Development guide for PizzaHouse Application

1. Environment requirement
2. IDE Setup
3. Database setup and initialize
4. Module dependencies
5. Hibernate ORM and Database structure
6. EJB
7. Controller and Router
8. Junit test
9. Initializer
10. Security

1. Environment requirement

|  |  |
| --- | --- |
| Java minimum version | 1.8 or above |
| IDE | Eclipse or equivalent |
| Application Server | Tomcat 8.5 or equivalent |
| Maven | Make sure maven is installed on the development computer |
| Docker | Make sure docker is installed on the development computer |
| Git | Git command or Git application |
| MySQL server | MySQL server 5.7.30 or above |
| Source | https://github.com/brianlaihkhk/coding-test-pizza-house |

1. IDE Setup

Below illustration are using Eclipse as the development example.

a. import as maven project

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

b. Setup Junit test 4 under Run as -> Run configurations -> Junit

Graphical user interface, text, application, chat or text message

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Also need to setup Jasypt password for decoding encrypted data :

Graphical user interface, text, application, email

Description automatically generated

Also if your JRE version is before 8u157, please install JCE policy into <JavaHome>/jre/lib/security folders

Download : <https://www.oracle.com/java/technologies/javase-jce8-downloads.html>

Graphical user interface, table

Description automatically generated

1. Database setup and initialize

Inside the Setup folder, there are 3 sql script

|  |  |
| --- | --- |
| 00\_DATABASE\_INIT.sql | Initialize the database, create user and set according permissions |
| 01\_CREATE\_TABLE.sql | Create table / schema script (Here we don’t use hibernate.hbm2ddl.auto to generate) |
| 02\_IMPORT\_DATA.sql | Import testing data |

4. Module dependencies

There are 3 modules in the project

|  |  |  |
| --- | --- | --- |
| Module | Description | Remarks |
| Common | Common modules that are shared with PizzaService and OrderConfirmationService | Please apply “mvn clean package install” for initial setup and module update.  Pom.xml of PizzaService and OrderConfirmationService relys on Common thus without doing this, those service will have error |
| PizzaService | Depends on Common module. PizzaService is the Webservice that receives customer order.   * Provide checking and validation of the input data * Provide session token for the access control * Provide expireation of the token for the access control * Calculating subtotal for each item and grand total for the order * Covert user order request to confirmation data and send to OrderConfirmation Service using secure channel (JWT encryption) | Depends on Common |
| OrderConfirmationService | Depends on Common module. OrderConfirmationService is to persist the ordered data into database, and send secure information back to the PizzaService.   * Provide purchased data to the database * Data written as each item and grand total for future analysis | Depends on Common |

5. Hibernate ORM and Database structure

All entity (POJO that persist to database are stored in com.pizzahouse.xxxx.entity package. Depends on the nature of the entity.

PizzaService

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Referenced Table | Association | Remarks |
| Com.pizzahouse. order.entity.User | USER | PK : USER\_ID |  |
| Com.pizzahouse. order.entity.Session | SESSION | FK: USER\_ID to USER.USER\_ID |  |
| Com.pizzahouse. order.entity.Pizza | PIZZA | FK: PIZZA\_SIZE\_ID, PIZZA\_TYPE\_ID, PIZZA\_TOPPING\_ID | Mapping table for Pizza based on the property |
| Com.pizzahouse. order.entity.PizzaSize | PIZZA\_SIZE | PK: PIZZA\_SIZE\_ID |  |
| Com.pizzahouse. order.entity.PizzaType | PIZZA\_TYPE | PK : PIZZA\_TYPE\_ID |  |
| Com.pizzahouse. order.entity.PizzaTopping | PIZZA\_TOPPING | PK : PIZZA\_TOPPING\_ID |  |

OrderConfirmationService

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Referenced Table | Association | Remarks |
| Com.pizzahouse.order.entity.Purchase | PURCHASE | PK: PURCHASE\_ID  Remarks : FK : USER\_ID is omitted due to Junit test case limitation. Please set the FK if it is in production mode. | * One Purchase contains multiple Purchase detail. |
| Com.pizzahouse.order.entity.PurchaseDetail | PURCHASE\_DETAIL | PK: PURCHASE\_DETAIL\_ID  FK : PURCHASE\_ID to PURCHASE.PURCHASE\_ID | * PURCHASE\_DETAIL\_ID is auto increment key * One Purchase contains multiple Purchase detail. Association using PURCHASE\_ID |

6. EJB

EJB confirmation is in WebConfiguration.java under com.pizzahouse.service.initialization and com.pizzahouse.order.initialization

|  |  |
| --- | --- |
| Org.slf4j.Logger | Slf4j logger EJB, all logger will be populated under @Autowired |
| Org.hibernate.session | DB session EJB, database connection will be configurated under @Autowired by Data |

7. Controller and Router

For the WS doc, please refer to WS.yml

Under spring mvc setup, all controller logic is placed inside com.pizzahouse.service.controller and com.pizzahouse.order.controller

PizzaService

|  |  |
| --- | --- |
| Com.pizzahouse.service.controller.Router | Router object, as a focal point for WS request receiving |
| Com.pizzahouse.service.controller.OrderService | Order related logic |
| Com.pizzahouse.service.controller.UserService | User related logic |

OrderConfirmationService

|  |  |
| --- | --- |
| Com.pizzahouse.order.controller.Router | Router object, as a focal point for WS request receiving |
| Com.pizzahouse.order.controller.ConfirmationService | Order confirmation related logic |

8. Junit test

PizzaService

|  |  |  |
| --- | --- | --- |
| Junit Test | Description | Dependency |
| IntegrationTest | E2e integration test | Database, OrderConfirmationService |
| PizzaMapTest | PizzaMap initialization test | Database |
| OrderServiceTest | Test place order and calculation logic | Database |
| UserServiceTest | User creation, get seesion token, login |  |

OrderConfirmationService

|  |  |  |
| --- | --- | --- |
| Junit Test | Description | Dependency |
| ConfirmationServiceTest | E2e integration test | Testing for confirmation data is persisted to DB |

9. Initializer

Preload and propagate data during startup

PizzaService

|  |  |  |
| --- | --- | --- |
| Initializer | Description | Remarks |
| Com.pizzahouse.service  .initialization.DataLoader | Populate Pizza mapping in initial startup |  |
| Com.pizzahouse.service  .initialization.PropertiesLoader | Populate Properties during startup | Requires Common |

OrderConfirmationService

|  |  |  |
| --- | --- | --- |
| Initializer | Description | Remarks |
| Com.pizzahouse.service  .initialization.PropertiesLoader | Populate Properties during startup | Requires Common |

10. Security

PizzaService

|  |  |
| --- | --- |
| SessionToken | Assign SessionToken once user login or registration, default expiry time is 30 days |
| JWT connection to OrderConfirmationService | Message are encrypted and send by encrypted strings between services |

OrderConfirmationService

|  |  |
| --- | --- |
| JWT connection to PizzaHouse | Message are encrypted and send by encrypted strings between services |