ABC’s Inventory Management System

Programmer Manual

Table of Contents

[INTRODUCTION 3](#_Toc527370497)

[Purpose 3](#_Toc527370498)

[What is the ABC Inventory Management App? 3](#_Toc527370499)

[TOOLS AND PLATFORM 3](#_Toc527370500)

[Platform 3](#_Toc527370501)

[Tools 3](#_Toc527370502)

[APPLICATION ARCHITECTURE 4](#_Toc527370503)

[Layer or architectural framework 4](#_Toc527370504)

[Architectural View 4](#_Toc527370505)

[PROGRAMMER’S GUIDE 7](#_Toc527370506)

[Database 7](#_Toc527370507)

[Server Application 8](#_Toc527370508)

[Client Application 10](#_Toc527370509)

# **INTRODUCTION**

## Purpose

The purpose of this document is to support rapid and accurate understanding of software in a way that would allow a programmer to become productive in maintaining the software immediately after reading the programmer documentation.

## What is the ABC Inventory Management App?

The ABC Inventory Management App is a desktop application that supports most convenient way to get detailed and precise information about the stock movement and stock theft. This would also allow employees to look up current stock contents of the warehouse and each store and then request stock as required. Employees will also be able to send stock digitally i.e. automatically make an update to the database when stock is sent. Moreover, distribution of stock would also be much easier for the warehouse staff as they will know the quantity of each stock for each store.

# **METHODOLOGY**

In this project, software development process and project management will be conducted using the Unified Process (UP) Framework. UP is a use-case driven, architecture centric, risk prioritized, iterative, incremental’ process. There are four phases of UP which are The Inception Phase, The Elaboration Phase, The Construction Phase and The Transition Phase.

This project will be conducted in two steps between two semesters. In the first step (semester 201830) the first two phases of the Unified Process will be conducted which focus to conduct the fundamental documents for creating the concept of what the project will achieve, demonstrating the team has the competency to achieve it and a small prototype for creating and validating an architecture and design to support implementation.

While the final two phases will be conducted in the second step of the project i.e. semester 201860. In this final two phases, the team is responsible for developing and testing the majority of project software by implementing the completely product. Then the team must demonstrate that the product is fit-for purpose and handing the product over for use and maintenance in a production environment.

Following the UP framework, this project will be iterative and incremental after each iteration. Therefore, a detailed iteration plan will be made for each iteration and each team member will be expected to conduct their tasks in accordance with the iteration plan and integrate their completed tasks. This way team members can keep track of what task was done when and when it was approved. Changes to the project and the iteration plan can also be made if required. Documentation changes can also be made as required. Finally, the final iteration will be conducted which will produce the final product of the project.

Similarly, after each iteration is completed, a team member will conduct an iteration assessment and write down an iteration burndown report. This will help clarify which task was completed and which task needs completion from the previous iteration. A check list will also be created for each task in the iteration to track the progress of the iteration which will help in the preparation of the iteration assessment report. Gantt charts will be used to plan and schedule the project. This will help team members assess time and resource needs and dependencies of the project. It will also provide a timeline of the entire project.

When all the iterations are completed, a final report will be prepared assessing the entire project to check if any changes need to be made.

# **TOOLS AND PLATFORM**

## Platform

ABC Inventory Management Desktop Application is developed by combining different platforms:

* The Java EE platform: is built on top of the Java SE platform. The Java EE platform provides an API and runtime environment for developing and running large-scale, multi-tiered, scalable, reliable, and secure network applications.
* JavaFX: is a platform for creating rich internet applications using a lightweight user-interface API. JavaFX applications use hardware-accelerated graphics and media engines to take advantage of higher-performance clients and a modern look-and-feel as well as high-level APIs for connecting to networked data sources. JavaFX applications is the client of Java EE platform services.

## Tools

These are the tools and frameworks which were used for developing ABC Inventory Management System:

|  |  |
| --- | --- |
| **Type** | **Tools/ Frameworks** |
| IDE | IntelliJ IDEA 2018.1 |
| Framework | Jersey RESTful service 2.26-b06 |
| Database Management System | MySQL 5.1.38 |
| Project management tool | Apache Maven |
| SDK | 1.8.0\_171 |
| Application Server | Tomcat 8.0.521 |

# **APPLICATION ARCHITECTURE**

## Layer or architectural framework

ABC Inventory Management System uses 3 layers architecture. The following diagram represents the layers of the system:

Data Layer

User Interface Layer

Business Logic Layer

As described in the diagram, 3 layers are:

