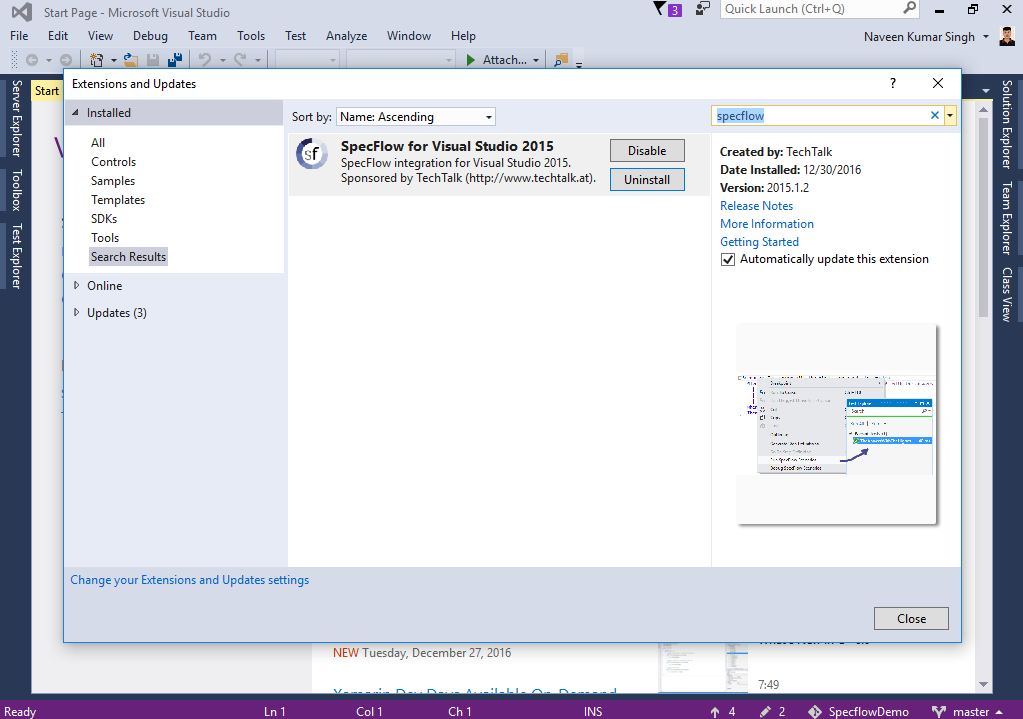
**racticing ATDD using SpecFlow, Nunit and C#**

I will be using Visual Studio 2015 with C# and will use class library project to start with. *Will demonstrate how to practice while creating MVC application with Selenium and Coded UI in upcoming blogs.*

**Step 1 – Install SpecFlow**

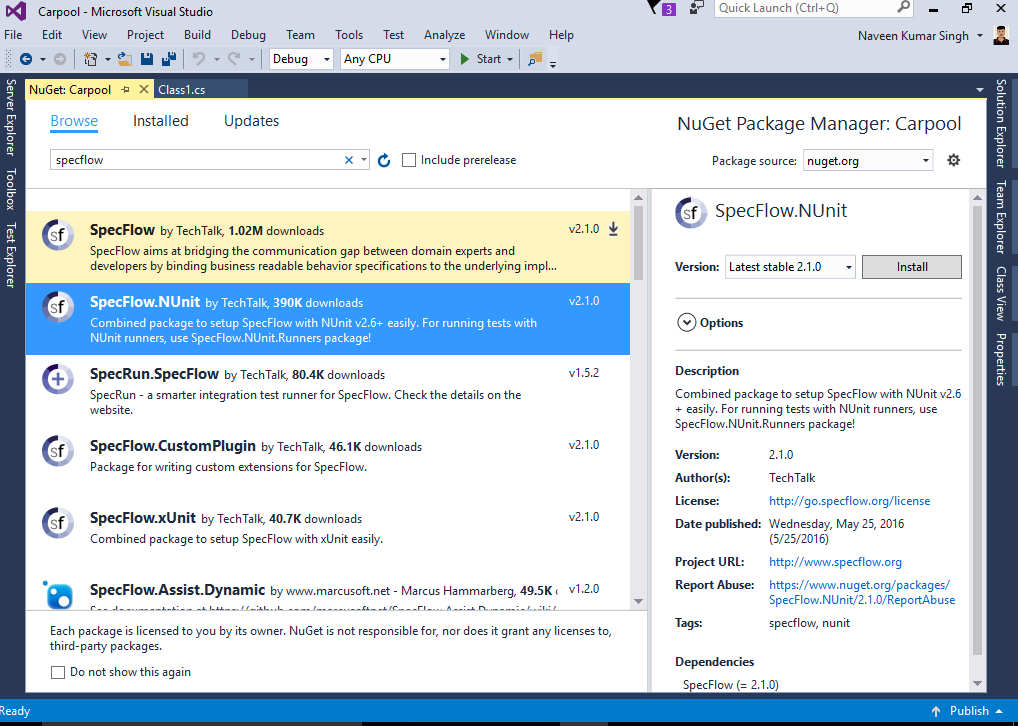
Open Visual Studio and go to Tools -> Extensions and Updates and search “SpecFlow for Visual Studio 2015”

