**Setup Instructions for Java Project**

1. **Development Environment**

* **IDE -** IntelliJ IDEA
* **JDK-** 10.0.2
* **Performance Tool :** JMeter
* **Cache** – Redis

redis.port: 8083

redis.host: https://localhost

redis.password: XXXXXXXX

redis.cachingTime: 5

* **Database** – MySQL

url: jdbc:mysql://localhost:3306/sakila?useSSl=false&serverTimeZone=UTC

username: XXXXXXXX

password: XXXXXXXX

Table structure:

CREATE TABLE `employees` (

`emp\_id` varchar(10) NOT NULL,

`first\_name` varchar(10) DEFAULT NULL,

`last\_name` varchar(20) NOT NULL,

`salary` int NOT NULL,

`gender` varchar(6) NOT NULL,

`dept` varchar(20) DEFAULT NULL,

PRIMARY KEY (`emp\_id`)

)

1. **Code Architecture Details**

* **src/main/java**: Contains source code.
* **src/test/java**: Contains test code.
* **pom.xml:** Maven build configuration file.
* **Rest Endpoint :**

Search : http://localhost:8081/employee/search

Insert : http://localhost:8081/employee/insert

Update : http://localhost:8081/employee/update

Delete : <http://localhost:8081/employee/delete/2>

Json for Search, Insert and Update

[

{

emp\_id: "1",

firstname: "John",

lastname: "Trump",

salary : 2500,

gender : "Male"

dept : "CmpSc"

},

{

emp\_id : "2",

firstname : "Mohn",

lastname : "Kansal",

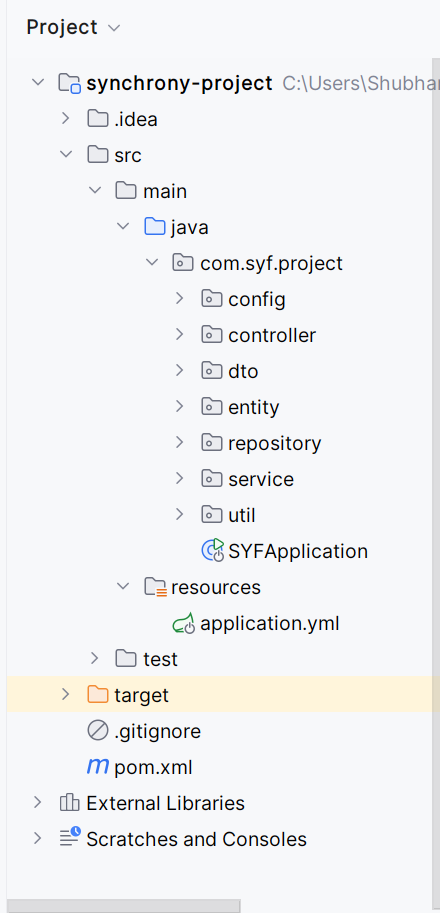
salary : 50000,

gender: "Male",

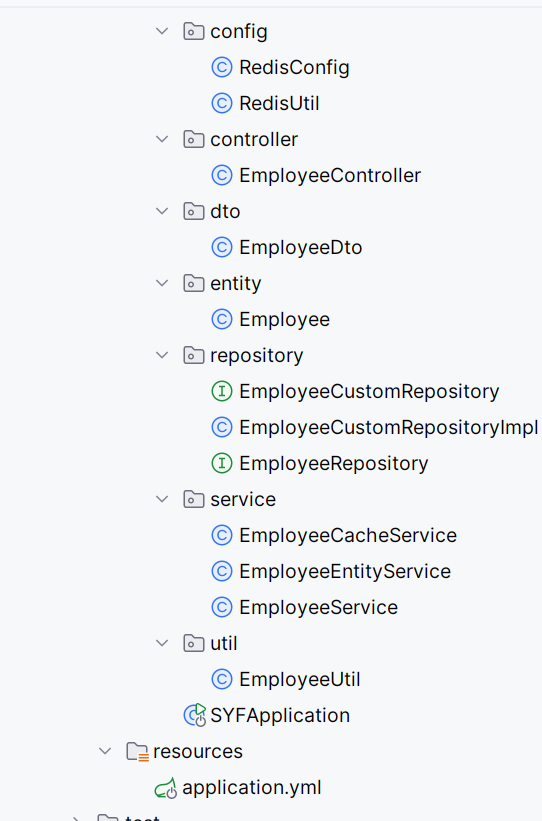
dept : "Marketing"

}

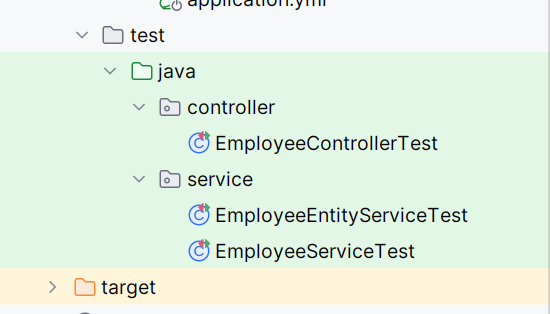
]



* **Controller Layer**: Created Rest Controller which handles HTTP Requests.
* **Service Layer**: Contains business logic for performing CURD operations using cache.
* **Repository Layer**: Helps us to interact with database using JPA.
* **Domain Layer**: Created entities and DTO for transferring of data.
* **Exception Handling**: Used try and catch for proper handling of errors.



* **Junit**: Created test classes for unit testing.



**Performance Reports**

* Run Performance with the help of JMeter

**Sequential Execution**:

* **Time Taken**: 10,000 milliseconds (10.000 seconds).
* **Throughput**: 100 tasks / 10.000 seconds = 10 tasks per second.
* **Latency**: While executing sequentially, each task takes around 100ms.

**Parallel Execution**:

* **Time Taken**: 2,000 milliseconds (2.00 seconds).
* **Throughput**: 100 tasks / 2.00 seconds = 50 tasks per second.
* **Latency**: While executing parallely, each task takes around 20ms.