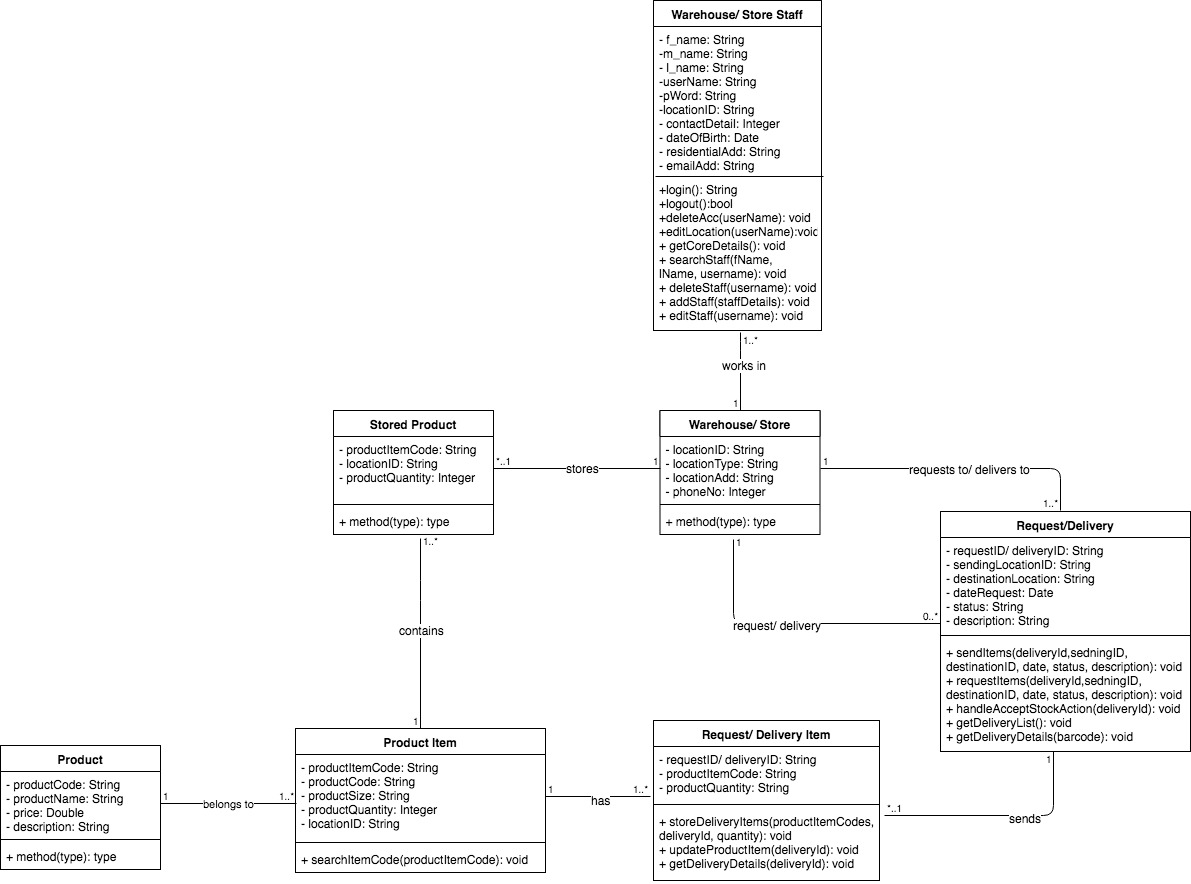
* User Interface Layer: plays a role as the client application (provided by ABC developer team) which is responsible for managing the interaction between user and application. This application gets the users’ requests, send it to Business Logic Layer (Server) and Data Layer then get the results from those layers to return back to users.
* Business Logic Layer: play a role as the server application which manages the logic of application. It is a bridge between the client application and database. This layer will be deployed in the cloud by using AWS. Elastic Beanstalk has been used to deploy the server side application. To make any further maintenance, developer should develop by using the server application.
* Data Layer: plays a role as the database. This layer is deployed in AWS cloud server so user can store and retrieve data without any limit about the distance. Amazon RDS was used to deploy the database.

## Architectural View

The following are the architectural views that will describe the software architecture:

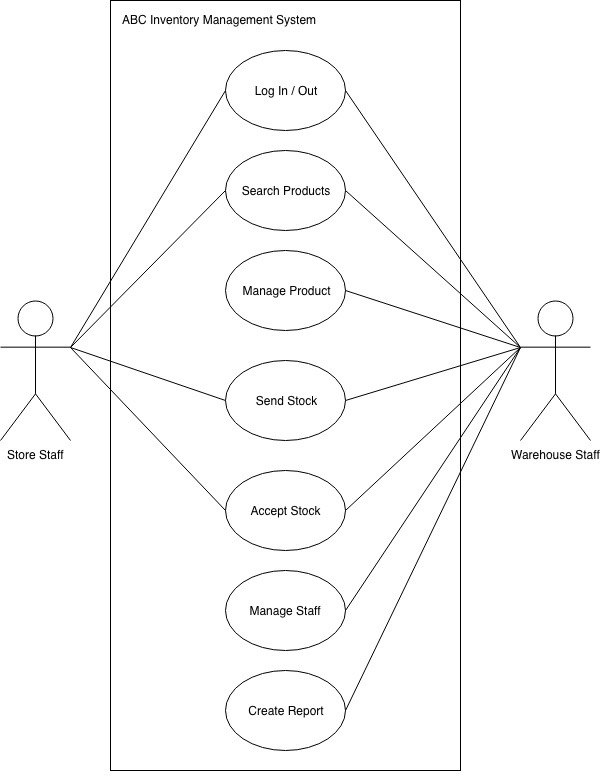
**Logical**

The following is the logical view that describes the software architecture:



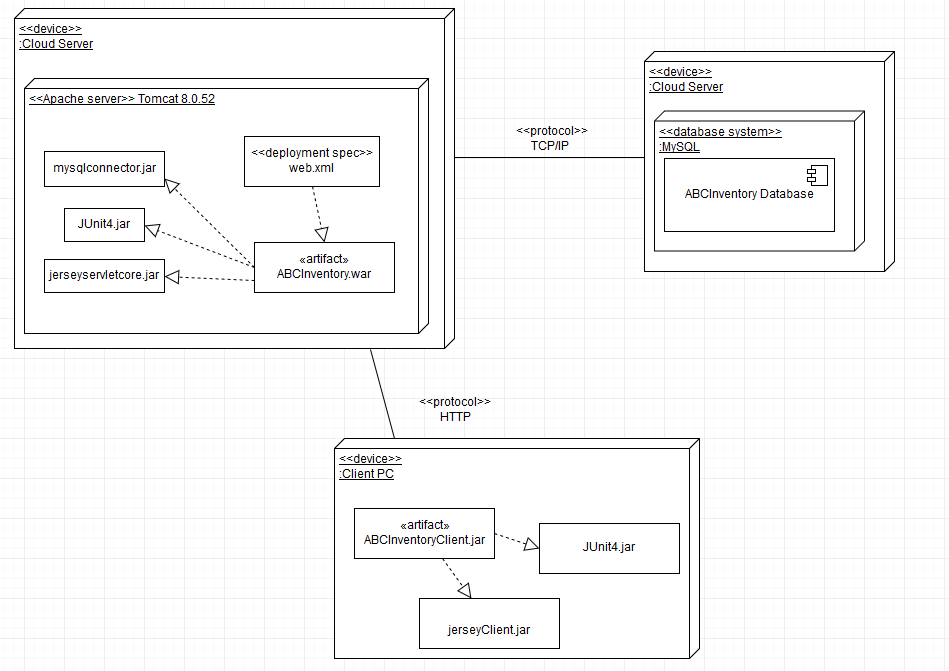
**Use Case**

The following is the use case view of the software architecture:



**Physical**

The following is the physical view that describes the architecture:

