## System Architectural Design

## 4.2.1 Choice of System Architecture



Figure 1: MVC implemented in a three-tier architecture

The above diagram is explained as the following:

* **Model**: Functions that interact with the database or perform complex operations. In this architecture, model includes business logic and data access in one layer. Contains data and business logic (Action, BLO (business logic object), DAO (data access object) and DTO (data transfer object) classes).
* **View**: The view displays the data and also takes input form user. It renders the model data into a form, such as graphics, buttons, and tables, and so on to display to the user.
* **Controller:** The controller handles all requests coming from the view or user interface. The data flows to whole application are controlled by controller.
* **Business Logic:** This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.
* **Data Access:** Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.

## Discussion of Alternative Designs

### Three-layer Architecture

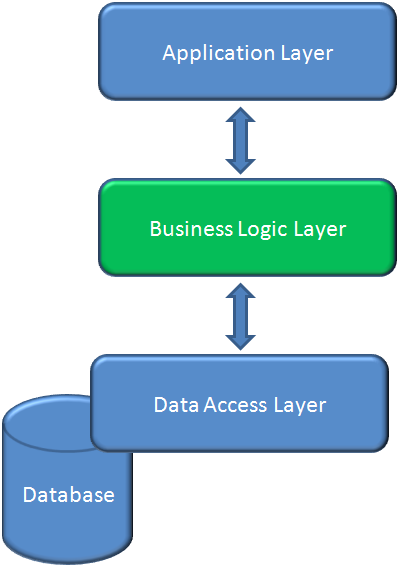


Figure 2: Three-layer architecture

* **Data layer:** is basically the server which stores all the application’s data.
* **Business layer:** is mainly working as the bridge between Data layer and Presentation layer. All the Data passes through the Business layer before passing to the presentation layer. Business layer is the sum of Business Logic and other components used to add business logic.
* **Presentation layer:** is the layer in which the users interact with an application. Presentation layer contents Shared UI code, Code Behind and Designers used to represent information to user.

### MVC (Model-View-Controller)



Figure 3 – MVC architecture

The above diagram is explained as the following:

* **Model**: Functions that interact with the database or perform complex operations.
* **View**: The view displays the data and also takes input form user. It renders the model data into a form, such as graphics, buttons, and tables, and so on to display to the user. In this system, we use html page.
* **Controller:** The controller handles all requests coming from the view or user interface. The data flows to whole application are controlled by controller. It forwards the request to the appropriate handler. Only the controller is responsible for accessing model and rendering it into various user interfaces.

### Description of System Interface

N/A