Instruction of running the project and testing step by step

Environment:

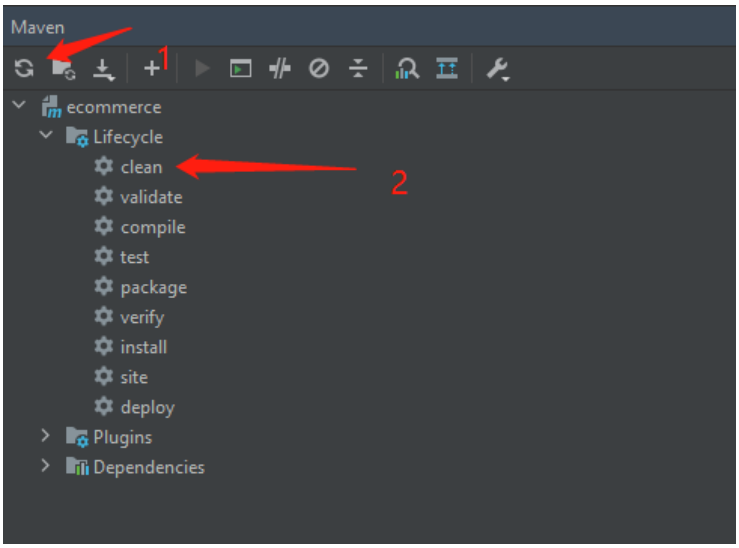
Java 1.8 or Java 16 (java 17 is not working on M1 Mac for some Mockito methods)

Jmeter latest binary zip file on https://jmeter.apache.org/download_jmeter.cgi

Postgresql: on AWS for now, will under maintenance until June

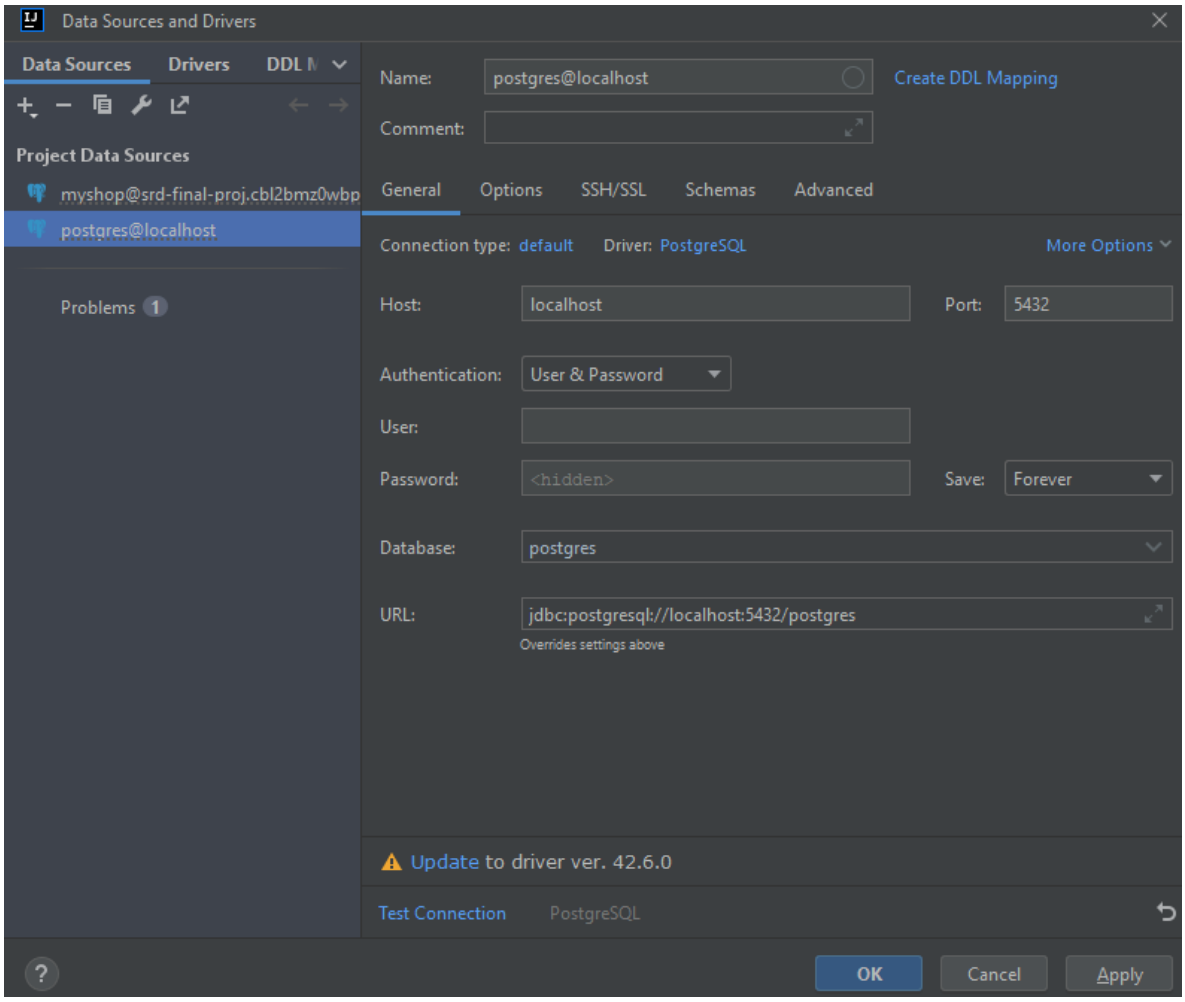
How to get the project start-up and running:

