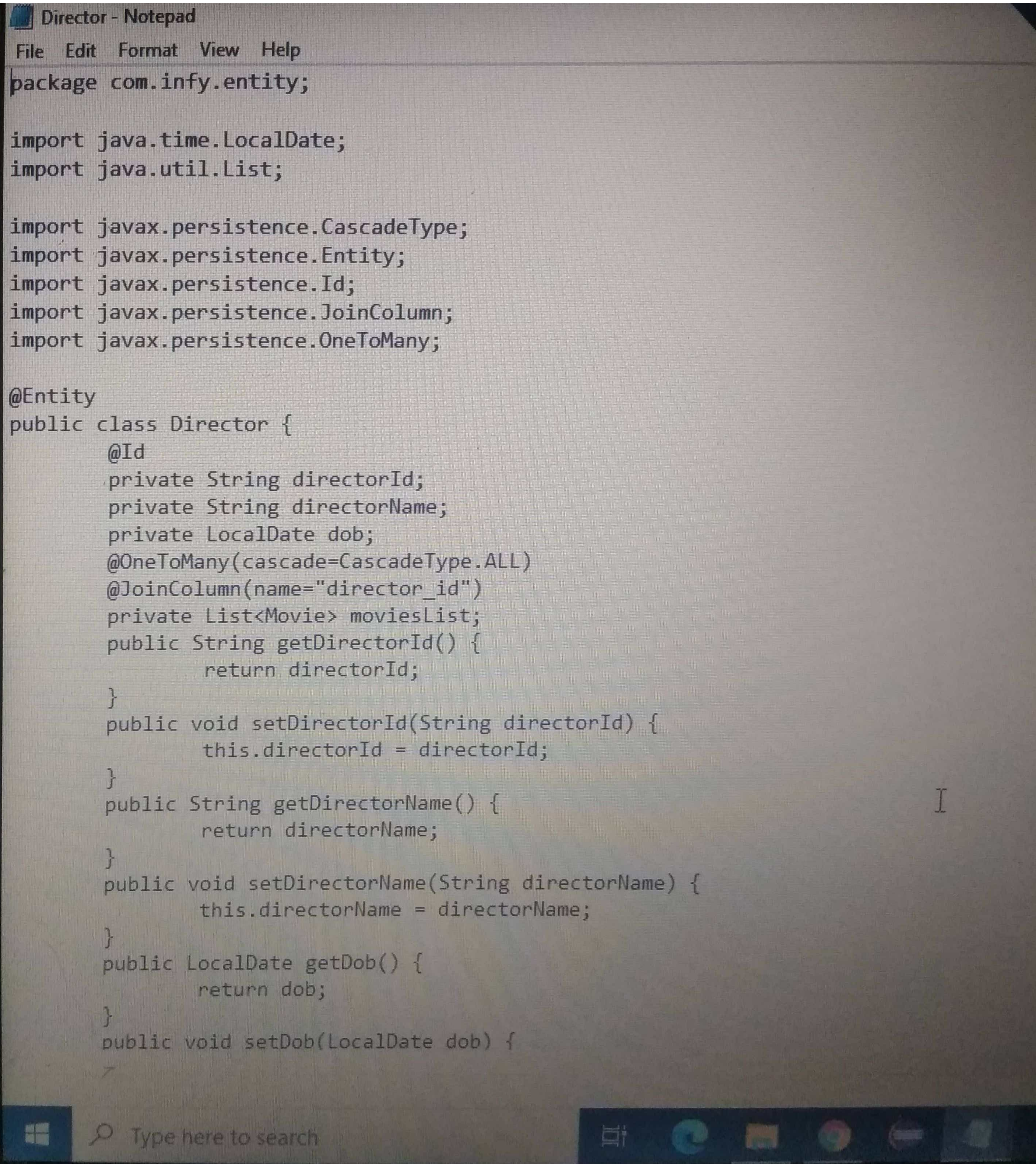
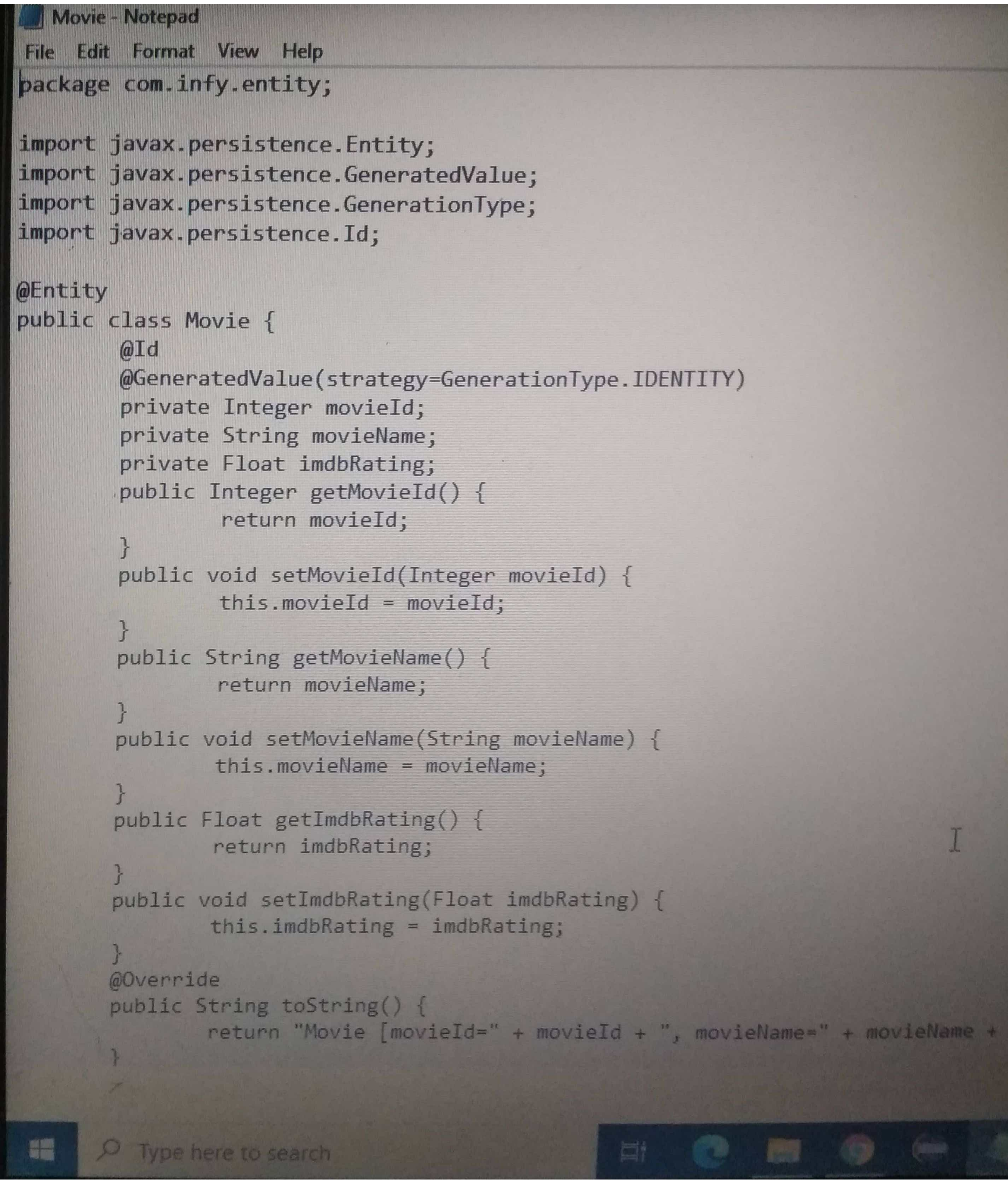


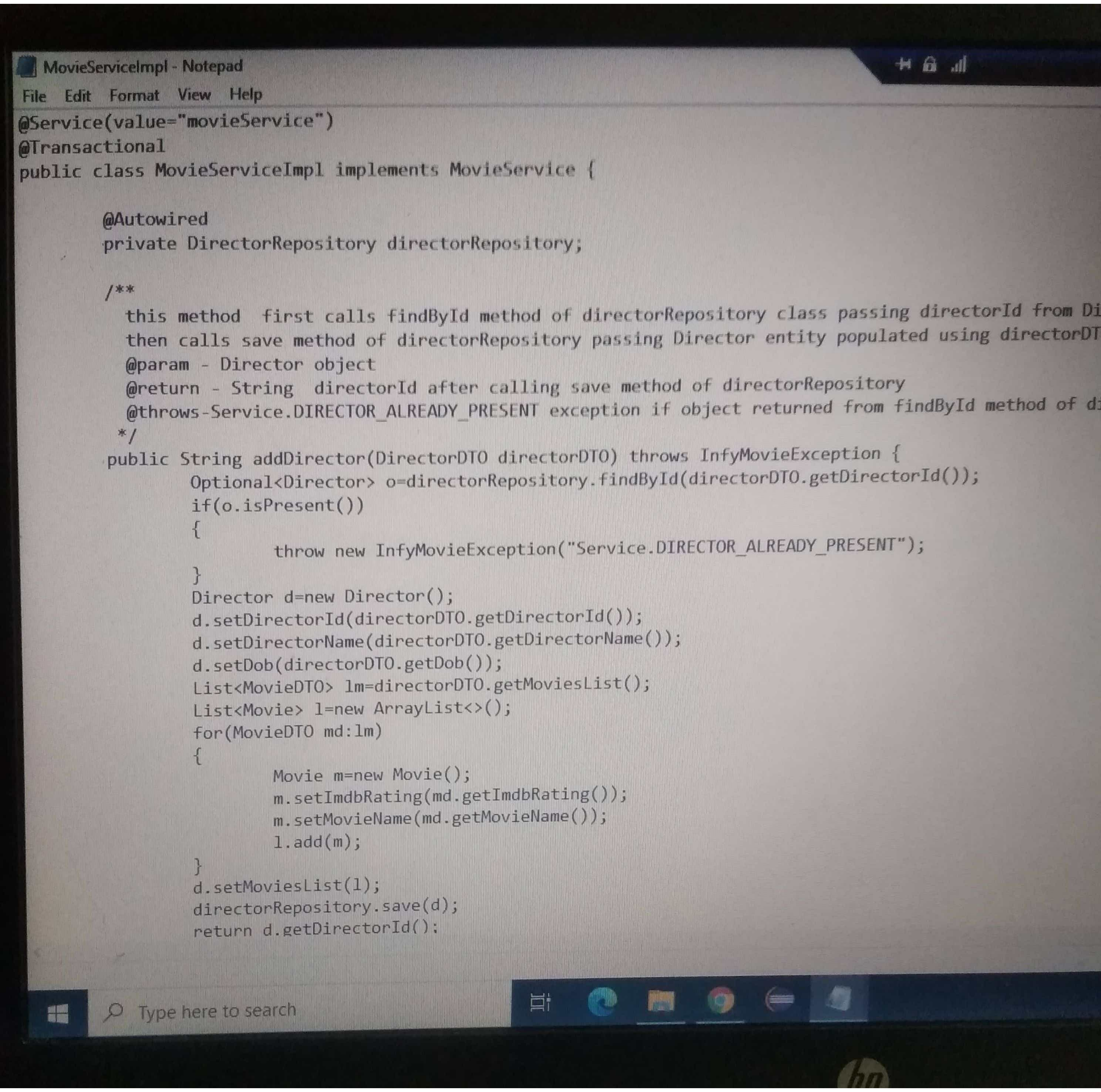
Director - Notepad

File Edit Format View Help

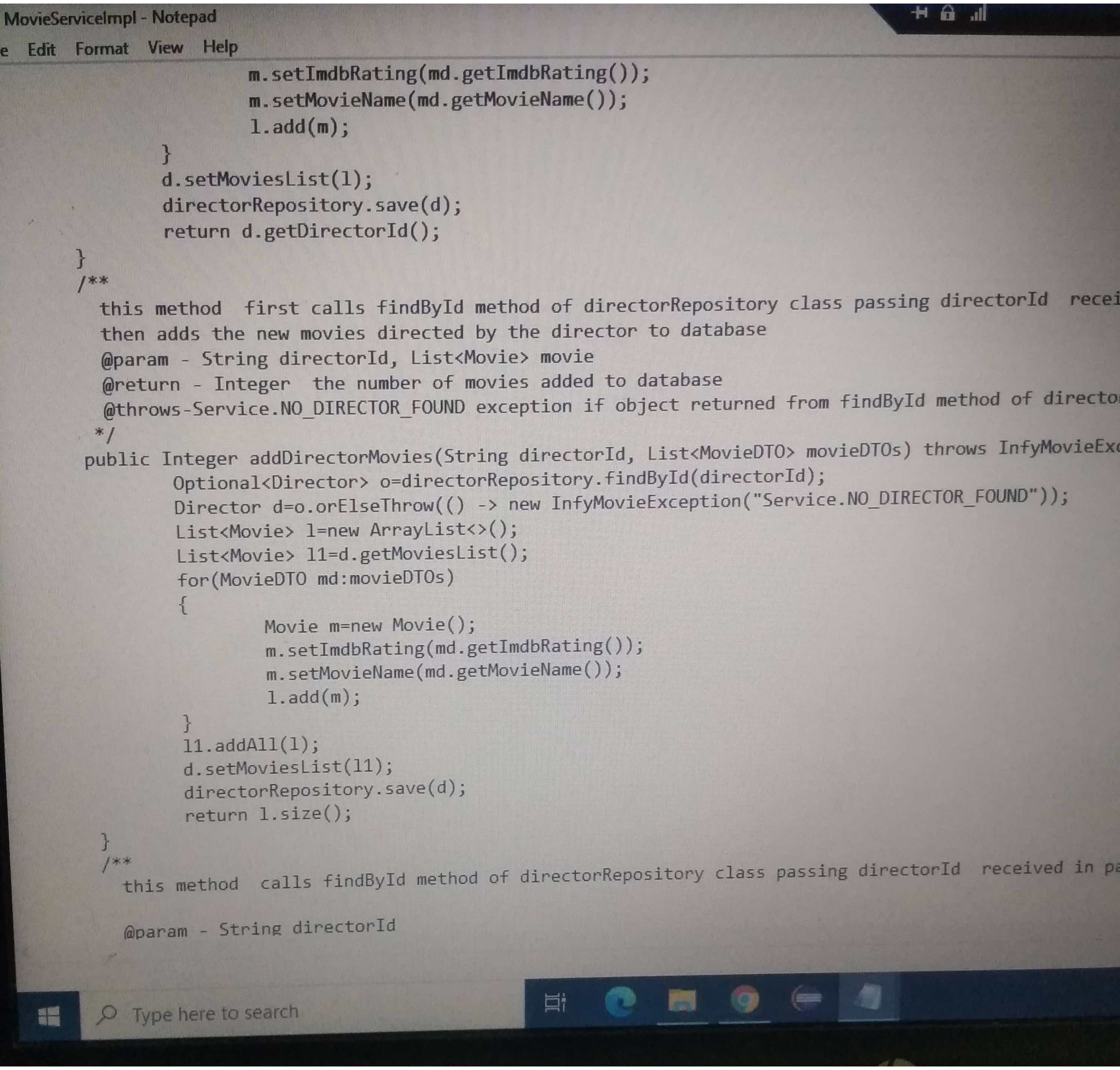
```
package com.infy.entity;

import java.time.LocalDate;
import java.util.List;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.OneToMany;

@Entity
public class Director {
	@Id
	private String directorId;
	private String directorName;
	private LocalDate dob;
	@OneToMany(cascade=CascadeType.ALL)
	@JoinColumn(name="director_id")
	private List<Movie> moviesList;
	public String getDirectorId() {
		return directorId;
	}
	public void setDirectorId(String directorId) {
		this.directorId = directorId;
	}
	public String getDirectorName() {
		return directorName;
	}
	public void setDirectorName(String directorName) {
		this.directorName = directorName;
	}
	public LocalDate getDob() {
		return dob;
	}
	public void setDob(LocalDate dob) {
```



