amount;**

**return this;**

**}**

**public BookingEntity setTotalSeats(int totalSeats) {**

**this.totalSeats = totalSeats;**

**return this;**

**}**

**public BookingEntity setStatus(Date bookedOn) {**

**this.bookedOn = bookedOn;**

**return this;**

**}**

**public BookingEntity setDateOfBooking(Date dateOfBooking) {**

**this.dateOfBooking = dateOfBooking;**

**return this;**

**}**

**public BookingEntity setSeatNumbers(List<String> seatNumbers) {**

**this.seatNumbers = seatNumbers;**

**return this;**

**}**

**public BookingEntity setPayment(PaymentEntity payment) {**

**this.payment = payment;**

**return this;**

**}**

**public BookingEntity setUserId(String userId) {**

**this.userId = userId;**

**return this;**

**}**

**}**

**package com.MyMoviePlan.entity;**

**import lombok.\*;**

**import javax.persistence.\*;**

**import java.io.Serializable;**

**import java.util.Date;**

**import java.util.List;**

**@Entity**

**@Data**

**@AllArgsConstructor**

**@NoArgsConstructor**

**@EqualsAndHashCode**

**@Table(name = "movies")**

**public class MovieEntity implements Serializable {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private int id;**

**private String name;**

**@Column(length = Integer.MAX\_VALUE, columnDefinition = "TEXT")**

**private String image;**

**@Column(name = "bg\_image", length = Integer.MAX\_VALUE, columnDefinition="TEXT")**

**private String bgImage;**

**@Column(length = 9000)**

**private String story;**

**private String year;**

**private String duration;**

**private String caption;**

**@Column(name = "added\_on")**

**@Temporal(TemporalType.DATE)**

**private Date addedOn;**

**@Temporal(TemporalType.DATE)**

**private Date release;**

**private String language;**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@ElementCollection**

**@CollectionTable(name = "movie\_genres", joinColumns = @JoinColumn(name = "movie\_id"))**

**@Column(name = "genre")**

**private List<String> genres;**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@OneToMany(targetEntity = ActorEntity.class, cascade = CascadeType.ALL)**

**@JoinColumn(name = "movie\_id", referencedColumnName = "id")**

**private List<ActorEntity> casts;**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@OneToMany(targetEntity = ActorEntity.class, cascade = CascadeType.ALL)**