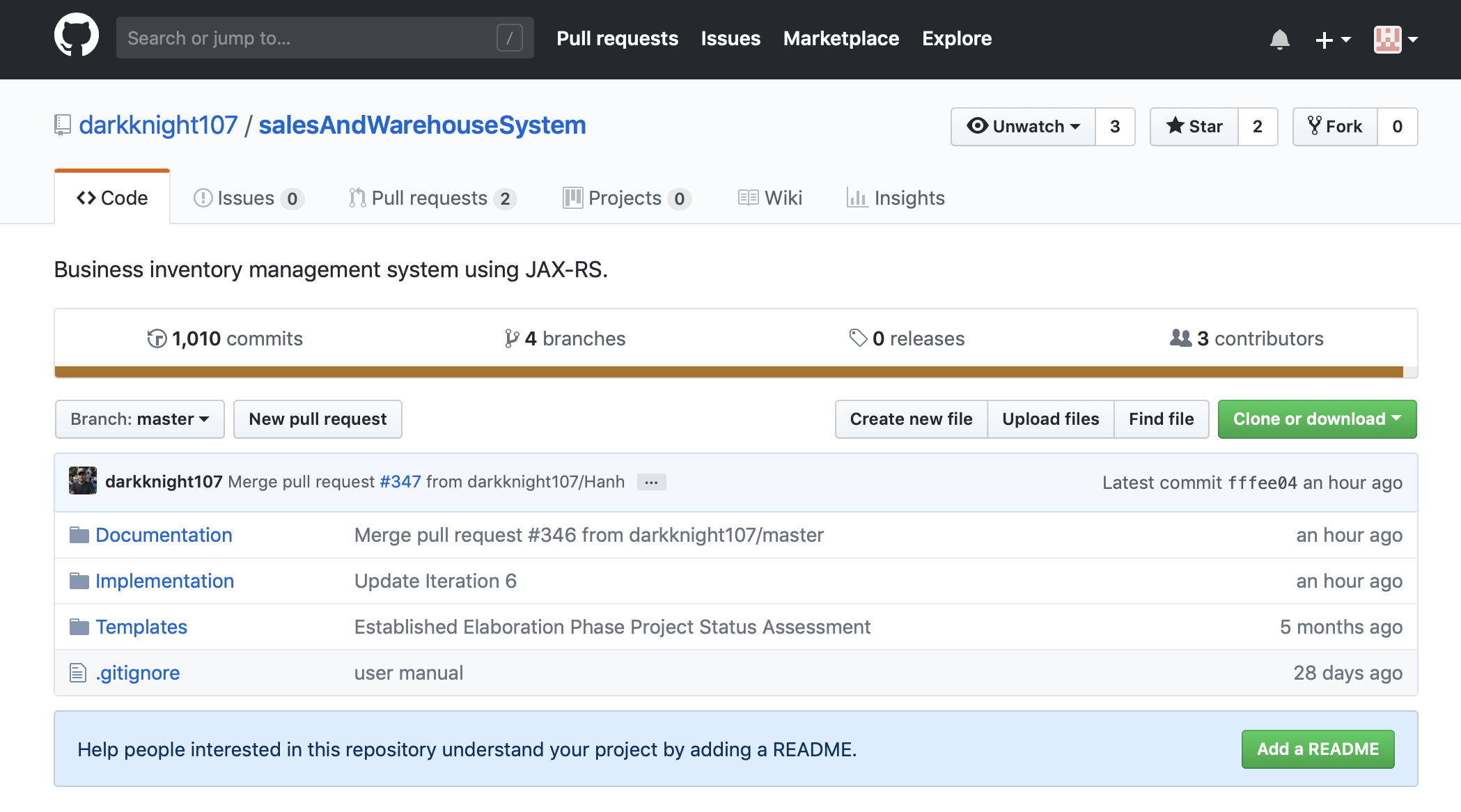


# **VERSION CONTROL REPOSITORY**

Git Hub is using as a code hosting platform for version control and collaboration. Therefore, the team members can work together on the projects from anywhere. Anyone can access the version control repository for contribution and maintenance for ABC Inventory Management System by accessing to this link.

|  |  |
| --- | --- |
|  | Link |
| Version control repository | <https://github.com/darkknight107/salesAndWarehouseSystem> |
| Version control document | <https://onedrive.live.com/edit.aspx?cid=a155e9eb8e919b6c&page=view&resid=A155E9EB8E919B6C!11992&parId=A155E9EB8E919B6C!11379&authkey=!ANIn2MdaFfbtttg&app=Word> |

There are 3 folders in the repository:

