**Course Overview:**

1. Why Windows PowerShell?
2. History
3. PowerShell or CMD
4. .NET Integration
5. Getting and Installing PowerShell
6. Configuring the console
7. PowerShell Profiles
8. PowerShell Commands
9. Pipeline
10. Creating a PowerShell Script
11. Executing a PowerShell Script
12. Variable and Constants
13. Comparison Operators
14. If Statements and Looping
15. Regular Expressions
16. Escape Character
17. Using Parameters with Script
18. PowerShell Providers
19. Working with files and folders
20. Reading and writing data
21. HTML/CSV/XML
22. Working with Processes
23. PowerShell and WMI
24. PowerShell 5.0
25. **Why Windows PowerShell?**

* Think way back what has happened in the world of Windows Computer administration. First, we had few admins, and few computers for administration purpose. As time pass by, we maintained the administration staff and added lot of computers. But Budget, cost, computations overhead came in. As the time progressed, we came to know that there are very few admins and way many computers. The process of administering, managing, and maintaining our Network got lot more complex.

The Evolution of Windows Administration:

1. The windows OS conquered the corporate and home desktop first, then the servers.
2. Administration was GUI based, that is it was Easy on single Machine
3. Overtime, environment grew larger as the IT department Shrunk in size.
4. Graphical Tools are difficult to automate
5. On the contrary, Command Line tools are easy to automate. For example: DOS, sequenced commands in a batch file. Relatively easy to perform multiple actions in a script. But the available actions were extremely limited.

***So, what exactly is PowerShell?***

PowerShell provides command line tools for windows administrator, similar to the tools used by UNIX/LINUX admins. It also adds functionalities (based on windows operating system, .Net framework)

PowerShell provides the following:

Cmdlets for performing common system administration tasks to manage Registry, Services, Processes, Event Logs, Windows Management Instrumentation (WMI). It is a task-based scripting language and support for existing scripts and command line tools. It has consistent design, common syntax, naming convention and powerful feature of pipelining. We also get simplified command-based capability of navigating the OS. That is the registry and other data stores can be accessed by using the same techniques that are used to navigate the file system. Powerful Object Manipulation capabilities, Objects can be directly manipulated or sent to other tools or databases. It has Extensible Interface that is Independent software vendors and enterprise developers can build the custom tools and utilities to administer their Software.