[](https://www.agilemania.com/wp-content/uploads/2017/01/specflow-download.png)

After installation, restart visual studio and create your project by clicking “New Project”. I am going to name it “**Carpool**”.

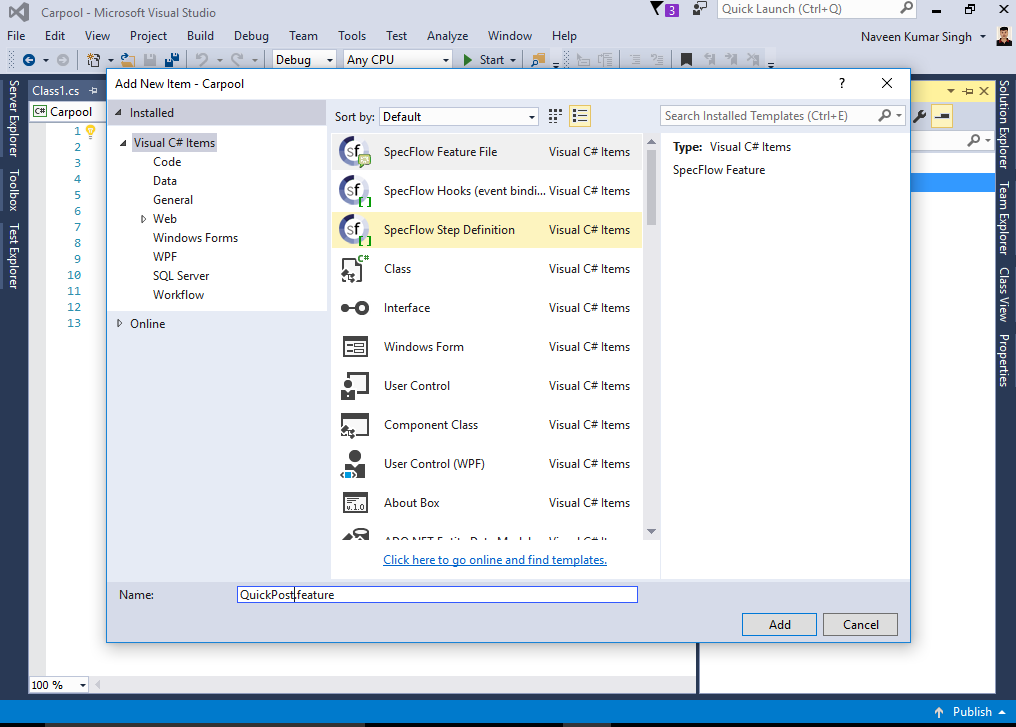
**Step 2 – Add SpecFlow and Nunit references.**

Add SpecFlow and Nunit reference to your project via “Manage NuGet Packages” like below.

[](https://www.agilemania.com/wp-content/uploads/2017/01/specflow-and-nunit-references.png)