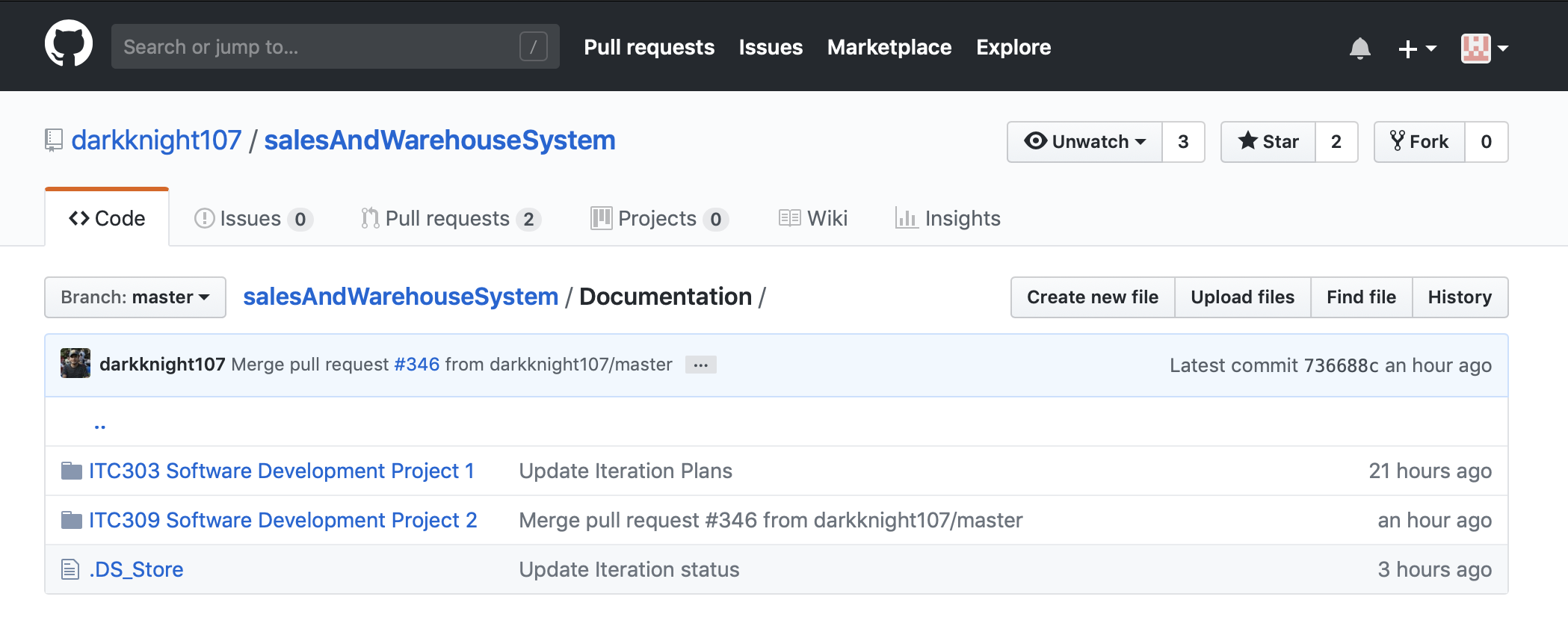


* Documentation: contains all the necessary documents for developing and maintaining the system.
* Implementation: contains the codes, executable system.
* Templates: contains the templates of documents.

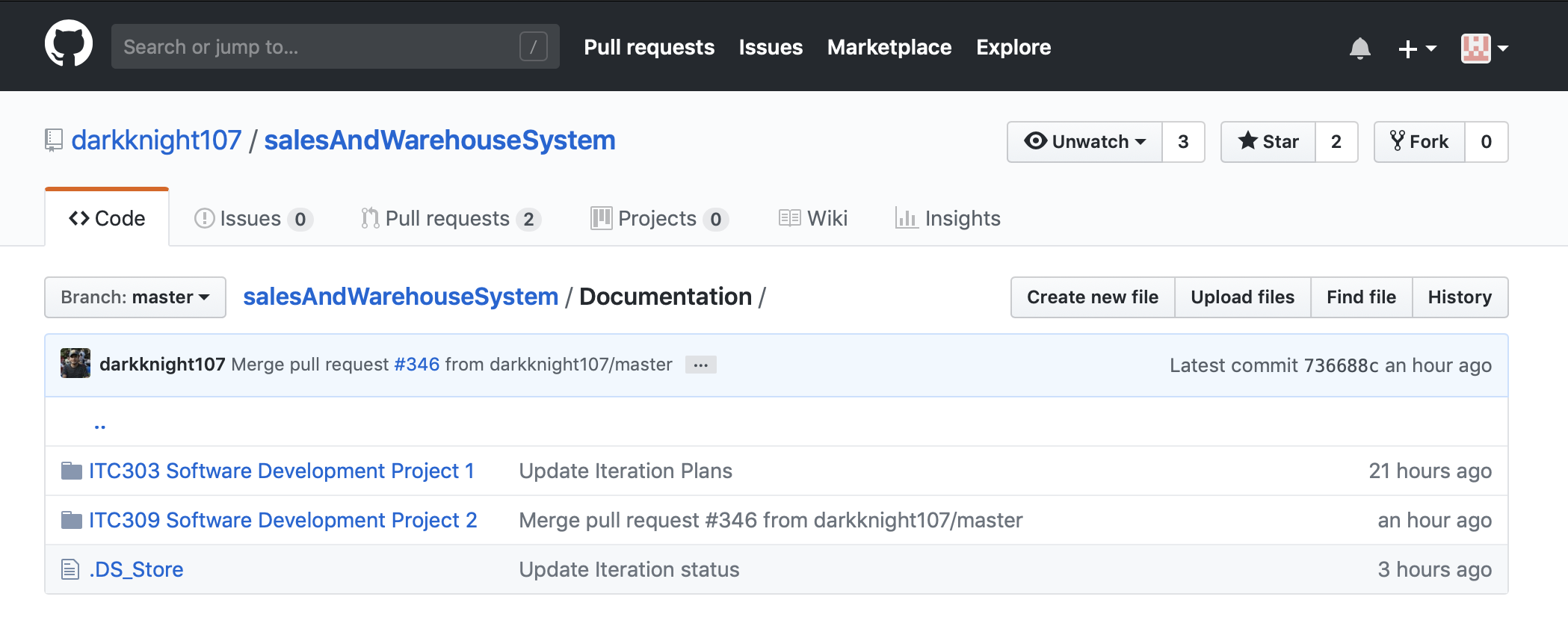
Documentation

The documentation folders contain two folders:

* ITC303 Software Development Project 1: contains the documents for the first 2 phases (Inception Phase, Elaboration Phase).

