Name: Marc Teixeira

Date: 08/07/2020

Course: CST-235

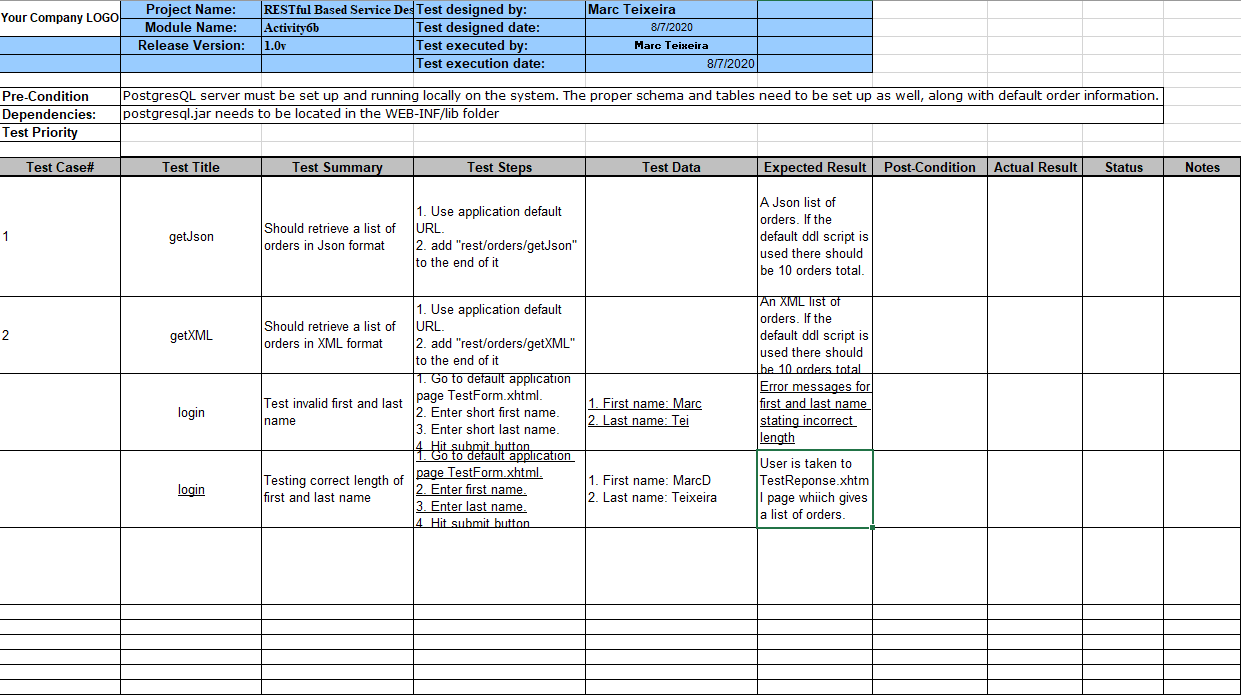
Instructor: Charbel Elkhoury

Git Link: <https://github.com/MTeixeiraGCU/CST-235/tree/master/WeekSix>

RESTful Based Service Design and Testing Activity

Part 1:

Image from mapped REST API excel sheet:

