**SEMINAR REPORT:**

**MODEL VIEW CONTROLLER**

**SUBMITTED BY,**

**ALEESHA KURIAN**

**S5 MCA**

Over the years, software development has gone through many changes. One of the biggest change that happened in the recent years, is the use of MVC pattern for developing software or web application. The Model–view–controller shortly known as MVC is a software architectural design for implementing user interfaces on computers. The MVC pattern is a great architecture no matter whatever the language you are using for the development.

MVC patterns separate input, processing, and output of an application. This model divided into three interconnected parts called the model, the view, and the controller. All of the three above given components are built to handle some specific development aspects of any web or software application.

In the MVC development, controller receives all requests for the application and then instruct the model to prepare any information required by the view. The view uses that data prepared by the controller to bring the final output.

**Three levels of MVC Model:**

**Model:**

This level is very important as it represents the data to the user. This level defines where the application’s data objects are stored. The model doesn’t know anything about views and controllers. So, whenever there are changes done in the model it will automatically notify observers that the changes are made. The model may be a single object or a structure of objects.

**Views:**

A view is a visual representation of the MVC model. This level creates an interface to show the actual output to the user. However, a view will not display anything itself. It is the controller or model that tells view what to display to the user. It also handles requests from the user and informs controller. A view is connected to its model and gets the data necessary for the presentation by asking certain questions. Sometimes, it also updates the model by sending appropriate messages. All these questions and messages are sent back to the model in such an easy terminology that it can easily understand the information sent by model or a controller.

**Controller:**

Controller is a level which acts like a brain of the entire MVC system.. A controller also acts as a link between a user and the system. It provides the user with input by providing appropriate views to present it appropriately on the screen. The controller understands user output, converts it into the appropriate messages and passes the same to views.

**Advantages of MVC Model:**

* MVC enforces separation that reduces complexity of project structure.
* MVC contain three main parts model, view and controller that help to manage complexity of application.
* MVC provides better support to TDD (Test driven development). TDD is related to the test first programming concepts of extreme programming. It helps us to reduced time in reworks and helps create loosely coupled code.

**Disadvantages**:

* ASP.Net MVC sites require more time to develop over traditional ASP.Net applications.
* There is no builtin rich UI control.
* In MVC, there is extensive use of jQuery / JavaScript / AJAX, it makes the form (application) complicated.
* ASP.NET MVC is difficult to learn, it requires a deep knowledge of the pattern.
* View state and post-back will not work as designed so that we cannot use a control such as Grid view, repeater and datalist control.

**Implementation of MVC in ASP.NET**

Step 1: - Select the project template

Step 2: - Select the appropriate ASP.NET One options

Step 3: - Add Controller

Step 4: - Add Views

Step 5: - Connect the view to the controller

Step 6: - Run the program

**Conclusion**  
The selection of technology varies from person to person. MVC has a few advantages, such as more control over HTML; support of TDD, builtin support of SEO based URLs, and so on. If you want these benefits then you can choose the MVC model. A Web form's abstraction of the Web provides a stateful environment, while MVC is consistent with the Stateless nature of the web.  I think, testability, refactoring capability and maintainability are the main factors of any application. The MVC model allows us to create highly testable and loosely coupled applications with Test Driven Development (TDD) and Separation of Concerns (SoC).