**@JoinColumn(name = "movie\_id", referencedColumnName = "id")**

**private List<ActorEntity> crews;**

**public MovieEntity(String name, String image, String bgImage, String story, String year,**

**String duration, String caption, Date addedOn, Date release, String language,**

**List<String> genres, List<ActorEntity> casts, List<ActorEntity> crews) {**

**this.name = name;**

**this.image = image;**

**this.bgImage = bgImage;**

**this.story = story;**

**this.year = year;**

**this.duration = duration;**

**this.caption = caption;**

**this.addedOn = addedOn;**

**this.release = release;**

**this.language = language;**

**this.genres = genres;**

**this.casts = casts;**

**this.crews = crews;**

**}**

**public MovieEntity setId(int id) {**

**this.id = id;**

**return this;**

**}**

**public MovieEntity setName(String name) {**

**this.name = name;**

**return this;**

**}**

**public MovieEntity setImage(String image) {**

**this.image = image;**

**return this;**

**}**

**public MovieEntity setBgImage(String bgImage) {**

**this.bgImage = bgImage;**

**return this;**

**}**

**public MovieEntity setStory(String story) {**

**this.story = story;**

**return this;**

**}**

**public MovieEntity setYear(String year) {**

**this.year = year;**

**return this;**

**}**

**public MovieEntity setDuration(String duration) {**

**this.duration = duration;**

**return this;**

**}**

**public MovieEntity setCaption(String caption) {**

**this.caption = caption;**

**return this;**

**}**

**public MovieEntity setAddedOn(Date addedOn) {**

**this.addedOn = addedOn;**

**return this;**

**}**

**public MovieEntity setRelease(Date release) {**

**this.release = release;**

**return this;**

**}**

**public MovieEntity setLanguages(String language) {**

**this.language = language;**

**return this;**

**}**

**public MovieEntity setGenres(List<String> genres) {**

**this.genres = genres;**

**return this;**

**}**

**public MovieEntity setCasts(List<ActorEntity> casts) {**

**this.casts = casts;**

**return this;**

**}**

**public MovieEntity setCrews(List<ActorEntity> crews) {**

**this.crews = crews;**

**return this;**

**}**

**}**

**MoviesShowsEntity**

**package com.MyMoviePlan.entity;**

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

**import lombok.\*;**

**import javax.persistence.\*;**

**import java.io.Serializable;**

**import java.util.Date;**

**import java.util.List;**

**@Entity**

**@Data**

**@AllArgsConstructor**

**@NoArgsConstructor**

**@EqualsAndHashCode**

**@Table(name = "movie\_shows")**

**public class MovieShowsEntity implements Serializable {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private int id;**

**@Temporal(TemporalType.DATE)**

**@Column(name = "show\_start")**

**private Date start;**

**@Temporal(TemporalType.DATE)**

**@Column(name = "show\_end")**

**private Date end;**

**@Column(name = "movie\_id")**

**private int movieId;**

**@JsonIgnore**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@ManyToOne(targetEntity = ShowEntity.class)**

**private ShowEntity show;**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@JoinColumn(name = "movie\_show\_id", referencedColumnName = "id")**

**@OneToMany(targetEntity = BookingEntity.class, cascade = CascadeType.ALL)**

**// @JoinTable(name = "movie\_show\_bookings",**

**// joinColumns = @JoinColumn(name = "movie\_show\_id", unique = false),**

**// inverseJoinColumns = @JoinColumn(name = "booking\_id", unique = false))**

**private List<BookingEntity> bookings;**

**@ToString.Exclude**

**@EqualsAndHashCode.Exclude**

**@OneToOne(targetEntity = PriceEntity.class, cascade = CascadeType.ALL)**

**@JoinColumn(name = "price\_id")**

**private PriceEntity price;**

**public MovieShowsEntity(int id, Date start, Date end, List<BookingEntity> bookings, int movieId) {**

**this.id = id;**

**this.start = start;**

**this.end = end;**

**this.bookings = bookings;**

**this.movieId = movieId;**

**}**

**public MovieShowsEntity setId(int id) {**

**this.id = id;**

**return this;**

**}**

**public MovieShowsEntity setStart(Date start) {**

**this.start = start;**

**return this;**

**}**

**public MovieShowsEntity setEnd(Date end) {**

**this.end = end;**

**return this;**

**}**

**public MovieShowsEntity setShow(ShowEntity show) {**

**this.show = show;**

**return this;**

**}**

**public MovieShowsEntity setMovieId(int movieId) {**

**this.movieId = movieId;**

**return this;**

**}**

**}**

Payment Entity

**package com.MyMoviePlan.entity;**

**import lombok.AllArgsConstructor;**

**import lombok.Data;**

**import lombok.EqualsAndHashCode;**

**import lombok.NoArgsConstructor;**

**import javax.persistence.\*;**

**import java.io.Serializable;**

**import java.util.Date;**

**@Entity**

**@Data**

**@AllArgsConstructor**

**@NoArgsConstructor**

**@EqualsAndHashCode**

**@Table(name = "payments")**

**public class PaymentEntity implements Serializable {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private int id;**