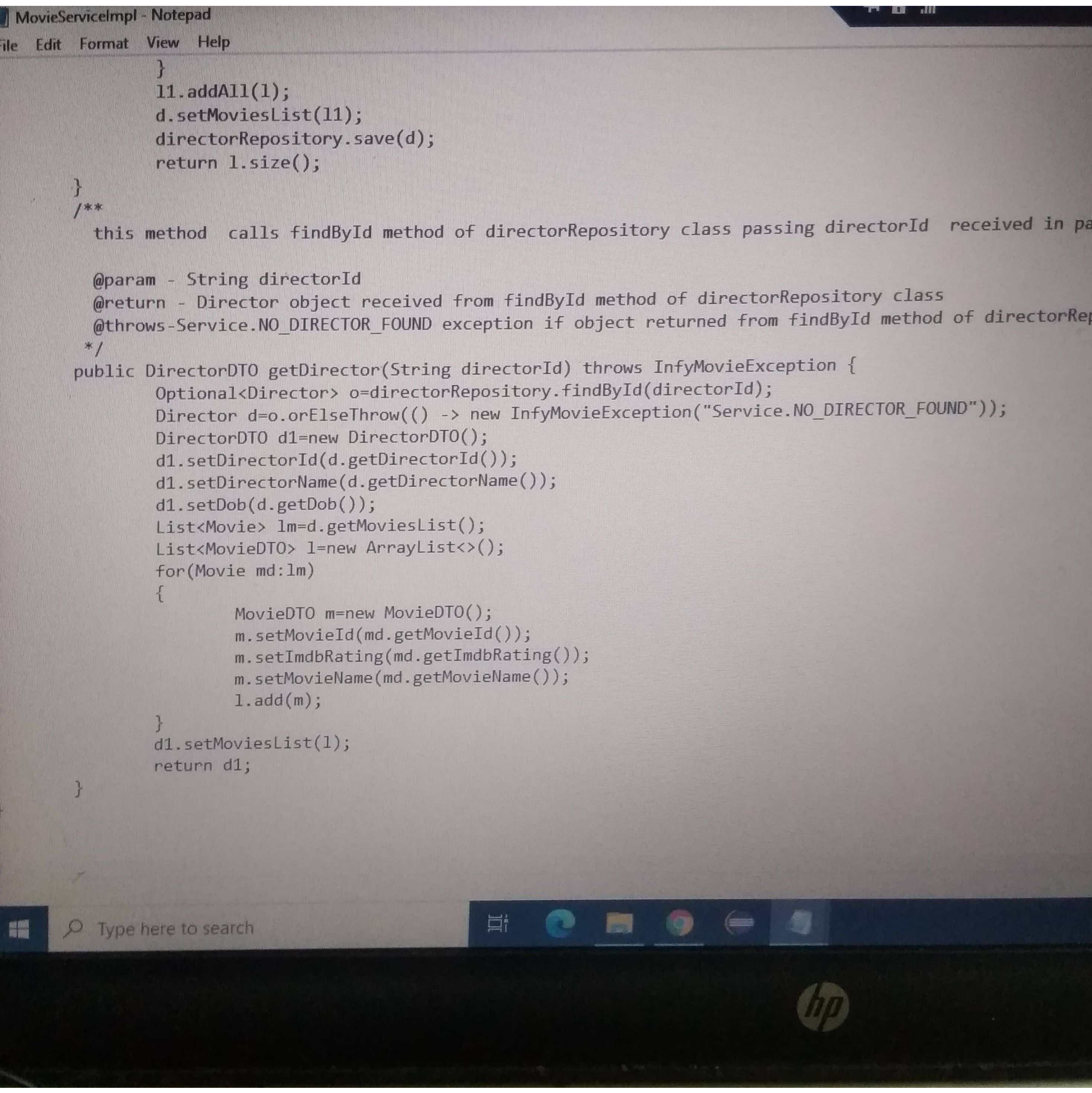
```
package com.infy.entity;

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

@Entity
public class Movie {
	@Id
	@GeneratedValue(strategy=GenerationType.IDENTITY)
	private Integer movieId;
	private String movieName;
	private Float imdbRating;
	public Integer getMovieId() {
		return movieId;
	}
	public void setMovieId(Integer movieId) {
		this.movieId = movieId;
	}
	public String getMovieName() {
		return movieName;
	}
	public void setMovieName(String movieName) {
		this.movieName = movieName;
	}
	public Float getImdbRating() {
		return imdbRating;
	}
	public void setImdbRating(Float imdbRating) {
		this.imdbRating = imdbRating;
	}
	@Override
	public String toString() {
		return "Movie [movieId=" + movieId + ", movieName=" + movieName +
	}
```



```
@Service(value="movieService")
@Transactional
public class MovieServiceImpl implements MovieService {

	@Autowired
	private DirectorRepository directorRepository;

	/**