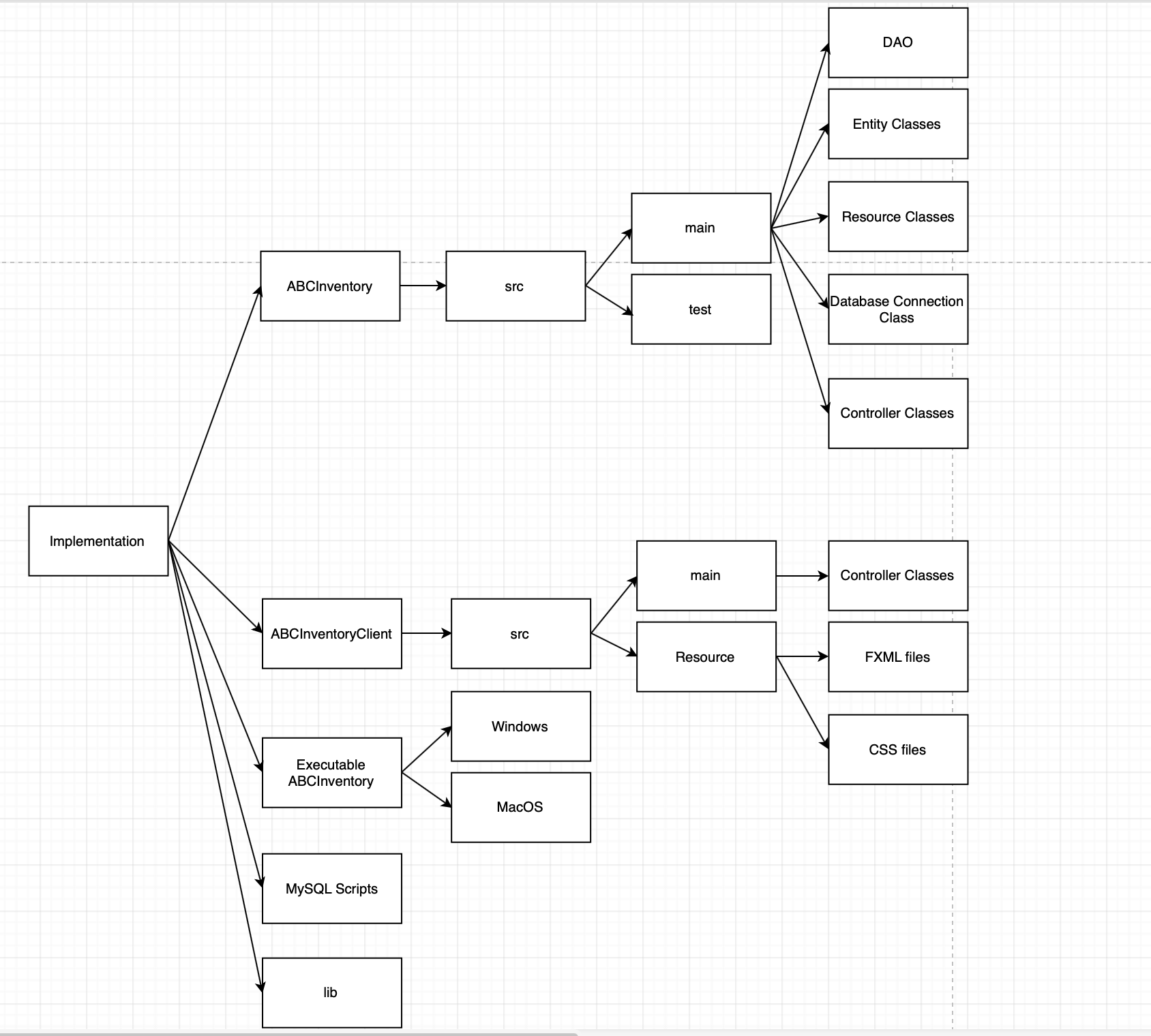


* ITC309 Software Development Project 2: contains the documents for the last 2 phases (Construction Phase, Transition Phase).



Implementation

The implementation is responsible for containing all the files which relate to implementing project such as server application, client application, database script. By accessing this folder, developer can get a deep insight about the project, and make the maintenance from it. The structural location of this folder is demonstrated by the below diagram.



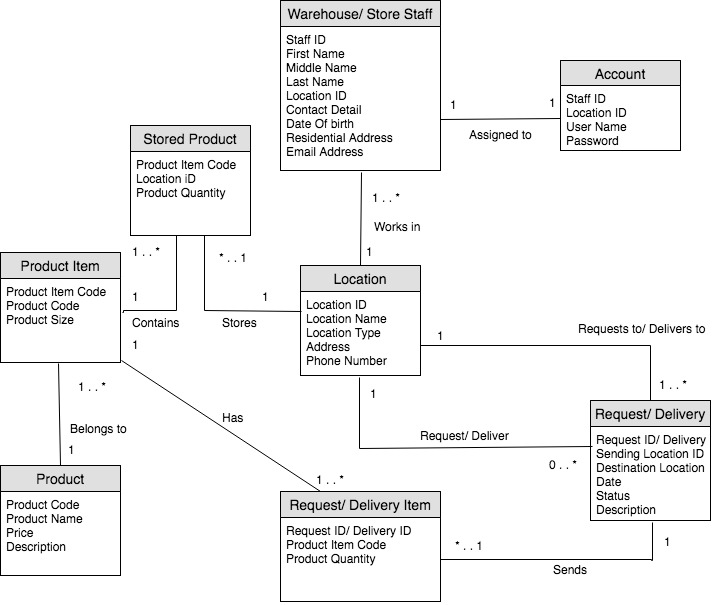
Note: The detail explanation for the server application and client application will be described in the last part: PROGRAMMER’S GUIDES. Please read this part to get more information

# **PROGRAMMER’S GUIDE**

As mentioned in the Application Architecture Part, ABC Inventory Management System is using 3 layers architecture which includes: client application, server application, database.

## Database

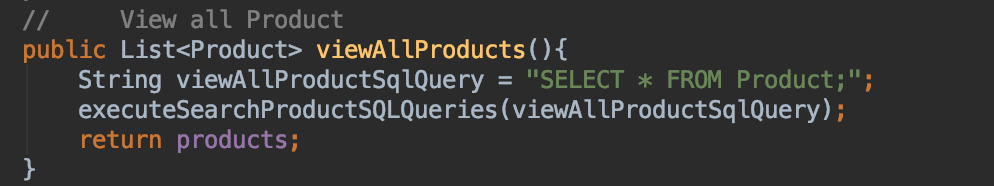
The database is designed base on logics of class diagram (relationships, entities, etc.) and share data by using JDBC standard.