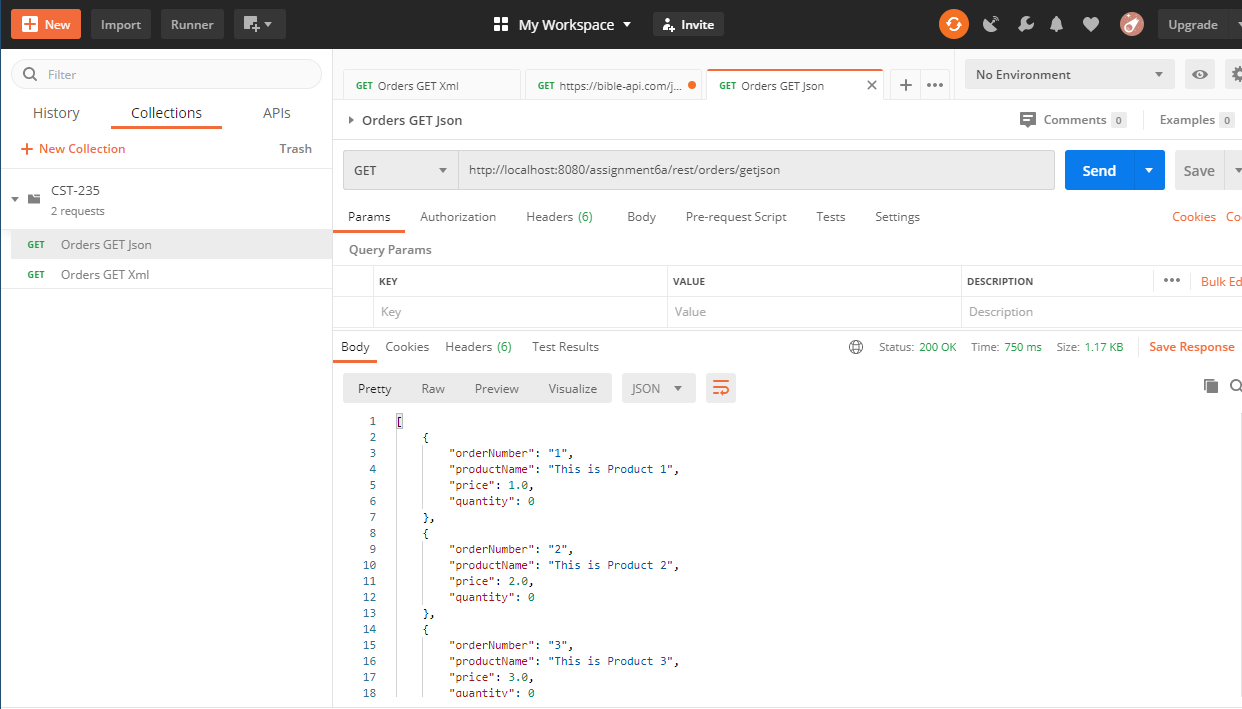


Link to excel sheet:

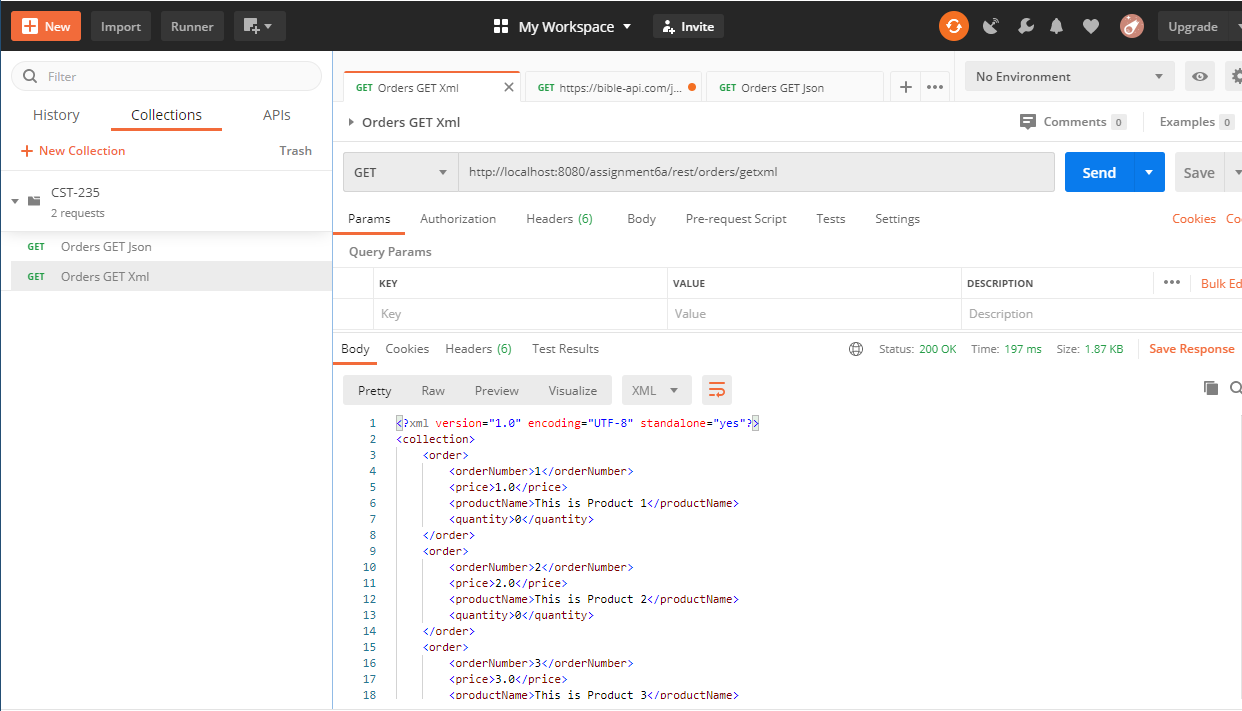
<https://github.com/MTeixeiraGCU/CST-235/blob/master/WeekSix/CST-235RESTAPITestCase.xls>