	  this method  first calls findById method of directorRepository class passing directorId from Di
	  then calls save method of directorRepository passing Director entity populated using directorDT
	  @param - Director object
	  @return - String  directorId after calling save method of directorRepository
	  @throws-Service.DIRECTOR_ALREADY_PRESENT exception if object returned from findById method of di
	 */
	public String addDirector(DirectorDTO directorDTO) throws InfyMovieException {
		Optional<Director> o=directorRepository.findById(directorDTO.getDirectorId());
		if(o.isPresent())
		{
			throw new InfyMovieException("Service.DIRECTOR_ALREADY_PRESENT");
		}
		Director d=new Director();
		d.setDirectorId(directorDTO.getDirectorId());
		d.setDirectorName(directorDTO.getDirectorName());
		d.setDob(directorDTO.getDob());
		List<MovieDTO> lm=directorDTO.getMoviesList();
		List<Movie> l=new ArrayList<>();
		for(MovieDTO md:lm)
		{
			Movie m=new Movie();
			m.setImdbRating(md.getImdbRating());
			m.setMovieName(md.getMovieName());
			l.add(m);
		}
		d.setMoviesList(l);
		directorRepository.save(d);
		return d.getDirectorId();
```



```
                m.setImdbRating(md.getImdbRating());
                m.setMovieName(md.getMovieName());
                l.add(m);
        }
        d.setMoviesList(l);
        directorRepository.save(d);
        return d.getDirectorId();
    }
    /**
     this method  first calls findById method of directorRepository class passing directorId  recei
     then adds the new movies directed by the director to database
     @param - String directorId, List<Movie> movie
     @return - Integer  the number of movies added to database
     @throws-Service.NO_DIRECTOR_FOUND exception if object returned from findById method of directo
    */
    public Integer addDirectorMovies(String directorId, List<MovieDTO> movieDTOs) throws InfyMovieExc
        Optional<Director> o=directorRepository.findById(directorId);