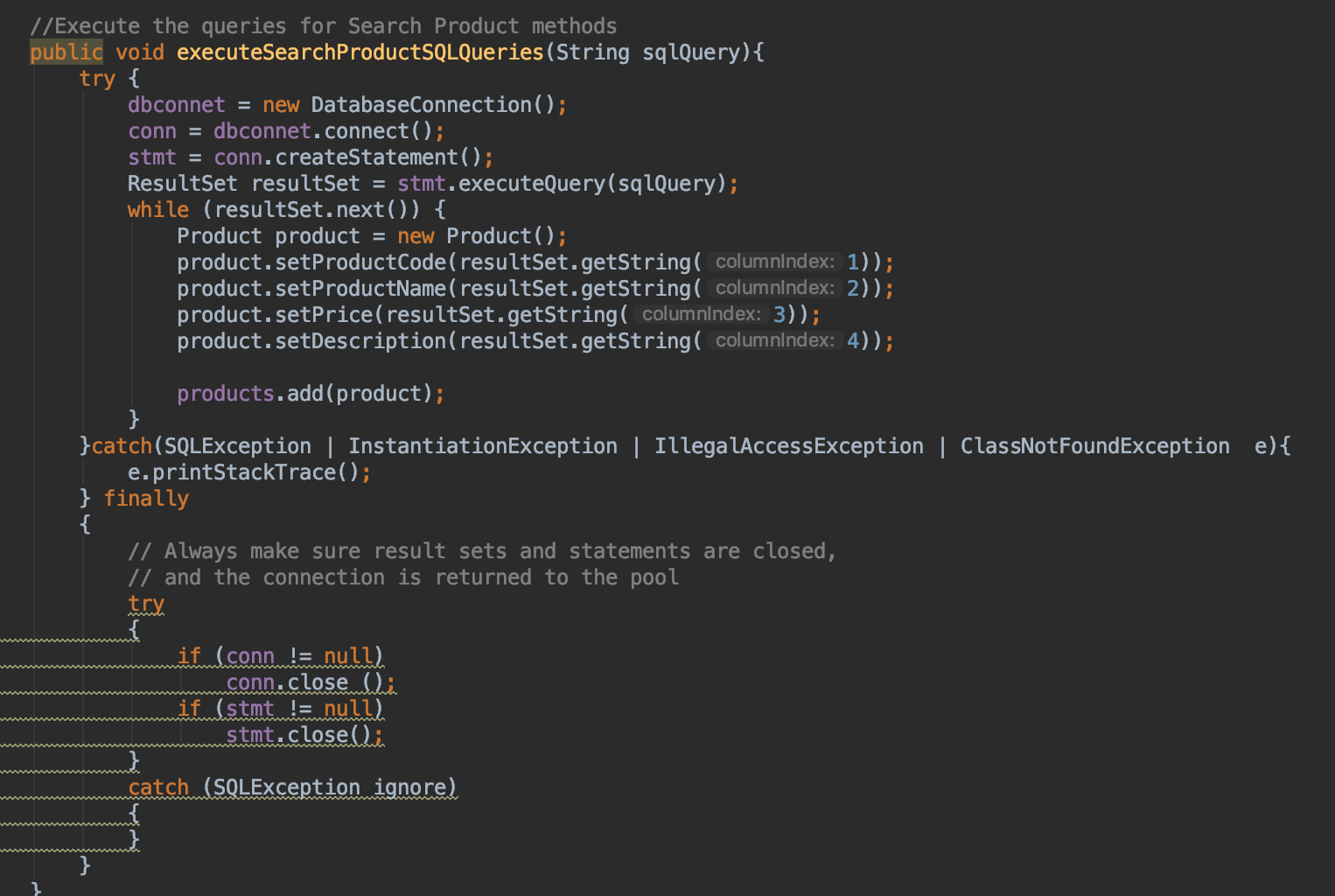
## Server Application

The Server Application named as ABCInventory. This application is deployed by using AWS Cloud Server. There are 4 main parts in this application:

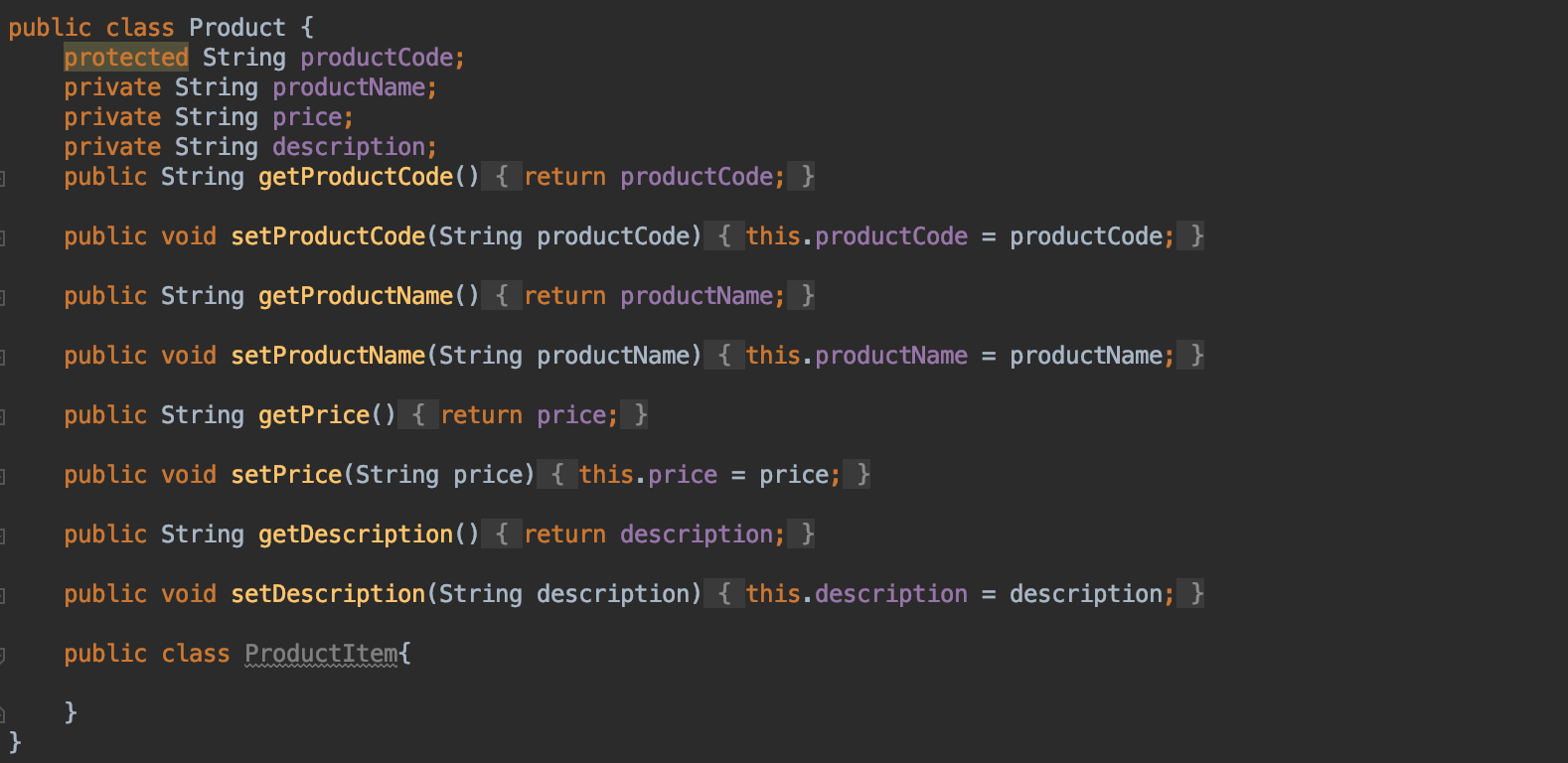
* DAO files: is the objects that provide an abstract interface to some type of database or other persistence mechanism. By mapping application calls to the persistence layer, the DAO provides some specific data operations without exposing details of the database. There are 4 DAO in this application: Account, Product, Staff and Transfer. Each DAO contains multiple methods, there are 2 main kind of methods:
  + execute SQL query methods: execute the receiving query to get the data from database.



