

YT Project Documentation

Introduction:

Since I have never coded with the Spring Boot framework before, I was able to quickly learn how to implement a Spring Boot program over the last few days. My implementation isn't exactly what was asked for as I don't store XML messages in my databases. Instead I store the title and url of a video within my data table shown in the picture below.

```
queue=# SELECT * FROM yt;
```

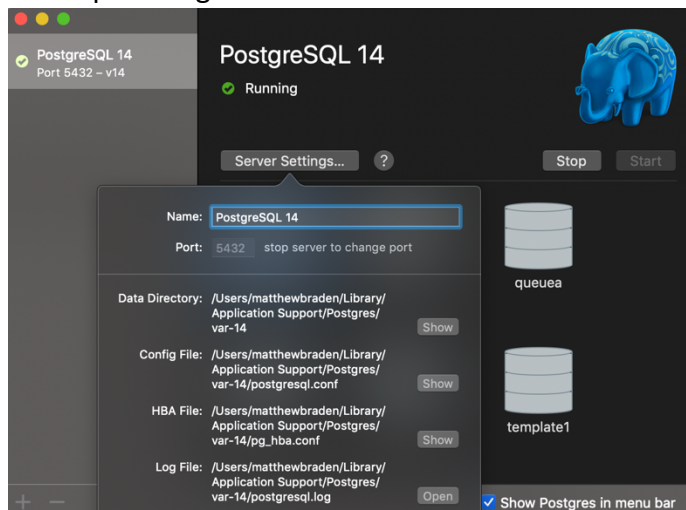
id	title	url
1	FUNDAMENTALS OF TELECOM DOMAIN	https://www.youtube.com/watch?v=vYHh9vNIwIU
2	Telecom Base Station Materials: A 3D Walkthrough	https://www.youtube.com/watch?v=-WyyrKbUruA
3	Telecom - Industry Overview	https://www.youtube.com/watch?v=xRzLqZpQDE
4	Basic Telecom Concepts	https://www.youtube.com/watch?v=xRFe9jWY0hg
5	Telecom wiring color code	https://www.youtube.com/watch?v=1QXBdo-kXo8
6	TELECOM 2.0 エントロピー (ENTROPIJA) FULL ALBUM	https://www.youtube.com/watch?v=f7dhzrQzIMU
7	Power Glove - Telecom	https://www.youtube.com/watch?v=XGmeq2E4LLQ
8	Telecom Pole Guying Installation - Sidewalk Guy	https://www.youtube.com/watch?v=Q75LJz7H83g
9	The Digital Telco: Examining Trends in the Telecom Industry and Their Impact on Customers	https://www.youtube.com/watch?v=xXfDv1YBIing
10	What a \$26 B deal between telecom giants means for 5G in Canada	

In my program I was able to implement the Singleton Design Pattern. The Singleton Pattern that I implemented involves using Singleton Beans as well as Autowired Singletons to communicate with the two databases.

The messages produced by the solution appears as an Object with the attributes of "id" as a Long, "title" as a String and "url" as a String.

How to compile the source code / How to execute the program:

1. Install PostgreSQL on your computer
2. Start up a PostgreSQL server and make sure that the port being used is 5432



3. Enter the psql server and create a database for both QueueA and QueueB

```
~ — psql -p5432 matthewbraden
Last login: Thu Feb  3 23:38:23 on ttys000
Matthews-MBP:~ matthewbraden$ /Applications/Postgres.app/Contents/Versions/14/bin/psql -p5432 "matthewbraden"
psql (14.1)
Type "help" for help. src
matthewbraden=# CREATE DATABASE QueueA;
CREATE DATABASE
matthewbraden=# CREATE DATABASE QueueB;
CREATE DATABASE
matthewbraden=#
```