        Director d=o.orElseThrow(() -> new InfyMovieException("Service.NO_DIRECTOR_FOUND"));
        List<Movie> l=new ArrayList<>();
        List<Movie> l1=d.getMoviesList();
        for(MovieDTO md:movieDTOs)
        {
                Movie m=new Movie();
                m.setImdbRating(md.getImdbRating());
                m.setMovieName(md.getMovieName());
                l.add(m);
        }
        l1.addAll(l);
        d.setMoviesList(l1);
        directorRepository.save(d);
        return l.size();
    }
    /**
     this method  calls findById method of directorRepository class passing directorId  received in pa

     @param - String directorId
```



```
        }
        l1.addAll(l);
        d.setMoviesList(l1);
        directorRepository.save(d);
        return l.size();
}
/**
 this method  calls findById method of directorRepository class passing directorId  received in pa

 @param - String directorId
 @return - Director object received from findById method of directorRepository class
 @throws-Service.NO_DIRECTOR_FOUND exception if object returned from findById method of directorRep
 */
public DirectorDTO getDirector(String directorId) throws InfyMovieException {
        Optional<Director> o=directorRepository.findById(directorId);
        Director d=o.orElseThrow(() -> new InfyMovieException("Service.NO_DIRECTOR_FOUND"));
        DirectorDTO d1=new DirectorDTO();
        d1.setDirectorId(d.getDirectorId());
        d1.setDirectorName(d.getDirectorName());
        d1.setDob(d.getDob());
        List<Movie> lm=d.getMoviesList();
        List<MovieDTO> l=new ArrayList<>();
        for(Movie md:lm)
        {
                MovieDTO m=new MovieDTO();
                m.setMovieId(md.getMovieId());
                m.setImdbRating(md.getImdbRating());
                m.setMovieName(md.getMovieName());
                l.add(m);
        }
        d1.setMoviesList(l);
        return d1;
}
```