CS603 – Web Engineering

PREPARED BY: DR. REEMA PATEL

MVC Framework

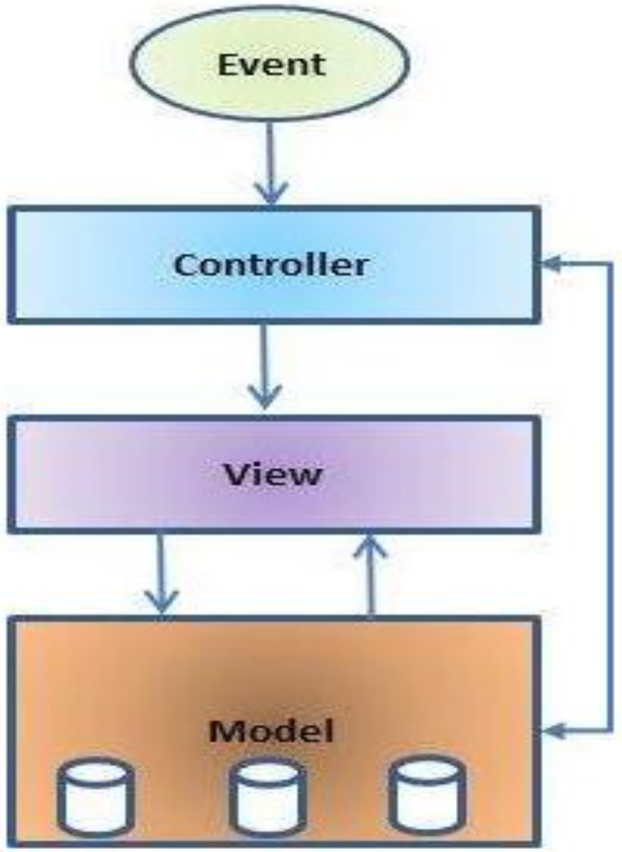
Model View Controller

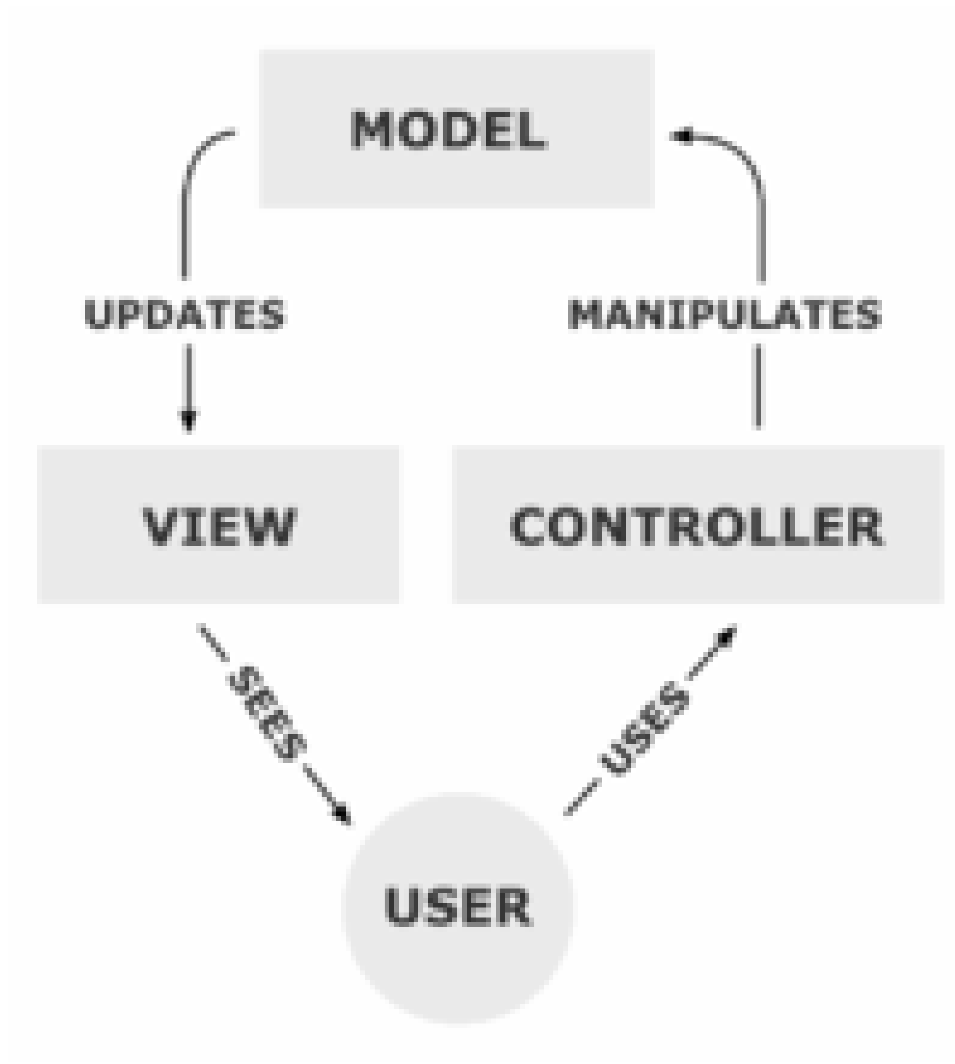
- **M**odel **V**iew **C**ontroller or MVC as it is popularly called, is a software design pattern for developing web applications.