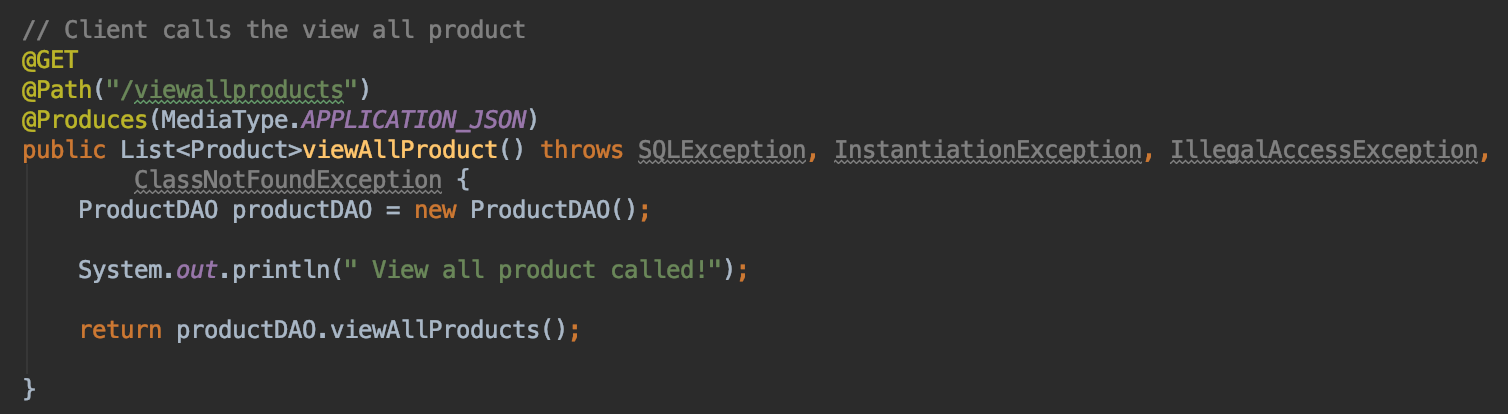
* + Functional methods: set the query for a specific function.



* Entity files: ordinary Java class that is marked (annotated) as having the ability to represent objects in the database. It includes: Product, Product Item, Search Account, Search Product, Staff, Stored Product, Transfer, Transfer Item. For example:



* Resource classes: are POJOs (Plain Old Java Objects) that are either annotated with @Path or have at least one method annotated with @Path or a request method designator such as @GET, @PUT, @POST, or @DELETE. *Resource methods* are methods of a resource class annotated with a request method designator. For examples:



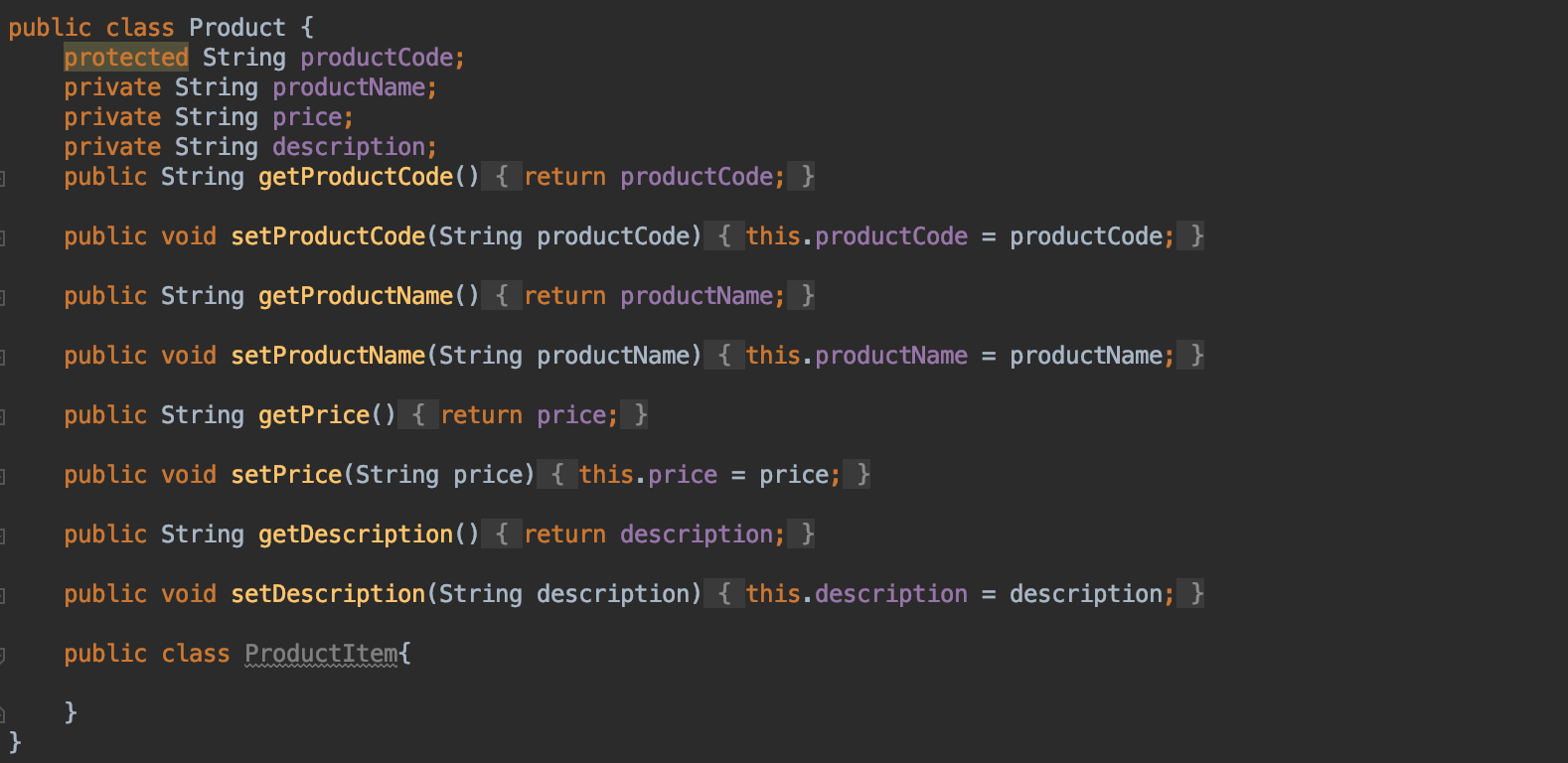
* Pom.xml file: contains information about the project and configuration details used by Maven to build the project.