1. Clone the repo from GitHub: https://github.com/Leafeon2233/spring-restapi-ecommerce `main branch`
2. Open the repo in IntelliJ and run maven reload, maven clean.



3. (Optional) add database in IntelliJ

   In IntelliJ File-> new -> datasource -> Postgresql



Fill up these blanks.

Host: srd-final-proj.cbl2bmz0wbpc.us-east-1.rds.amazonaws.com

User: std

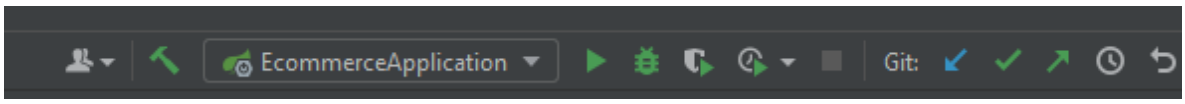
Password: admin12345

Database: myshop

Please update the driver if needed.

Click Test Connection and OK to see the database is connected and then you can see the tables.

4. Run the project



Find the Ecommerce-Application and run. As config, it should be running on port 3000

5. Run API tests:

In `src/test/java/com/rene/ecommerce/apitest`

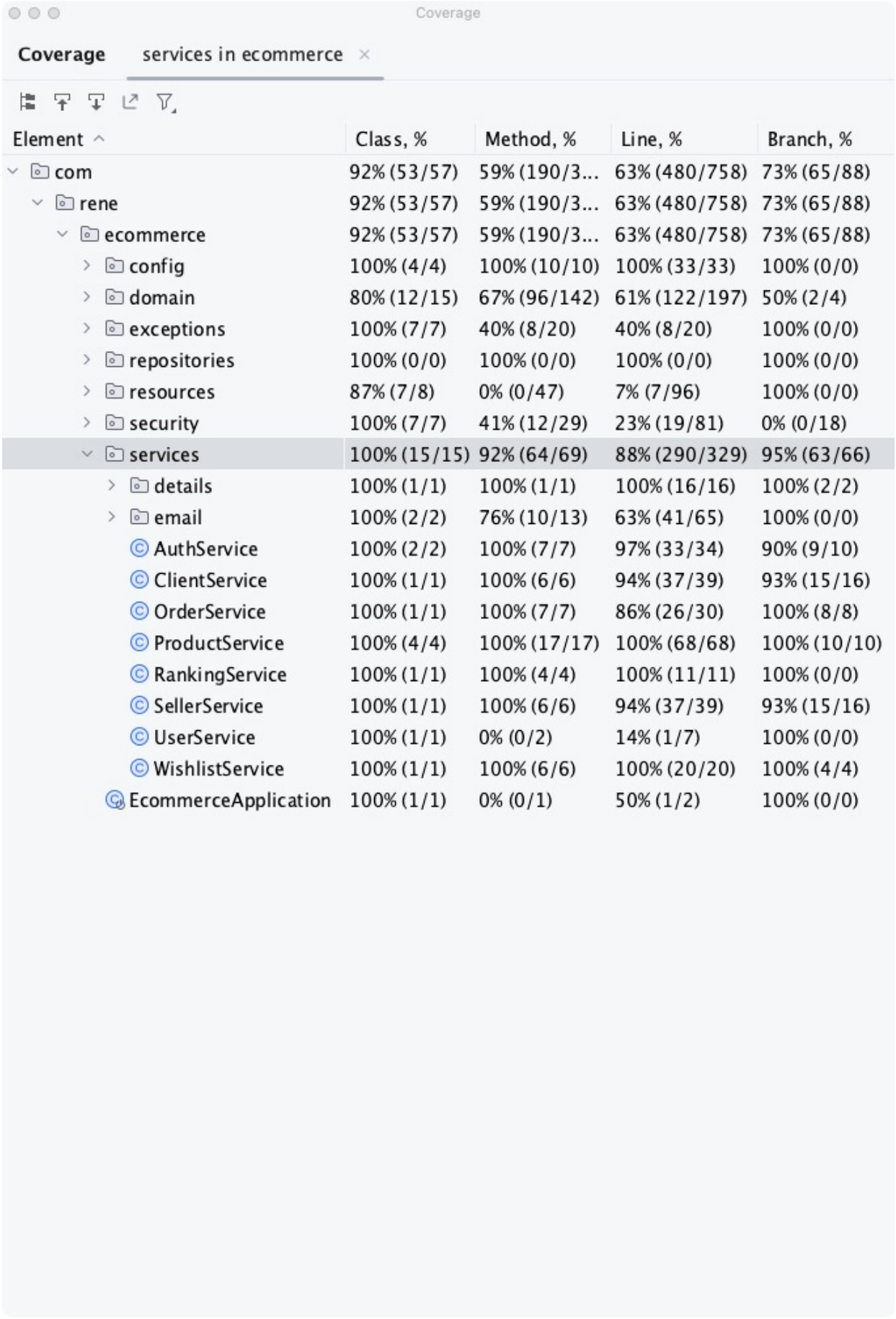
You can pick up specific test case you want to run.