**private double amount;**

**@Column(name = "payment\_date")**

**@Temporal(TemporalType.DATE)**

**private Date paymentDate;**

**@Column(name = "card\_number", length = 20)**

**private String cardNumber;**

**@Column(name = "card\_expiry\_month", length = 5)**

**private String cardExpiryMonth;**

**@Column(name = "card\_expiry\_year", length = 5)**

**private String cardExpiryYear;**

**@Column(name = "card\_cvv", length = 5)**

**private String cardCVV;**

**public PaymentEntity(double amount, Date paymentDate, String cardNumber, String cardExpiryMonth,**

**String cardExpiryYear, String cardCVV) {**

**this.amount = amount;**

**this.paymentDate = paymentDate;**

**this.cardNumber = cardNumber;**

**this.cardExpiryMonth = cardExpiryMonth;**

**this.cardExpiryYear = cardExpiryYear;**

**this.cardCVV = cardCVV;**

**}**

**public PaymentEntity setId(int id) {**

**this.id = id;**

**return this;**

**}**

**public PaymentEntity setAmount(double amount) {**

**this.amount = amount;**

**return this;**

**}**

**public PaymentEntity setPaymentDate(Date paymentDate) {**

**this.paymentDate = paymentDate;**

**return this;**

**}**

**public PaymentEntity setCardNumber(String cardNumber) {**

**this.cardNumber = cardNumber;**

**return this;**

**}**

**public PaymentEntity setCardExpiryMonth(String cardExpiryMonth) {**

**this.cardExpiryMonth = cardExpiryMonth;**

**return this;**

**}**

**public PaymentEntity setCardExpiryYear(String cardExpiryYear) {**

**this.cardExpiryYear = cardExpiryYear;**

**return this;**

**}**

**public PaymentEntity setCardCVV(String cardCVV) {**

**this.cardCVV = cardCVV;**

**return this;**

**}**

**}**

Price Entity

**package com.MyMoviePlan.entity;**

**import lombok.AllArgsConstructor;**

**import lombok.Data;**

**import lombok.EqualsAndHashCode;**

**import lombok.NoArgsConstructor;**

**import javax.persistence.\*;**

**import java.io.Serializable;**

**@Entity**

**@Data**

**@AllArgsConstructor**

**@NoArgsConstructor**

**@EqualsAndHashCode**

**@Table(name = "prices")**

**public class PriceEntity implements Serializable {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private int id;**

**private double general;**

**private double silver;**

**private double gold;**

**public PriceEntity(double general, double silver, double gold) {**

**this.general = general;**

**this.silver = silver;**

**this.gold = gold;**

**}**

**}**

Show Entity

package com.MyMoviePlan.entity;

import com.fasterxml.jackson.annotation.JsonIgnore;

import lombok.\*;

import javax.persistence.\*;

import java.io.Serializable;

import java.util.List;

@Entity

@Data

@AllArgsConstructor

@NoArgsConstructor

@EqualsAndHashCode

@Table(name = "shows")

public class ShowEntity implements Serializable {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String name;

@Column(name = "start\_time")

private String startTime;

@JsonIgnore

@ToString.Exclude

@EqualsAndHashCode.Exclude

@ManyToOne(targetEntity = AuditoriumEntity.class)

private AuditoriumEntity auditorium;

// @JsonManagedReference

@ToString.Exclude

@EqualsAndHashCode.Exclude

@OneToMany(targetEntity = MovieShowsEntity.class, cascade = CascadeType.ALL)

@JoinColumn(name = "show\_id", referencedColumnName = "id")

private List<MovieShowsEntity> movieShows;

public ShowEntity(String name, String startTime, List<MovieShowsEntity> movieShows) {

this.name = name;

this.startTime = startTime;

this.movieShows = movieShows;

}

public ShowEntity setId(int id) {

this.id = id;

return this;

}

public ShowEntity setName(String name) {

this.name = name;

return this;

}

public ShowEntity setStartTime(String startTime) {

this.startTime = startTime;

return this;

}

public ShowEntity setAuditorium(AuditoriumEntity auditorium) {

this.auditorium = auditorium;

return this;