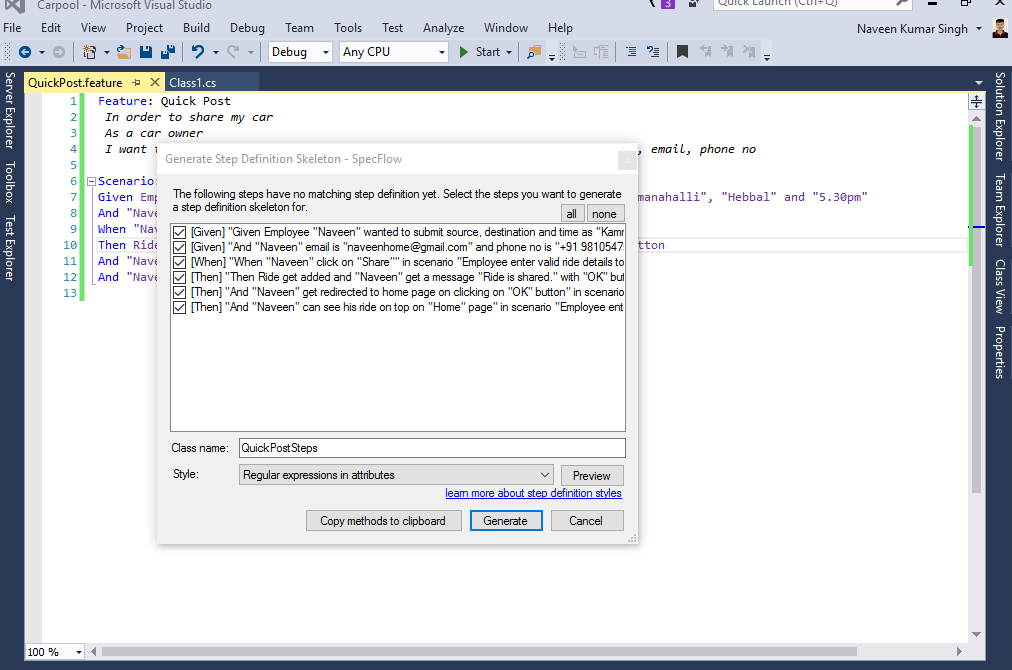
**Step 3 – Create feature file**

Add first feature file to Project -> Add -> New Items and you will see option to create new feature file like below and name it QuickPost.feature. Remove all default content from feature file and copy above written feature.

[](https://www.agilemania.com/wp-content/uploads/2017/01/quickpostfeature.png)

**Step 4 – Generate Step definition**

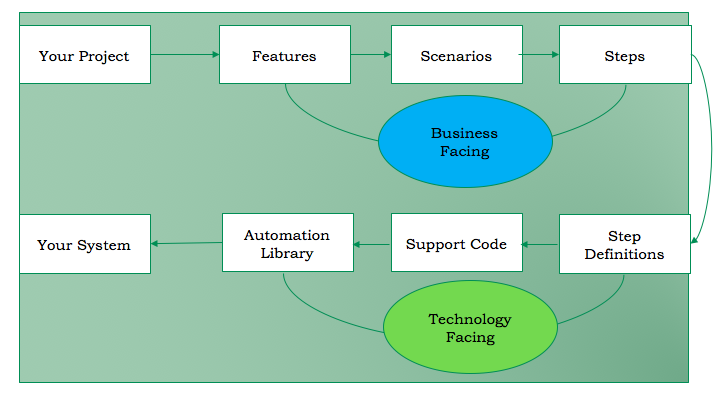
Right click on feature file and you will get option “Generate Step Definitions”. You will get below screen and click “Generate” here. It will ask you to save your step file on default location (same folder where you have feature file). Save it at default location and you will find a file name “QuickPostSteps.cs” in your solution explorer.

[](https://www.agilemania.com/wp-content/uploads/2017/01/quickpoststeps.png)

**What is Step Definitions in SpecFlow?**

In order to test our scenario, we need to create step definitions that bind the statements in the test scenario to the application code. SpecFlow automatically generate a skeleton for the automation code that we can extend as necessary. Steps are not one to one binding and it uses regular expression. That means if there are similar steps with same statement in another feature file or in another scenario then it will reuse.

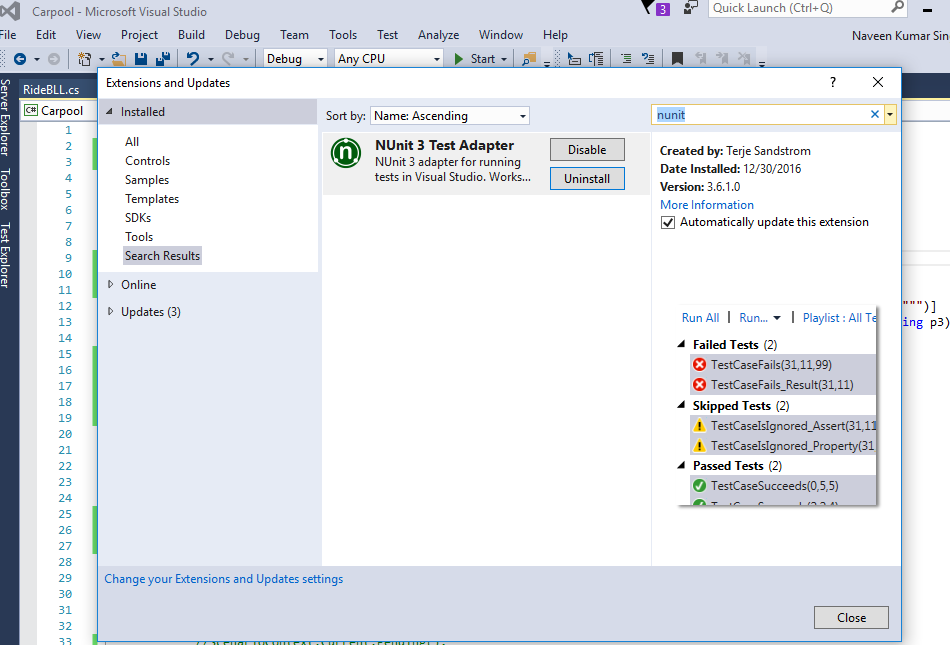
Below ATDD lifecycle explains steps involved in practicing ATDD methodology wherein some steps required business involvement and rest can be performed by development team.