Please do not run the whole directory at once, there might be some conflicts between some apis. For example, if you delete a existed client and you cannot send a buy product request afterwards.

6. Run white-box tests(can be tested without running the project):

   In `src/test/java/com/rene/ecommerce/services`

   We have tested every services and reached **95%** of BC in services.



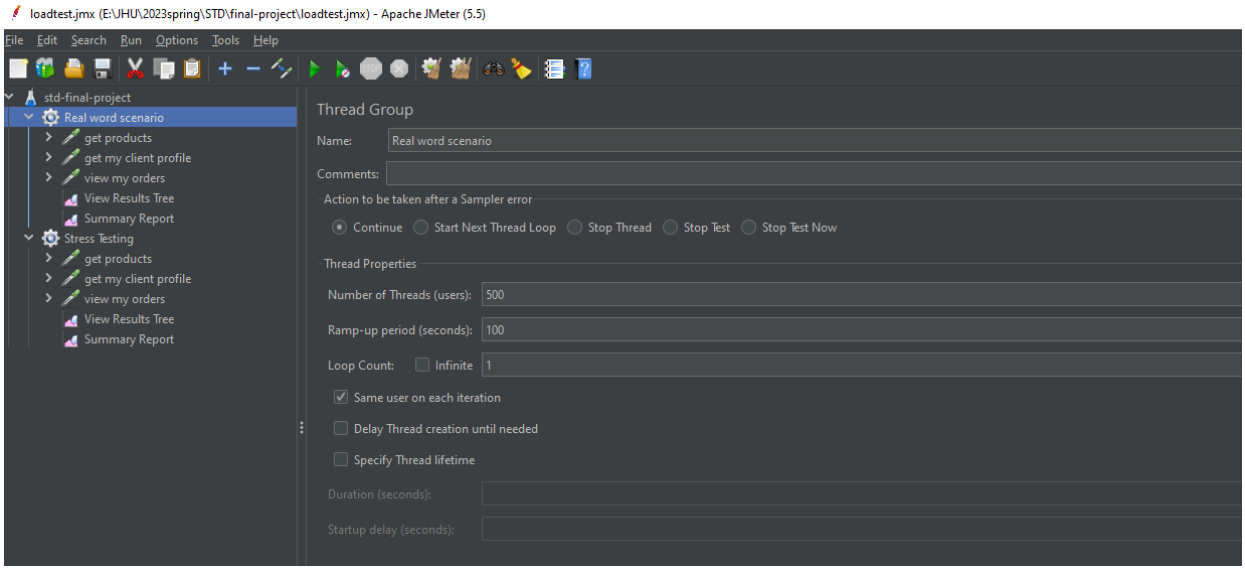
7. Run load test.

   In `src/test/java/com/rene/ecommerce` find the `loadtest.jmx`

Run the `Jmeter`

- After you download the zip file, extract it into your desired folder.
- Set the `JMETER_HOME` environment variable to point to the extracted directory.
- Navigate to the `bin` directory inside the `JMETER_HOME` directory.
- Run `jmeter.bat` (Windows) or `jmeter.sh` (Linux/Mac).

Open the `loadtest.jxm` file:



Result:

For normal scenario:

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
get products	500	740	45	6272	1645.70	0.00%	5.0/sec	9.47	1.53	1936.0
get my client profile	500	532	53	6359	1288.66	0.00%	5.1/sec	3.00	1.54	604.0
view my orders	500	280	44	5617	774.72	0.00%	5.3/sec	2.32	1.65	445.0
TOTAL	1500	517	44	6359	1300.63	0.00%	15.0/sec	14.57	4.58	995.0

Three most used request get 517ms latency in average which is very acceptable.

For stress testing, 500 users sending request in 1 seconds.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
get products	500	2572	171	5461	1214.81	0.00%	77.6/sec	146.63	23.63	1936.0
get my client profile	500	4981	1013	7095	980.50	0.00%	40.7/sec	23.99	12.32	604.0
view my orders	500	5150	1998	8170	695.82	0.00%	35.9/sec	15.60	11.11	445.0
TOTAL	1500	4234	171	8170	1536.13	0.00%	92.2/sec	89.61	28.19	995.0

The average latency is 4.23 seconds.

But we believe the bottleneck of the load testing may due to database capability. And different deployment methods may cause different performances.

End

Thanks for the semester!