}

public ShowEntity setMovieShows(List<MovieShowsEntity> movieShows) {

this.movieShows = movieShows;

return this;

}

}

UserEntity

package com.MyMoviePlan.entity;

import com.MyMoviePlan.model.UserRole;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.EqualsAndHashCode;

import lombok.NoArgsConstructor;

import org.hibernate.annotations.GenericGenerator;

import javax.persistence.\*;

import java.io.Serializable;

@Entity

@Data

@AllArgsConstructor

@NoArgsConstructor

@EqualsAndHashCode

@Table(name = "users")

public class UserEntity implements Serializable {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY, generator = "uuid2")

@GenericGenerator(name = "uuid2", strategy = "uuid2")

private String id;

@Column(length = 50)

private String name;

@Column(nullable = false, length = 50, unique = true)

private String email;

@Column(nullable = false, length = 10, unique = true)

private String mobile;

@Column(length = 60)

private String gender;

private String password;

private Boolean terms;

@Column(name = "is\_account\_non\_expired")

private Boolean isAccountNonExpired;

@Column(name = "is\_account\_non\_locked")

private Boolean isAccountNonLocked;

@Column(name = "is\_credentials\_non\_expired")

private Boolean isCredentialsNonExpired;

@Column(name = "is\_enabled")

private Boolean isEnabled;

@Column(name = "user\_role", length = 20)

@Enumerated(EnumType.STRING)

private UserRole userRole;

public UserEntity(String name, String email, String mobile, String gender, String password, Boolean terms,

Boolean isAccountNonExpired, Boolean isAccountNonLocked,

Boolean isCredentialsNonExpired, Boolean isEnabled, UserRole userRole) {

this.name = name;

this.email = email;

this.mobile = mobile;

this.gender = gender;

this.password = password;

this.terms = terms;

this.isAccountNonExpired = isAccountNonExpired;

this.isAccountNonLocked = isAccountNonLocked;

this.isCredentialsNonExpired = isCredentialsNonExpired;

this.isEnabled = isEnabled;

this.userRole = userRole;

}

public UserEntity setId(String id) {

this.id = id;

return this;

}

public UserEntity setName(String name) {

this.name = name;

return this;

}

public UserEntity setEmail(String email) {

this.email = email;

return this;

}

public UserEntity setMobile(String mobile) {

this.mobile = mobile;

return this;

}

public UserEntity setGender(String gender) {

this.gender = gender;

return this;

}

public UserEntity setPassword(String password) {

this.password = password;

return this;

}

public UserEntity setActive(Boolean active) {

terms = active;

return this;

}

public UserEntity setAccountNonExpired(Boolean accountNonExpired) {

isAccountNonExpired = accountNonExpired;

return this;

}

public UserEntity setAccountNonLocked(Boolean accountNonLocked) {

isAccountNonLocked = accountNonLocked;

return this;

}

public UserEntity setCredentialsNonExpired(Boolean credentialsNonExpired) {

isCredentialsNonExpired = credentialsNonExpired;

return this;

}

public UserEntity setEnabled(Boolean enabled) {

isEnabled = enabled;

return this;

}

public UserEntity setUserRole(UserRole userRole) {

this.userRole = userRole;

return this;

}

public UserEntity setTerms(Boolean terms) {

this.terms = terms;

return this;

}

public UserEntity(String id, String name, String email, String mobile, String gender, String password,

Boolean terms, Boolean isAccountNonExpired, Boolean isAccountNonLocked, Boolean isCredentialsNonExpired,

Boolean isEnabled, UserRole userRole) {

super();

this.id = id;

this.name = name;

this.email = email;

this.mobile = mobile;

this.gender = gender;

this.password = password;

this.terms = terms;

this.isAccountNonExpired = isAccountNonExpired;

this.isAccountNonLocked = isAccountNonLocked;

this.isCredentialsNonExpired = isCredentialsNonExpired;

this.isEnabled = isEnabled;

this.userRole = userRole;