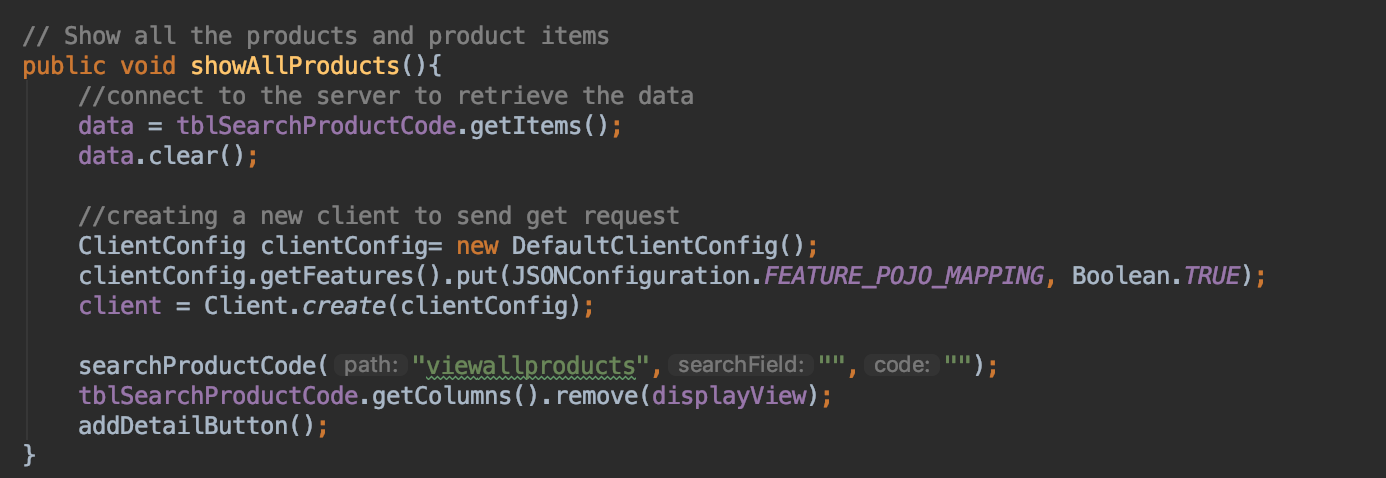
## Client Application

The Client Application named as ABCInventoryClient. This application is developed for user to use the system. There are 3 main parts of this application:

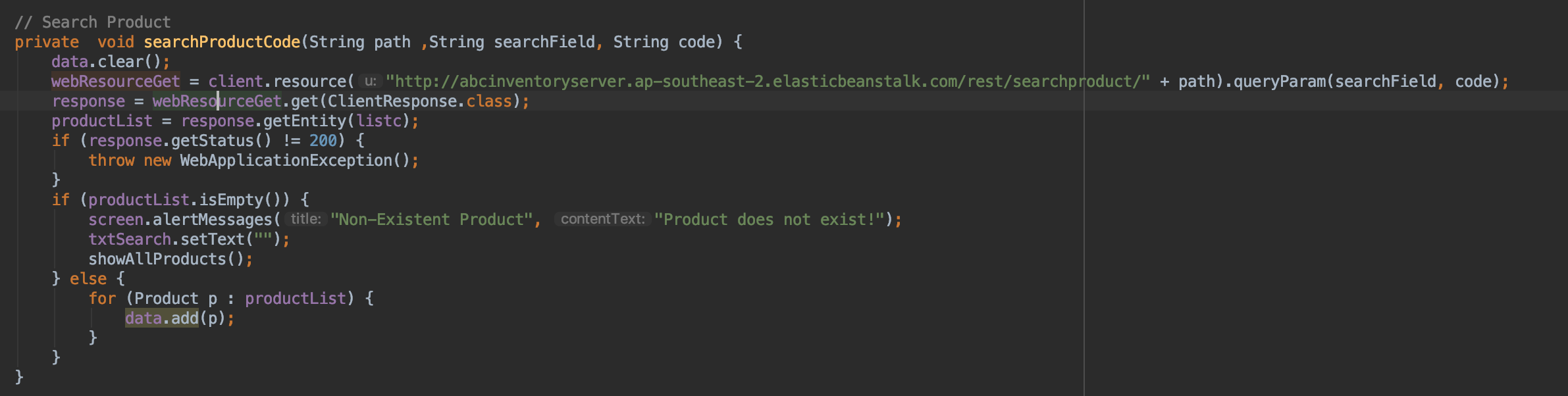
* Entity files: ordinary Java class that is marked (annotated) as having the ability to represent objects in the database. It includes: Product, Product Item, Search Account, Search Product, Staff, Stored Product, Transfer, Transfer Item. For example:



* Controller classes: get the users’ request, send it to server application and get the results from server to display into client interface. The data will be sent and received by using JSON data format. The class have 2 main kinds of method:
  + Functional methods: execute the functional requests.



* + Controlling methods: play a role as the bridge between client and server. It gets the functional request from client and get the result from server by using a url.



* + FXML files: allow application developers to separate the logic for building the UI from the business logic

