20th March 2018

ASSIGNMENT 2: JAVA EE WITH JSP

1. SERVICE ORIENTED ARCHITECTURE

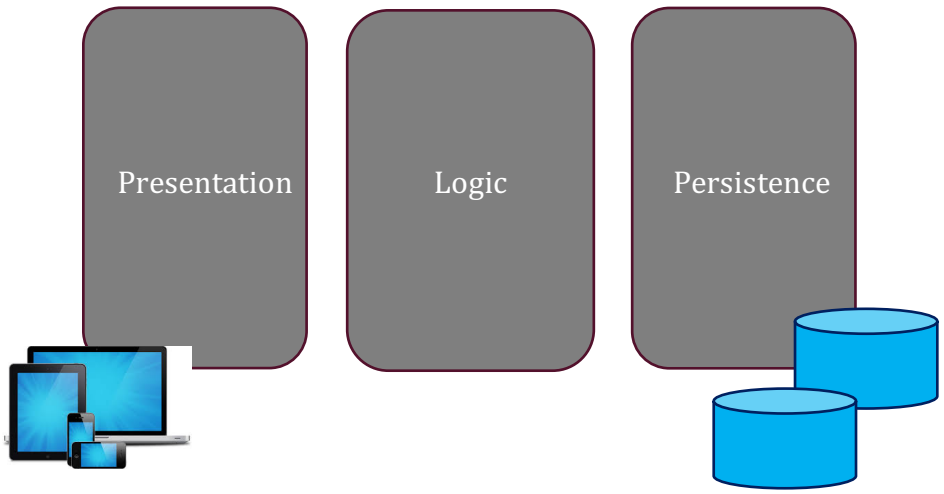

Figure 1 Service Oriented Architecture

A 3-layered architecture framework will be implemented in NetBeans. The Presentation Layer consists of web and mobile clients/browsers, which are anticipated to scale to large number of users. The Logic Layer consists of the methods used to retrieve and/or persist data in the Persistence Layer. The Logic Layer consists of methods created in Netbeans 8.2 with JDK 8, installed with GlasshFish Server, which will function as Application Server. The Application Server will connect to MySQL Database 5.7.21, a component in the Persistence Layer. MySQL is preferred over Derby because it is more stable, easier to use on Mac OS X , has external administrative tool (MySQL Workbench) and supports wider variety of languages aside from Java. It has also better rankings than Derby.

This architecture is analogous to the Model-View-Controller architectural pattern. In this approach, internal representation of information is separated to presentation/interaction of application with the user. It offers similar benefits as SOA architecture described above:

(i) Modularity of design considerations
(ii) Easy to maintain and upgrade
(iii) Easy collaboration among developers due to its modularity

The Model, is responsible for managing the data of the application and receives user input from the controller. It consists of Session Beans, i.e. Enterprise Java Beans with Java Persistence API (JPA) Entity Classes. The JPA Entity classes contain named queries, fields representing columns in tables and relationships representing foreign keys. These classes are derived from the tables in the Database whereas the Enterprise Java Beans contain Session Façade files which contain the Create, Read, Update, Delete (CRUD) logic dealing with basic data access methods. Both Entity Classes and Session Beans are easily created, to generate model quickly for the application. The View, is responsible for the presentation of the application to the user. It consists of Java Server Pages (JSP), pages that are in .jsp format. It is similar to HTML except that it could contain Java scripts and JSP-Standard Tags (JSTL) in addition to static markup content. As with HTML, it uses styling from Cascading Style Sheets (CSS). The JSPs are placed in a special folder called */Web/WEB-INF/*view instead of */Web*, so that it could not be unintentionally/intentionally invoked via URL on browser. Users interact with application via JSP, which forwards the inputs through GET/POST methods to controller. The Controller is responsible to handle user inputs by starting actions needed to generate model for request and then forwarding the request via Request Dispatcher to the appropriate view/page. The Controller folder in the project consists of all Servlets required for different functionalities. The MVC approach can best be described with the following diagram.

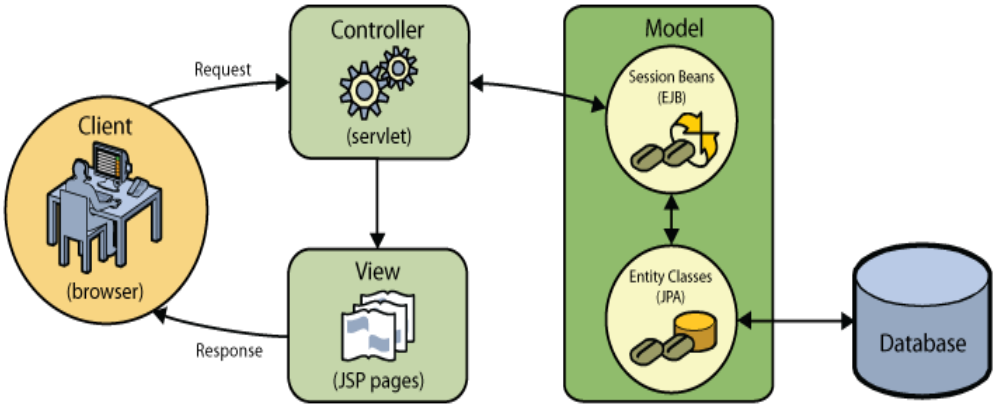

Figure 2: Model-View-Controller Design Pattern (from Netbeans Documentation)

2. FUNCTIONALITIES IMPLEMENTATION

A. Use Cases (As per project requirements)

- User visits **LogIn Page**
- User register in website (if not an existing user) or login
- On **Welcome Page**: User is able to view pinterest event theme. User is able to navigate to Settings Page / Statistics Page via links at the top right corner
- On **Settings Page:** User is able to edit his/her own details

- On **Statistics Page**: User is able to view number of times he/she has logged in in the past two weeks , i.e. 14 days