}

@Override

public String toString() {

return "UserEntity [id=" + id + ", name=" + name + ", email=" + email + ", mobile=" + mobile + ", gender="

+ gender + ", password=" + password + ", terms=" + terms + ", isAccountNonExpired="

+ isAccountNonExpired + ", isAccountNonLocked=" + isAccountNonLocked + ", isCredentialsNonExpired="

+ isCredentialsNonExpired + ", isEnabled=" + isEnabled + ", userRole=" + userRole + "]";

}

public Boolean getIsAccountNonExpired() {

return isAccountNonExpired;

}

public void setIsAccountNonExpired(Boolean isAccountNonExpired) {

this.isAccountNonExpired = isAccountNonExpired;

}

public Boolean getIsAccountNonLocked() {

return isAccountNonLocked;

}

public void setIsAccountNonLocked(Boolean isAccountNonLocked) {

this.isAccountNonLocked = isAccountNonLocked;

}

public Boolean getIsCredentialsNonExpired() {

return isCredentialsNonExpired;

}

public void setIsCredentialsNonExpired(Boolean isCredentialsNonExpired) {

this.isCredentialsNonExpired = isCredentialsNonExpired;

}

public Boolean getIsEnabled() {

return isEnabled;

}

public void setIsEnabled(Boolean isEnabled) {

this.isEnabled = isEnabled;

}

public String getId() {

return id;

}

public String getName() {

return name;

}

public String getEmail() {

return email;

}

public String getMobile() {

return mobile;

}

public String getGender() {

return gender;

}

public String getPassword() {

return password;

}

public Boolean getTerms() {

return terms;

}

public UserRole getUserRole() {

return userRole;

}

}

Bean Supplier

package com.MyMoviePlan.util;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.security.crypto.password.PasswordEncoder;

@Configuration

public class BeanSupplier {

@Bean

public PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder(10);

}

// @Bean

// public FilterRegistrationBean corsFilter() {

// final UrlBasedCorsConfigurationSource source = new UrlBasedCorsConfigurationSource();

// CorsConfiguration config = new CorsConfiguration();

// config.setAllowCredentials(Boolean.TRUE);

// config.addAllowedOrigin(CorsConfiguration.ALL);

// config.addAllowedHeader(CorsConfiguration.ALL);

// config.addAllowedMethod(CorsConfiguration.ALL);

// source.registerCorsConfiguration("/\*\*", config);

// FilterRegistrationBean bean = new FilterRegistrationBean();

// bean.setFilter(new CorsFilter());

// bean.setOrder(0);

// return bean;

// }

}

package com.MyMoviePlan.util;

import io.jsonwebtoken.Claims;

import io.jsonwebtoken.JwtException;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

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

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.stereotype.Component;

import java.io.Serializable;

import java.util.Date;

import java.util.HashMap;

import java.util.Map;

import java.util.function.Function;

@Component

public class JWTUtil implements Serializable {

public static final long JWT\_TOKEN\_VALIDITY = 5 \* 60 \* 60;

private static final long serialVersionUID = 234234523523L;

@Value("${jwt.secret}")

private String secretKey;

//retrieve username from jwt token

public String getUsernameFromToken(final String token) {

return getClaimFromToken(token, Claims::getSubject);

}

//retrieve expiration date from jwt token

private Date getExpirationDateFromToken(final String token) {

return getClaimFromToken(token, Claims::getExpiration);

}

private <T> T getClaimFromToken(final String token, final Function<Claims, T> claimsResolver) {

final Claims claims = getAllClaimsFromToken(token);

return claimsResolver.apply(claims);

}

//for retrieving any information from token we will need the secret key

private Claims getAllClaimsFromToken(final String token) {

Claims claims = null;

try {

claims = Jwts.parser()

.setSigningKey(secretKey)

.parseClaimsJws(token)

.getBody();

