

Understanding MVC architecture

Topics : [Codeigniter](#)

Written on [February 29, 2024](#)

MVC (Model-View-Controller) is a software architectural pattern commonly used in web development to separate the concerns of an application into three interconnected components: Model, View, and Controller. Each component has distinct responsibilities, which promotes modularity, maintainability, and scalability in software development.

Here's a breakdown of each component in the MVC architecture:

1. Model

- The Model represents the data and business logic of the application.
- It interacts with the database, performs data manipulation, and enforces business rules.
- In the context of web development, models are often responsible for querying and updating the database, as well as encapsulating data-related operations.
- Models are independent of user interface concerns and do not directly interact with views or controllers.

2. View

- The View represents the presentation layer of the application.
- It is responsible for rendering the user interface and displaying data to the users.
- Views are typically implemented using HTML, CSS, and templating engines.
- In MVC, views receive data from controllers or directly from models and generate the appropriate output for the user.
- Views are passive components and do not contain business logic or data manipulation code.

3. Controller

- The Controller acts as an intermediary between the Model and View components.
- It receives user input, processes requests, and interacts with the model to retrieve or update data.
- Controllers contain application logic related to handling user actions and determining the appropriate response.
- In MVC, controllers orchestrate the flow of the application, invoke model operations, and pass data to the views for rendering.
- Controllers are responsible for handling routing, request/response management, and enforcing application-wide policies.

How MVC Works:

1. **User Interaction:** A user interacts with the application through the user interface (e.g., web

browser), triggering requests for specific actions.

2. **Controller Handling:** The controller receives the user's request, determines the appropriate action to take, and invokes the corresponding methods in the model layer.
3. **Model Processing:** The model performs the requested operations, such as querying the database, updating data, or applying business logic.
4. **Data Presentation:** After processing the request, the model returns data to the controller, which passes it to the view for rendering.
5. **View Rendering:** The view receives the data from the controller and generates the appropriate output for the user, typically in the form of HTML pages or other content types.
6. **Response Sent:** The rendered output is sent back to the user's browser for display, completing the request-response cycle.

© Copyright **Aryatechno**. All Rights Reserved. Written tutorials and materials by [Aryatechno](#)