**What is Model View Controller (MVC)?**

MVC is a software design pattern or software development methodology. The main objective of MVC is promote the code usability and impl

In a typical application you will find these three fundamental parts:

* Data (Model)- (i.e deal with data to carry which use setter and getter means that is model)
* An interface to view and modify the data (View)-(i.e. UI part which shows data, template + model values)
* Operations that can be performed on the data (Controller)- (i.e. Business logic)

The MVC pattern, in a nutshell, is this:

1. The **model** represents the data, and does nothing else. The model does NOT depend on the controller or the view.
2. The **view** displays the model data, and sends user actions (e.g. button clicks) to the controller. The view can:
   * be independent of both the model and the controller; or
   * actually **be** the controller, and therefore depend on the model.
3. The **controller** provides model data to the view, and interprets user actions such as button clicks. The controller depends on the view and the model. In some cases, the controller and the view are the same object.

i.e. perform operation on model and then sent that model to view for showing on browser that why we say Controller is depend on Model and view.

Rule 1 is the golden rule of MVC so I'll repeat it:

**The model represents the data, and does nothing else. The model does NOT depend on the controller or the view.**

Let's take an address book application as an example. The model is a list of Person objects, the view is a GUI window that displays the list of people, and the controller handles actions such as "Delete person", "Add person", "Email person", etc. The following example does not use MVC because the model depends on the view.

**Advantages of MVC:-**

The primary advantage of the MVC design pattern is this:

* **MVC makes model classes reusable without modification.**

The primary advantage of the MVC design pattern is this:

* **MVC makes model classes reusable without modification.**

The model-view-controller pattern proposes three main components or objects to be used in software development:

* A *Model* , which represents the underlying, logical structure of data in a software application and the high-level class associated with it. This object model does not contain any information about the user interface.
* A *View* , which is a collection of classes representing the elements in the user interface (all of the things the user can see and respond to on the screen, such as buttons, display boxes, and so forth)
* A *Controller* , which represents the classes connecting the model and the view, and is used to communicate between classes in the model and view.

**Model view controller** is a software architecture design pattern.  It provides solution to layer an application by separating three concerns business, presentation and control flow.

* The **Model** can be some DAO layer or some Service Layers which give some information about request or requested information or Model can be a POJO which encapsulates the application data given by the controller.
* The **View** is responsible for rendering the model data and in general it generates HTML output that the client's browser can interpret.
* The **Controller** is responsible for processing user requests and building appropriate model and passes it to the view for rendering.