} catch (JwtException exception) {

throw new JwtException("Invalid Token");

}

return claims;

}

//check if the token has expired

private Boolean isTokenExpired(final String token) {

final Date expiration = getExpirationDateFromToken(token);

return expiration.before(new Date());

}

//generate token for user

// public String generateToken(UserDetails userDetails) {

// Map<String, Object> claims = new HashMap<>();

// return doGenerateToken(claims, userDetails.getUsername());

// }

public String generateToken(final String username) {

Map<String, Object> claims = new HashMap<>();

return doGenerateToken(claims, username);

}

//while creating the token -

//1. Define claims of the token, like Issuer, Expiration, Subject, and the ID

//2. Sign the JWT using the HS512 algorithm and secret key.

private String doGenerateToken(final Map<String, Object> claims, final String username) {

return Jwts.builder()

.setClaims(claims)

.setSubject(username)

.setIssuedAt(new Date(System.currentTimeMillis()))

.setExpiration(new Date(System.currentTimeMillis() + JWT\_TOKEN\_VALIDITY \* 1000))

.signWith(SignatureAlgorithm.HS512, secretKey)

.compact();

}

//validate token

public Boolean validateToken(final String token, final UserDetails userDetails) {

final String username = getUsernameFromToken(token);

return (username.equals(userDetails.getUsername()) && !isTokenExpired(token));

}

public String getUserName(final String header) {

return getUsernameFromToken(header.substring(7));

}

}

**Frontend Work**

Index.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="utf-8" />

    <title>My Movie Plan</title>

    <base href="/" />

    <meta name="viewport" content="width=device-width, initial-scale=1" />

    <link rel="icon" type="image/x-icon" href="favicon.ico" />

    <!-- Bootstrap Css -->

    <link

      href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css"

      rel="stylesheet"

      integrity="sha384-+0n0xVW2eSR5OomGNYDnhzAbDsOXxcvSN1TPprVMTNDbiYZCxYbOOl7+AMvyTG2x"

      crossorigin="anonymous"

    />

    <!-- Animated CSS -->

    <!-- <link

      rel="stylesheet"

      href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/4.1.1/animate.min.css"

      crossorigin="anonymous"

    /> -->

    <link rel="preconnect" href="https://fonts.gstatic.com" />

    <link

      href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500&display=swap"

      rel="stylesheet"

    />

    <link

      href="https://fonts.googleapis.com/icon?family=Material+Icons"

      rel="stylesheet"

    />

    <link rel="preconnect" href="https://fonts.gstatic.com" />

    <link

      href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500&display=swap"

      rel="stylesheet"

    />

    <link

      href="https://fonts.googleapis.com/icon?family=Material+Icons"

      rel="stylesheet"

    />

    <link rel="preconnect" href="https://fonts.gstatic.com" />

    <link

      href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500&display=swap"

      rel="stylesheet"

    />

    <link

      href="https://fonts.googleapis.com/icon?family=Material+Icons"

      rel="stylesheet"

    />

  </head>

  <body class="mat-typography">

    <app-root></app-root>

    <script

      src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/js/bootstrap.bundle.min.js"

      integrity="sha384-gtEjrD/SeCtmISkJkNUaaKMoLD0//ElJ19smozuHV6z3Iehds+3Ulb9Bn9Plx0x4"

      crossorigin="anonymous"

    ></script>

    <script type="text/javascript">

      const myCarousel = document.querySelector("#movies-carousel");

      if (myCarousel) {

        const carousel = new bootstrap.Carousel(myCarousel, {

          interval: 2000,

          wrap: false,

        });

      }

    </script>

  </body>

</html>

Main.ts

import { enableProdMode } from '@angular/core';

import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';

import { AppModule } from './app/app.module';

import { environment } from './environments/environment';

if (environment.production) {

  enableProdMode();

}

platformBrowserDynamic().bootstrapModule(AppModule)

  .catch(err => console.error(err));