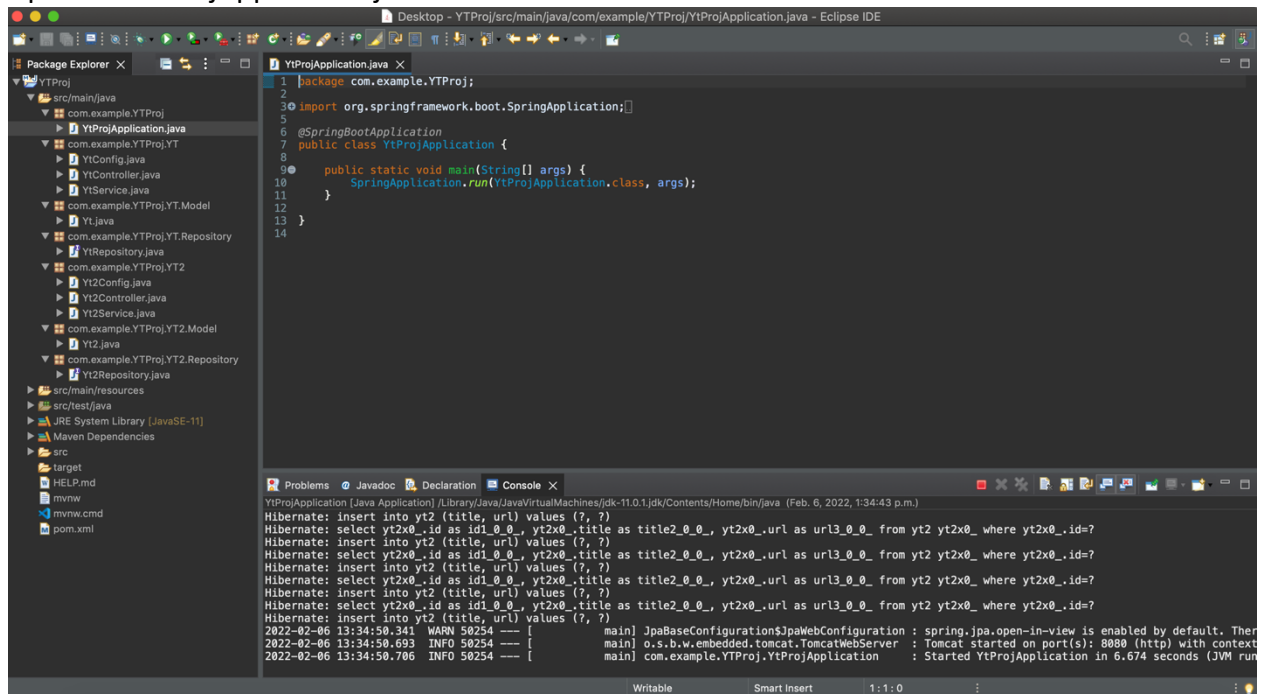
4. The next thing to do is to grant all privileges to both QueueA and QueueB

```
~ — psql -p5432 matthewbraden
List of databases
+-----+-----+-----+-----+-----+-----+
| Name      | Owner      | Encoding | Collate  | Ctype    | Access privileges |
+-----+-----+-----+-----+-----+-----+
| matthewbraden | matthewbraden | UTF8     | en_US.UTF-8 | en_US.UTF-8 |                    |
| postgres      | postgres      | UTF8     | en_US.UTF-8 | en_US.UTF-8 |                    |
| queuea         | matthewbraden | UTF8     | en_US.UTF-8 | en_US.UTF-8 |                    |
| queueb         | matthewbraden | UTF8     | en_US.UTF-8 | en_US.UTF-8 |                    |
| student        | matthewbraden | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =Tc/matthewbraden +
|                |                |          |             |             | matthewbraden=Ctc/matthewbraden +
| template0      | postgres      | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +
|                |                |          |             |             | postgres=Ctc/postgres +
| template1      | postgres      | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +
|                |                |          |             |             | postgres=Ctc/postgres +
(7 rows)

matthewbraden=# GRANT ALL PRIVILEGES ON DATABASE "queuea" TO matthewbraden;
GRANT
matthewbraden=# GRANT ALL PRIVILEGES ON DATABASE "queuea" TO postgres;
GRANT
matthewbraden=# GRANT ALL PRIVILEGES ON DATABASE "queueb" TO matthewbraden;
GRANT
matthewbraden=# GRANT ALL PRIVILEGES ON DATABASE "queueb" TO postgres;
GRANT
matthewbraden=#
```