- **Model–view–controller** (**MVC**) is a software architecture pattern which separates the representation of information from the user's interaction with it .

Architecture of MVC

- A Model View Controller pattern is made up of the followin three components:
- The Business Layer (Model logic)
- The Display Layer (View logic)
- The Input Control (Controller logic)




MODEL
UPDATES
MANIPULATES
VIEW
CONTROLLER
SEES
USES
USER

Model

- The Model is the part that does the work--it models the actual problem being solved
- The Model should be independent of both the Controller and the View
  - But it provides services (methods) for them to use
- Independence gives flexibility, robustness

Model

- The model is responsible for managing the data of the application.
- It responds to the request from the view and it also responds to instructions from the controller to update itself
- It is the lowest level of the pattern which is responsible for maintaining data.
- The Model represents the application core (for instance a list of database records).
- It is also called the domain layer

View

- The View shows what the Model is doing
- The View is a passive observer; it should not affect the model
- The Model should be independent of the View, but (but it can provide access methods)
- The View should not display what the Controller thinks is happening

View

- The View displays the data (the database records).
- A **view** requests information from the model, that it needs to generate an output representation.
- MVC is often seen in web applications, where the view is the HTML page.

Controller

- The Controller decides what the model is to do
- Often, the user is put in control by means of a GUI
  - in this case, the GUI and the Controller are often the same
- The Controller and the Model can almost always be separated (what to do versus how to do it)
- The design of the Controller depends on the Model
- The Model should not depend on the Controller

Controller

- **The Controller** is the part of the application that handles user interaction.
- Typically controllers read data from a view, control user input, and send input data to the model.
- It handles the input, typically user actions and may invoke changes on the model and view.

Workflow in MVC - Example

Though MVC comes in different flavours, the control flow generally works as follows:

1. The user interacts with the user interface in some way (e.g., user presses a button)
2. A controller handles the input event from the user interface, often via a registered handler or callback.
3. The controller accesses the model, possibly updating it in a way appropriate to the user's action (e.g., controller updates user's shopping cart).

Workflow in MVC - Example

4. A view uses the model to generate an appropriate user interface (e.g., view produces a screen listing the shopping cart contents).

The view gets its own data from the model. The model has no direct knowledge of the view.

Dependence hierarchy

- There is usually a kind of hierarchy in the MVC pattern.
- The Model knows only about itself.
- That is, the source code of the Model has no references to either the View or Controller.

