**General Development:**

1. Describe the process of creating a new Datasource (Sqlite in this example) using only configuration properties.

```
spring.datasource.url=jdbc:sqlite:/app/chinook.db
spring.datasource.driverClassName=org.sqlite.JDBC
spring.datasource.username=sa
spring.datasource.sqlite.password=sa
spring.jpa.database-platform=com.wabi.challengeone.SQLiteDialect
hibernate.dialect=com.wabi.challengeone.SQLiteDialect
hibernate.hbm2ddl.auto=create-drop
hibernate.show_sql=true
# A Dialect Class has to be declared with data types, foreign keys,
tables, primary key, etc definitions
```

2. What's a DTO? Why would you use it?

   It’s an object that is used to encapsulate data, and send it from one subsystem of an application to another.

```
// This is using Lombok
package com.wabi.challengeone.dto;
import lombok.Builder;
import lombok.Data;
import lombok.extern.java.Log;
@Data
@Builder
@Log
public class FilmDTO {
   private int Year;
   private int Score;
   private String Title;
}
```

3. Define REST API and which are the main HTTP methods

   Is an architectural style for an application program interface (API) that uses HTTP requests to access and use data

| HTTP Method | Idempotent | Safe |
|---|---|---|
| OPTIONS | yes | yes |
| GET | yes | yes |
| HEAD | yes | yes |
| PUT | yes | no |
| POST | no | no |
| DELETE | yes | no |
| PATCH | no | no |

4. Could you describe the process of creating a new annotation in Springboot?

Create an Interface with the @Target method and @Retention policy and the @Aspect class that what happens and when around the execution process (after, before, around…). Then write the annotation before the Class or Method defitinion.

5. Describe what's Spring AOP Pattern.

   Spring AOP omplements Object-Oriented Programming (OOP) by providing another way of thinking about program structure. In addition to classes, AOP gives you *aspects*. Aspects enable modularization of concerns such as transaction management that cut across multiple types and objects.

6. Explain the main difference of this two methods:

```
public Mono<String> requestExternalWebflux() {
    WebClient client = WebClient.builder().build();
    return client.get()
            .uri(URI.create("https://……...."))
            .retrieve()
            .bodyToMono(String.class)
            .doOnNext(logger::info);



}

public String requestExternalResttemplate() {
    RestTemplate restTemplate = new RestTemplate();
    String returnable = restTemplate
            .exchange
                    (       "https://……....",
                            HttpMethod.GET,
                            RequestEntity.EMPTY,
                            String.class
                    ).getBody();
    logger.info(returnable);
    return returnable;
}
```

   *RestTemplate* uses the Java Servlet API, which is based on the thread-per-request model. This means that the thread will block until the web client receives the response

   *WebClient* uses an asynchronous, non-blocking solution provided by the Spring Reactive framework.

**Development methodologies:**

1. Describe whats TDD (Test Driven Development): is a process of developing software where a test is written prior to writing code.
2. Describe whats DDD (Domain Driven Development): approach to development that connects the implementation to an evolving model; placing the focus of the project

on the core domain (sphere of knowledge), the logic behind it, and forces collaboration between technical and nontechnical parties to improve the model

3. Describe whats BDD (Behavioral Driven Development): is a way of combining business requirements with code and allows you to understand the behavior of the system from a business/end-user perspective

**Pipeline:**

1. Please, describe what this Dockerfile do:

```
FROM openjdk:12-alpine
RUN apk add bash
WORKDIR /app
COPY target/app.jar app.jar
RUN addgroup -S spring && adduser -S spring -G spring
RUN chown -R spring:spring /app
RUN chmod 755 /app
EXPOSE 8080
USER spring:spring
ENTRYPOINT ["java", "-jar", "/app/app.jar"]
```