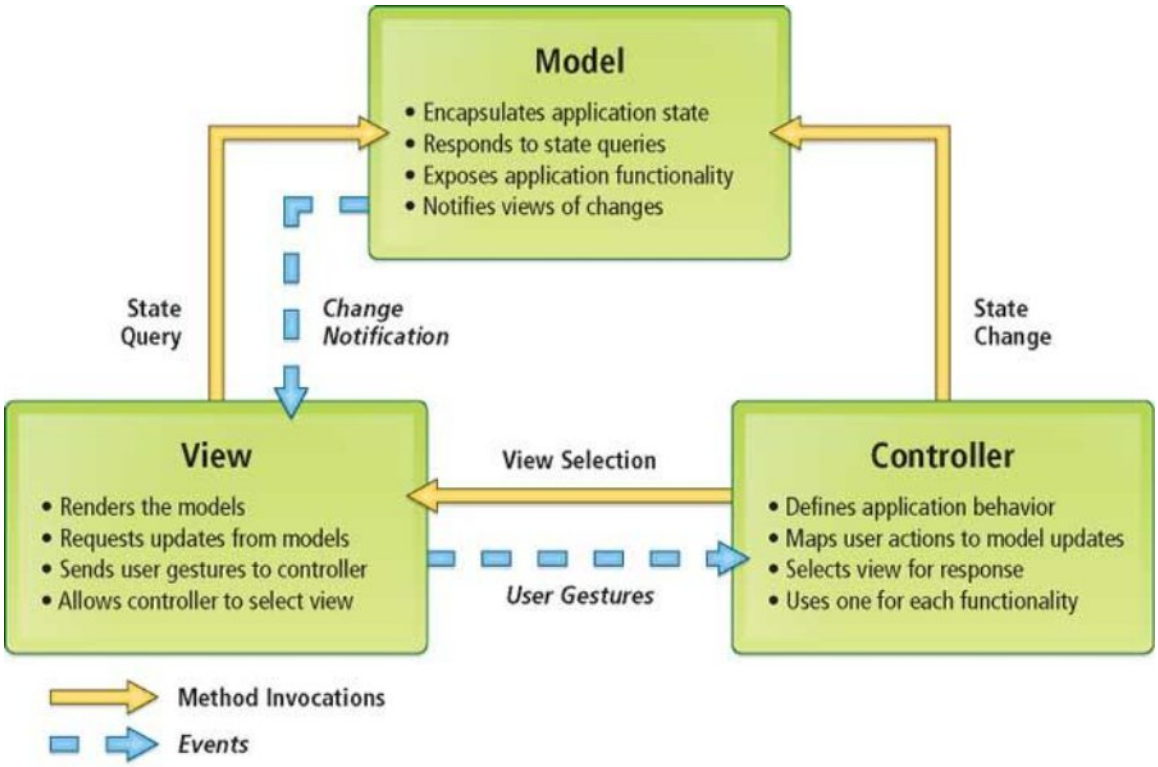
MVC ARCHITECTURE



Model View Controller, an architectural pattern that separates an application into three main components: the model, the view, and the controller

A preconfigured development approach eliminates common design decisions from the outset. Microsoft has created technology that allows us to develop very rapidly in a predefined structured application that makes a lot of the decisions for us thus speeding up the development cycle.

Solving a Classic Problem

To illustrate what MVC architecture offers developers, when compared to the classic Active Server Pages (ASP) and ASP.NET Web Forms programming models. Classic ASP and Web Forms applications tend to violate the programming paradigm known as the "separation of concerns." As a code base evolves in classic ASP programming or Web Forms, business logic becomes mixed with presentation logic in the user interface, data access bleeds into the business layer, and so forth. Markup for these applications can get messy and noncompliant since the developer does not exercise full control over the emitted HTML.

The beauty of Model View Controller architecture in general is that it enforces from project inception a bifurcation of data, logic, and UI without compromising the functionality end users require.

MVC separates typical developer concerns, isolating them into groups of information within the application where they should be.

Application data provided by a data access layer stays within the domain layer (or the **M**odel of **MVC**).

User interface elements for rendering and presentation are explicitly isolated in one to many views (the **V**iew).

Set of controller classes routes the trip from database to view and back again (the **C**ontroller). Binary, JSON (JavaScript Object Notation), XML, or code on demand supporting multiple representations of data or features is no problem with ASP.NET MVC because it separates concerns while giving the developer full control over the HTML.

However, many applications are data driven and require advanced mapping functionality. Tackling those applications in ASP.NET MVC makes the most sense.

MVC facilitates much more robust and reliable unit testing than Web Forms or classic ASP programming typically allows. Plain and simple, ASP.NET MVC was designed and built to produce testable software.