5. Open the Eclipse IDE and import the YTProj project into the workspace

6. Open the YTProjApplication.java file and run the code



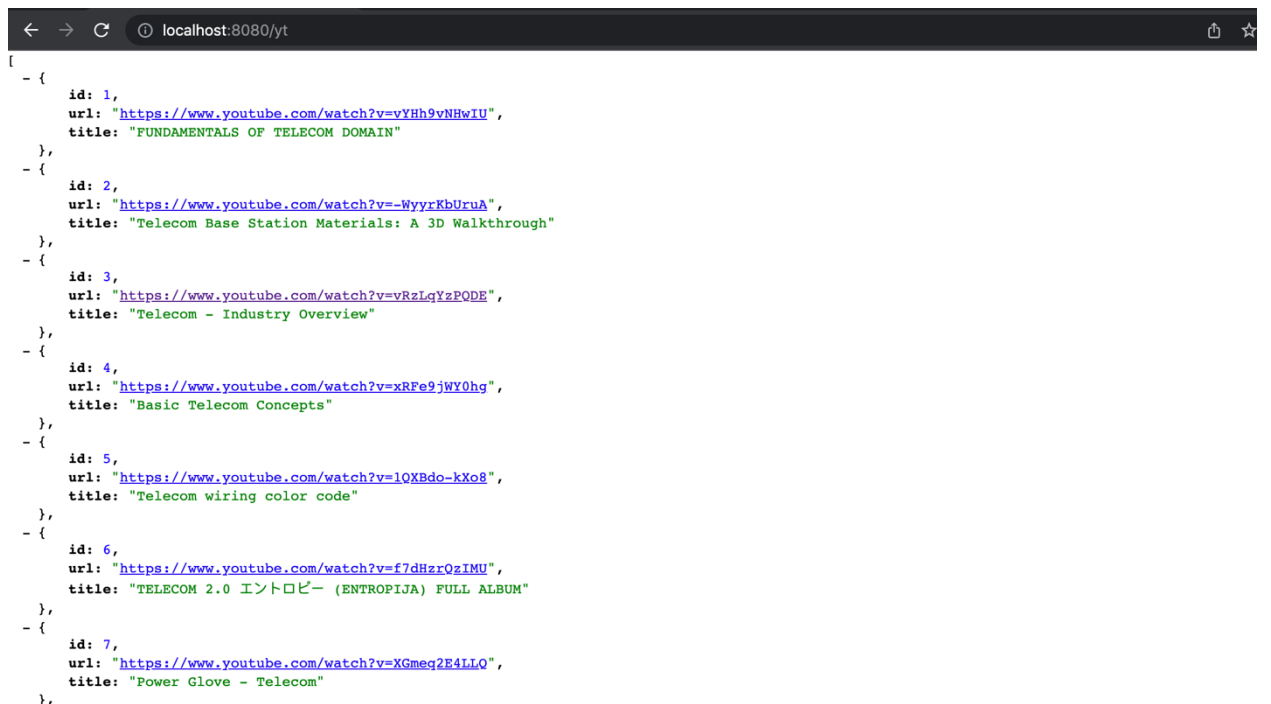
The screenshot shows the Eclipse IDE with the YTProjApplication.java file open. The code is as follows:

```
1 package com.example.YTProj;
2
3 import org.springframework.boot.SpringApplication;
4
5 @SpringBootApplication
6 public class YTProjApplication {
7     public static void main(String[] args) {
8         SpringApplication.run(YTProjApplication.class, args);
9     }
10 }
11
12
13
14
```

The console output shows the application running successfully, with the following log messages:

```
2022-02-06 13:34:50.341 WARN 50254 --- [main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Ther
2022-02-06 13:34:50.693 INFO 50254 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context
2022-02-06 13:34:50.706 INFO 50254 --- [main] com.example.YTProj.YTProjApplication : Started YTProjApplication in 6.674 seconds (JVM run
```

