**Architectural Design Pattern**

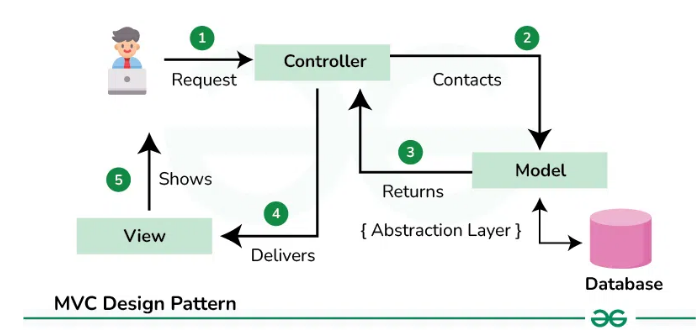
**MVC(Model-View-Controller):**

This architectural pattern is popular and useful for web development or web-based application.

**Definition of MVC:**

The **Model View Controller (MVC)** design pattern specifies that an application consists of a data model, presentation information, and control information. The pattern requires that each of these be separated into different objects.

**Components of the MVC design pattern:**



**Fig: MVC Design Pattern**

**1. Model**

The Model component in the MVC (Model-View-Controller) design pattern represents the data and business logic of an application. It is responsible for managing the application’s data, processing business rules, and responding to requests for information from other components, such as the View and the Controller.

**2. View**

Displays the data from the Model to the user and sends user inputs to the Controller. It is passive and does not directly interact with the Model. Instead, it receives data from the Model and sends user inputs to the Controller for processing.

**3. Controller**

Controller acts as an intermediary between the Model and the View. It handles user input and updates the Model accordingly and updates the View to reflect changes in the Model. It contains application logic, such as input validation and data transformation.

**Communication between the components:**

* **User Interraction with View:**

The user interacts with the view such as clicking button or entering text into a form.

* **View receives user input:**

The view receives the user input and sends this input to the Controller.

* **Controller processes user input:**

The controller processes the user input from the view. It interprets the input, performs any necessary operation (validation, updating the model etc.), and decides how to responds.

* **Controller updates Model:**

The controller updates the model based on the user input or application logic.

* **Model notifies View of changes:**

If the Model changes, it notifies the view.

* **View requests data from Model:**

The view requests data from the Model to update its display.

* **Controller updates View:**

The Controller updates the View based on the changes in the Model or in response to user input.

* **View renders Updated UI:**

The View renders the updated UI based on the changes made by the Controller.

**Benefits of Using the MVC Pattern:**

1. MVC has the feature of scalability that in turn helps the growth of application.
2. The components are easy to maintain because there is less dependency.
3. A model can be reused by multiple views that provides reusability of code.
4. The developers can work with the three layers (Model, View, and Controller) simultaneously.
5. The extending and testing of application are easier.

**MVC Architectural Design Patter for LivSmart:**

Explain the MVC architectural design pattern for LivSmart

1. **Model:**

The Model represents the data layer and handles the logic for the application, such as retrieving data, saving data and applying business logic.

In this projects Models are:

UserModel, ServiceModel, EventModel, BookingModel,PaymentModel, ParkingModel, ComplaintsModel, NotificationModel, SecurityLogModel, EmergencyAlertModel, MarketPlaceModel, DirectoryModel, DashboardModel

1. **View:**

The View represents the UI layer, responsible for displaying data to the user and capturing user inputs. The View in Android can be the Activity, Fragment, or custom UI components like layouts and widgets.

It communicates with the Controller to receive updates on the data and display it.

In this project Views are:

RegistrationView, LoginView, ServiceView, PaymentView, EventView, BookingView, ProfileView, DashBoardView, ParkingView, ComplaintsView, NotificationView, SecurityLogView, EmergencyAlertView, MarketPlaceView, DirectoryView

1. **Controller:**

The Controller acts as the intermediary between the View and the Model. It responds to user input, interacts with the Model to fetch or update data, and updates the View accordingly.

In this project Controllers are:

UserController, ServiceController, EventController, BookingController, PaymentController,ParkingController,ComplaintsController,NotificationController,SecurityLogController,EmergencyAlertController,MarketPlaceController, DirectoryController, DashboardController