**Volunteer Work Schedule System**

**Architectural Diagram**

**Prepared by: Abe Gustafson**

**Date: Oct 12, 2016**

Figure 1 shows the architectural diagram for the Volunteer Work Schedule System

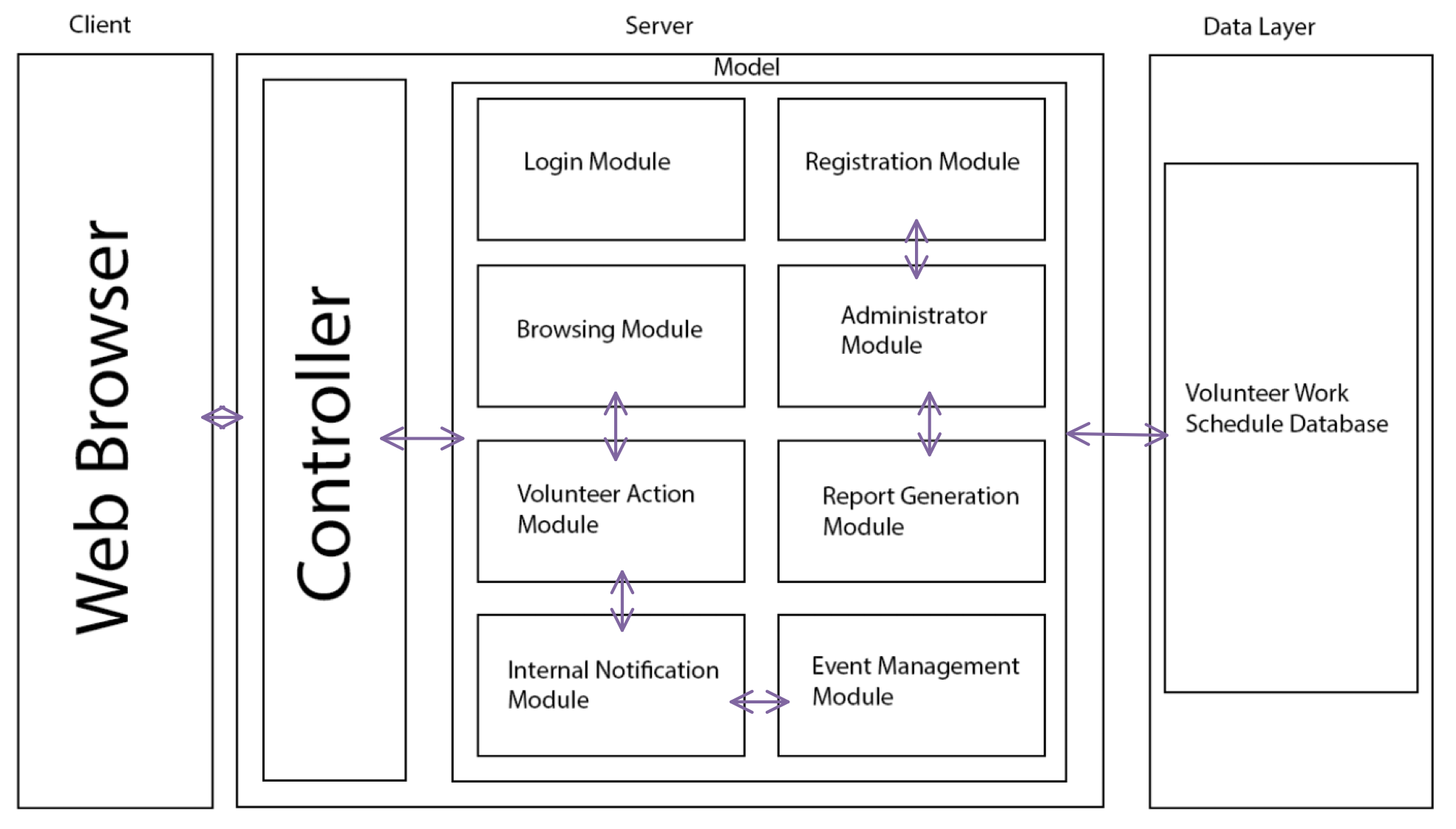


Figure 1: Architectural Diagram for Volunteer Work Schedule System

Being a web-based application, the architecture uses a client-server style. A web browser which connects to the server will act as the client. Details of GUI diagram used by the client will be shown in a separate document. The architecture uses a MVC (Model-View-Controller) pattern; consequently, the diagram shows the three components – Model which, in this case, consists of a collection of modules, View represented by the GUI and Controller. The controller is responsible for interactions with the client – getting requests from the client and sending response back to the client. While getting requests from the client, the controller processes some information from the client and then passes the request to the model. There are eight modules in the model; they are described below. Each module description also indicates which functional requirements in the requirements document [1] are implemented by the corresponding functionality or functionalities and so the readers are recommended to refer to [1] for indexes of these functionalities.

Login Module: This module controls login and logout functionalities (LN.1 and LN.2).

Browsing Module: This module contains implementation details corresponding to browse available volunteer events (User.2). Figure 1 shows an interconnection between this module and the Volunteer Action module. This connection indicates that a user can perform a Volunteer Action directly from the Browsing Module. This connection is explained further in User.3 (Sign up for event) and User.4 (Withdraw from event).

Registration Module: This module includes the implementation details of the functionality corresponding to user registration and create new admin account (User.7, and Admin.4). Figure 1 shows the connection between the User Registration Module and the Volunteer Work Schedule Database. This connection is explained in User.7 (User Registration). The interconnection with the Administrator module indicates that Administrators also use this module directly when creating other Administrator accounts (Admin.4). The connection also means that only administrators can create other administrator accounts.

Event Management Module: The functionalities to add an event (User.1), edit an event (User.5), and delete an event (User.6) are all implemented by this module. The interconnection between this module and the Internal Notification module indicates that some of these functions can result in notifications. This interconnection is explained in User.5 (edit an event), and User.6 (delete an event).

Volunteer Action Module: This module contains implementation details corresponding to the sign up for volunteer event and withdraw from volunteer event functionalities (User.3 and User.4). The interconnection between this module and the Internal Notification module indicates that some of these functions can result in notifications. This interconnection is explained in User.3 (sign up for volunteer event), and User.4 (withdraw from volunteer event).

Internal Notification Module: This module includes the implementation details of send/receive notifications (User.8 and User.9). Along with being used for automated messages, users can also send their own notifications, described in User.8 (send notification).

Report Generation Module: This module implements all report generation functionalities (Admin.1). The interconnection with the Administrator module indicates that only an administrator can generate reports.

Administrator Module: This module holds the unblock account and create new preset volunteer roles functionalities (Admin.2 and Admin.3), as well as having control over the administrator privileges. The administrator module can be easily expanded in the future without compromising the other parts of the system when more administrative functionalities are added.

The interconnection between the Server and the Volunteer Work Schedule System Database indicates that all components of the server including the Controller and all the six modules can access the database. Similarly, the interconnection between Controller and the Model indicates that the Controller can access all modules.

**References**

1. Abe Gustafson, Software Requirements Document for Volunteer Work Schedule System, prepared for CS 741: Software Engineering Principles, Oct 2016.
2. Kasi Periyasamy, Architectural Diagram for Transportation Service System, prepared for CS 741: Software Engineering Principles, Oct 2016.