**Overview**

Project 2 should be a small full stack application using frameworks (Angular & Spring Boot ). Associates should choose a p2 idea of their own. They should use BDD to design the application and test it. Testing Requirements • All endpoints should be tested with Postman o This includes having them verify the status code returned from the endpoint. • All DAO methods must have tests showing that they work. • Any service method that has logic in it must be tested o These tests should use mocking • Feature files for all features • Step implementations for those features • Tests should be documented • All user stories should be documented Goals • Associates should learn how BDD is used to build applications and features • Associates should learn how to test and work on different parts and levels of the application • Associates should learn how to create basic testing documents

**Front End:**

Angular APP (SPA) Login Form (Component) – Login Validation (with backend data) – Rest end point (http client) Home Page Component (Event Details related to this particular client) - CRUD

**Backend:**

Spring Boot (Restful Webservice)

**DB Layer:**

Postgres running on AWS RDS

**Testing**

Junit Mockito Selenium IDE Test Cucumber BDD Jasmine, Karma & Protractor (SPA)

**Team (Max Size: 3)**

Scrum Master (1) Scrum Team (2)

(Front End, Back End, DB & Testing)

Product Owner – N/A [You will be choosing your project names and other details] – Product Backlog, (order & prioritize the must have, should have, could have features of the requirement) – 3 Sprint (5 days each) Sprint backlog

All the REST end points (web Service) needs to be tested using postman, swagger. Adding Junit test cases, Testing using Mockito Testing using jasmine, karma is also important for front end layer DB Scripts should be located inside the resources folder (All table creation query, sample record insert query for all the tables)

Package structure for the back end com.revature.<app\_name> com.revature.<app\_name>.controller com.revature.<app\_name>.model com.revature.<app\_name>.repo com.revature.<app\_name>.service com.revature.<app\_name>.util com.revature.<app\_name>.exception com.revature.<app\_name>.config

Needed Dependencies

1. Spring Starter Web
2. Spring Boot Starter (default)
3. Postgres Driver
4. Spring data jpa
5. Swagger2 dep (external)
6. Swagger2 ui (external)
7. Devtools
8. Lombok
9. Mockito
10. Thymeleaf
11. H2/MySQL (Optional)

Packages for the front end

Src – app – component1, component2, etc.,

Database – follow db naming convention Java – Follow java coding standards

Add sufficient comments to your code. Properly align the code before you submit. Configure Logging and use Lombok

Java Coding Conventions

* Class name should start with Capital Letters Only
* Properties & method name should start with lower case letter only
* If more than one word for class name, method name & property name, then First letter of each word needs to capitalised (Follow Camel Casing convention)
* All public, static, final properties needs to be in ALL CAPITAL LETTERS only
* Add sufficient comments to the code
* Properly format the code
* Use full & proper variable names

DB Naming Convention

* Table names, column names everything will be in lower case only
* If more than one word, then each word will be separated by underscore.
* Except under score no other special symbols are allowed in table, column name

Front End Requirements

* All the Screens will have similar look & Feel (Apply same CSS style across the application)
* Use some frameworks like bootstrap or material-ui to create the Front end code

**Example :**

Online Shopping

1. User Service – Spring Boot Project (Front End) – Login, Logout, Update Profile, forgot password
2. Product Service – Add product, Edit Product, Delete Product, List all products (Front End, Back End, DB code)
3. Shipping Service
4. Rating Service
5. Payment Service
6. Product Backlog (All the requirements will be captured in this excel file)
7. Sprint Backlog (Task selected by the team for each sprint)
8. Burn down chart (Effort Utilization information)

Finally you can also create project presentation, explaining the code metrics of your project.

Code Metrics – Total lines of code, Total number of Java Classes (Controller, Repo, Model, Exception, Service, Util, DB Tables, Angular Components, Front End Services,Routes, Total number of Junit Test cases)

Capture screenshot of each page of the application along with brief details about the user inputs & expected output details.