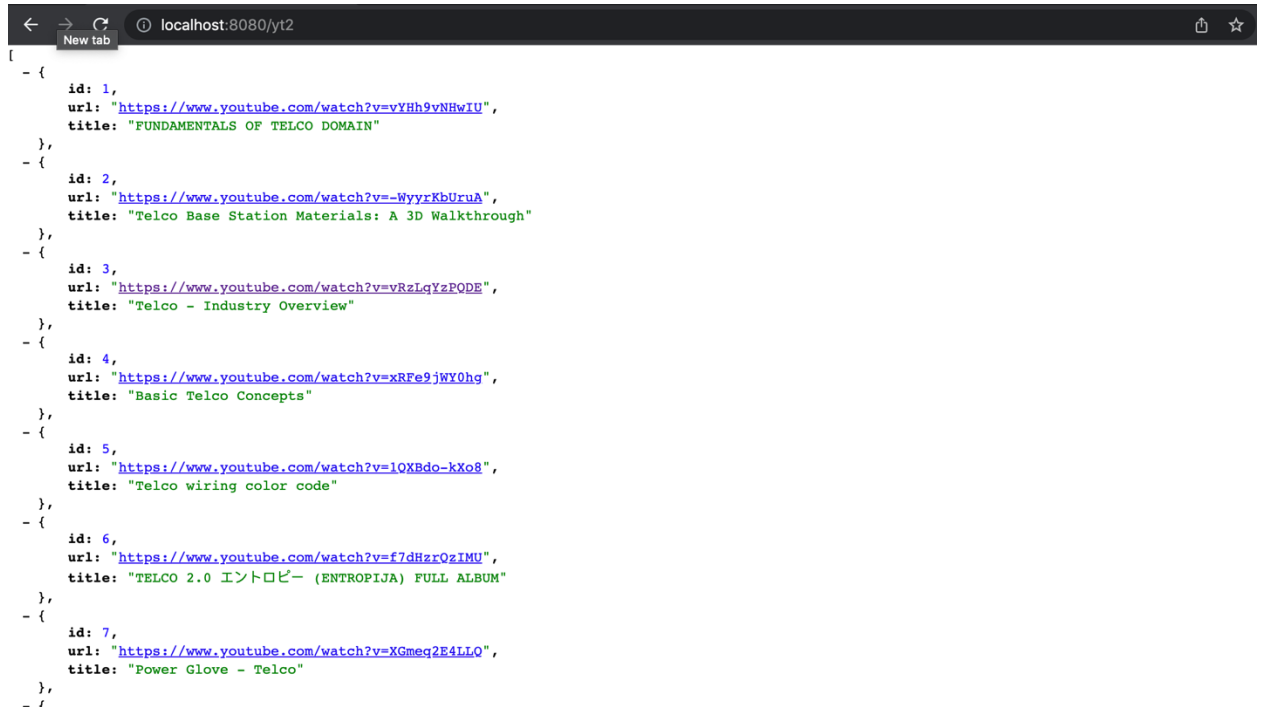
7. Open your web browser to the address <http://localhost:8080/yt> to see the data that is stored in QueueA



The screenshot shows a web browser displaying the data stored in QueueA. The data is as follows:

```
[
  {
    id: 1,
    url: "https://www.youtube.com/watch?v=vYHh9vNHwIU",
    title: "FUNDAMENTALS OF TELECOM DOMAIN"
  },
  {
    id: 2,
    url: "https://www.youtube.com/watch?v=-WyyrKbUruA",
    title: "Telecom Base Station Materials: A 3D Walkthrough"
  },
  {
    id: 3,
    url: "https://www.youtube.com/watch?v=vRzLqYzPODE",
    title: "Telecom - Industry Overview"
  },
  {
    id: 4,
    url: "https://www.youtube.com/watch?v=xRFe9jWY0hg",
    title: "Basic Telecom Concepts"
  },
  {
    id: 5,
    url: "https://www.youtube.com/watch?v=IQXBdo-kXo8",
    title: "Telecom wiring color code"
  },
  {
    id: 6,
    url: "https://www.youtube.com/watch?v=f7dHzrOzIMU",
    title: "TELECOM 2.0 エントロピー (ENTROPIJA) FULL ALBUM"
  },
  {
    id: 7,
    url: "https://www.youtube.com/watch?v=XGmeq2E4LLQ",
    title: "Power Glove - Telecom"
  }
]
```

8. Open your web browser to the address <http://localhost:8080/yt2> to see the data that is stored in QueueB



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/yt2'. The page content is a JSON array of seven video objects. Each object contains an 'id', a 'url' (a YouTube link), and a 'title'. The titles are related to Telco and industry concepts.

```
[
  - {
    id: 1,
    url: "https://www.youtube.com/watch?v=vYHh9vNRwIU",
    title: "FUNDAMENTALS OF TELCO DOMAIN"
  },
  - {
    id: 2,
    url: "https://www.youtube.com/watch?v=-WyyrKbUruA",
    title: "Telco Base Station Materials: A 3D Walkthrough"
  },
  - {
    id: 3,
    url: "https://www.youtube.com/watch?v=vRzLqYzPODE",
    title: "Telco - Industry Overview"
  },
  - {
    id: 4,
    url: "https://www.youtube.com/watch?v=xRFe9jWY0hg",
    title: "Basic Telco Concepts"
  },
  - {
    id: 5,
    url: "https://www.youtube.com/watch?v=lQXBdo-kXo8",
    title: "Telco wiring color code"
  },
  - {
    id: 6,
    url: "https://www.youtube.com/watch?v=f7dHzrOzIMU",
    title: "TELCO 2.0 エントロピー (ENTROPIJA) FULL ALBUM"
  },
  - {
    id: 7,
    url: "https://www.youtube.com/watch?v=XGmeq2E4LLQ",
    title: "Power Glove - Telco"
  },
  - ]
```

Conclusion:

Looking back at my implementation, I wish I could've had a bit more time to try to learn the Spring Boot framework. Since I was not able to complete the YT Project as it was intended to, I would consider learning how to send and retrieve XML messages to a JMS database.