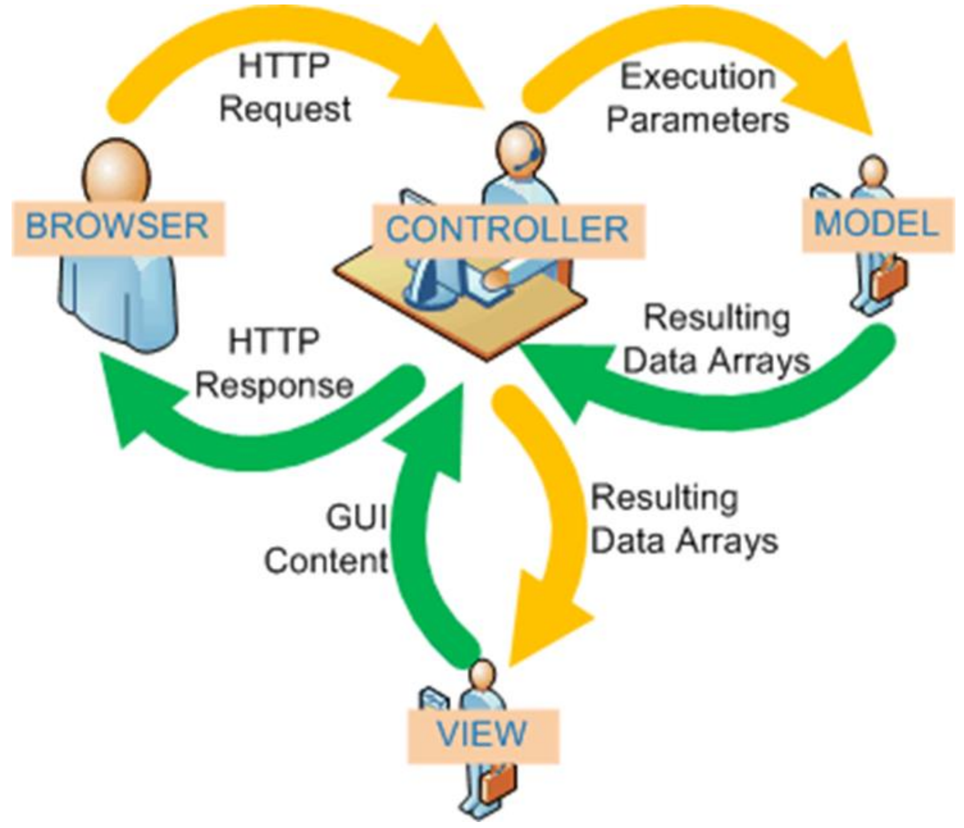
Dependence hierarchy

- The View however, knows about the Model. It will poll the Model about the state, to know what to display.
- That way, the View can display something that is based on what the Model has done.
- But the View knows nothing about the Controller.
- The Controller knows about both the Model and the View.

Why dependence hierarchy is used?

- The reason to keep it this way is to minimize dependencies.
- No matter how the View class is modified, the Model will still work.
- Even if the system is moved from a desktop operating system to a smart phone, the Model can be moved with no changes.
- But the View probably needs to be updated, as will the Controller.

Working of MVC in web application



Normal Web Page vs. MVC

- The MVC programming model is a lighter alternative to traditional Web Page/Forms.
- It is a lightweight, highly testable framework, integrated with all existing features, such as Security, and Authentication.

Advantages

- Clear separation between presentation logic and business logic.
- Each object in mvc have distinct responsibilities.
- parallel development
- easy to maintain and future enhancements
- All objects and classes are independent of each other.

Disadvantages

- Increased complexity
- Inefficiency of data access in view
- Difficulty of using MVC with modern user interface too.
- For parallel development there is a needed multiple programmers.
- Knowledge on multiple technologies is required.