Description of MVC:

MVC: This architecture splits the app into three different parts, which are the model, view, and controller. Every part has a different purpose and every part transmits information to each other by displaying views and processed information, handling I/O, and processing user requests.

1. Model - A dynamic data structure that is separate from the user interface. Through model classes, all data logic that the user interacts with is defined. We use models to define data transferred between the view and the controller. For example, our database system has a “User” entity, and we have a model class that defines the attributes of that entity, First and Last name. That data can be displayed, updated, or changed through MVC processing.

This is a data structure that is distinct from the UI. The model has the information that stores data or transfers data from the view or controller. For instance, in the database, there is an entity called “Questions”, and there is a class that contains the necessary parts of the entity, such as the question prompt and the question number. The attributes can be modified using the processing that takes place during MVC procedures.

2. View - The user interface logic of the application; communicates directly with the controller to render views requested from the user. A view can be a data input form that requests to update data using MVC architecture. An example of this is our system’s “User Profile” page with input boxes for first name, last name, etc. that the user can interact with to update their information. Once the form is submitted by the user, it sends the data update request to the controller, which then sends the update to the “User” data model.

The model represents the logic that has an avenue of direct communication with the controllers to generate the views that the user would like to see. For instance, in the “Edit Domains” page, the user can edit the name of the domain that is being used. After the user has completed the information, the view gives an update request to the controller, which updates the repository.

.

3. Controller - Interface between the Model and View components that processes all logic and incoming requests to manipulate data and render views. Our “Dashboard” controller handles all data input requests from the user via view pages, as well as processes requests to update the database's “User” model. The user can view their profile, update their profile information, and submit the request which is received by the controller. The controller then processes the request and communicates with the “User” model to change the information in their profile . Finally, the controller sends the modified data back to the view.

This serves as a proxy between the view and model. The controller is used as the processor so that the views can be shown on the screen as well as modifying data. The “Survey Management Controller” processes all inquiries from the super admin per the web pages. The super admin can view the question, edit the question, and set the question to inactive or active. The model is prompted to edit the question information due to the request of the controller. The last action that the controller does is that it relegates the changed data to the webpage.