Part 2:

Screenshot from Postman of getJson:

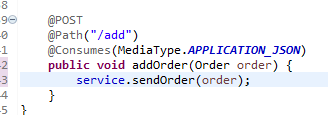


Screenshot from Postman of getXML:



Part 3:

Screenshot from Jboss of POST where new order is placed in the request payload as a Json object:



Screenshot from possible client method using /orders/add REST service:



Part 4:

Research:

There are several best practices to consider when designing REST services. They range from considerations of its usage, the way it can be expanded on, and what formats to use for responses. The most common best practices are semantic. They help keep a resemblance of standard for developers to work from. Listing a few of them shows that they are mainly just to get an understanding of the structure a REST service should take.

* Know how HTTP and REST work together.
* Make sure to use Json payloads a response instead of plain text.
* Avoid action words or verbs in the URI.
* When referring to resources, try to use plurals.
* Do not nest resources with many to many relationships.

These are just a few of the more prominent ones that show up. Since there is a lack of any real standardized format, some of the practices can create conflicts or overlaps. For example, having a trailing slash on a URI is completely up to the designer. The only solution that fits is to handle requests that do not fit the desired format with the proper error codes. There are also some less obvious practices that can help. Versioning for example is recommended early on. Since a REST API relies on the consistency of its format, small changes can ruin applications that are using your service. By keeping versions of the service, you can allow users to decide when to upgrade instead of forcing it each time.

When setting up a new REST service, it can be a daunting task to test every possible request that comes in. Applications such as Postman can help integrate all the possible testing features you would need. Some of the features that Postman offers are:

* Automatically testing your API
* Creating shared workspaces
* Generating documentation
* Client-side code generation in multiple languages
* Designing services using mock a database structure

These are just a few of the highlights of Postman. Even with a team of developers, covering every possible aspect of an API can be difficult. However, with applications such as Postman, development can be streamlined, and regular maintenance is quicker and more reliable.

References

Maciej (June 2019), “9 Best Practices to implement in REST API development” retrieved from: <https://www.merixstudio.com/blog/best-practices-rest-api-development/>

Manca, Florimond (Aug. 2018), “RESTful API Design: 13 Best Practices to Make Your Users Happy”, retrieved from: <https://florimond.dev/blog/articles/2018/08/restful-api-design-13-best-practices-to-make-your-users-happy/>

Postman (2020), “The collaboration Platform for API Development” , retrieved from: <https://www.postman.com/>