[](https://www.agilemania.com/wp-content/uploads/2017/01/atdd-cycle.png)

**Step 5 – Run your first scenario**

Go to Test -> Window -> Test Explorer and you will find your tests listed in test explorer. If your test not showing that means you need to install an extension called “Nunit 3 Test Adaptor”.

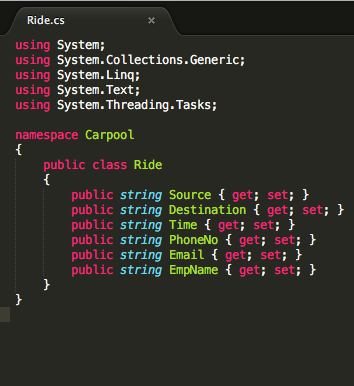
To install it go to Tools -> Extensions and Updates and search “Nunit 3 Test Adaptor”. After installation, restart visual studio and build your solution to see your tests in test explorer.

[](https://www.agilemania.com/wp-content/uploads/2017/01/nunit-test-adaptor.png)

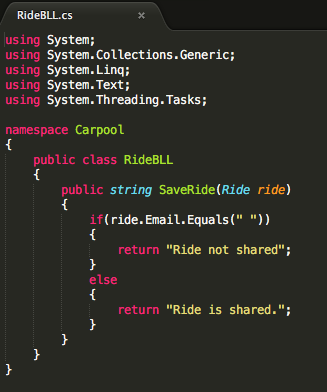
**Step 6 – Writing production code to pass our 1st scenario**

I will use Nunit for assertion and will demonstrate how to pass scenario by adding minimum code just to demonstrate capability of SpecFlow.

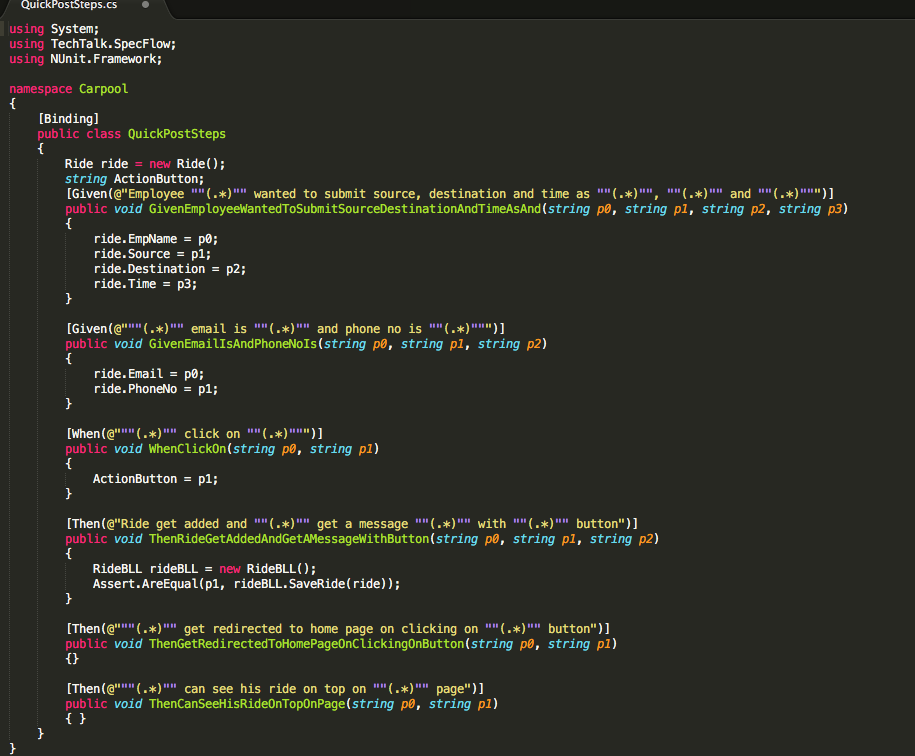
Add a model class called “Ride.cs”

[](https://www.agilemania.com/wp-content/uploads/2017/01/ride-class.png)

Add another Business Logic class “RideBLL.cs”

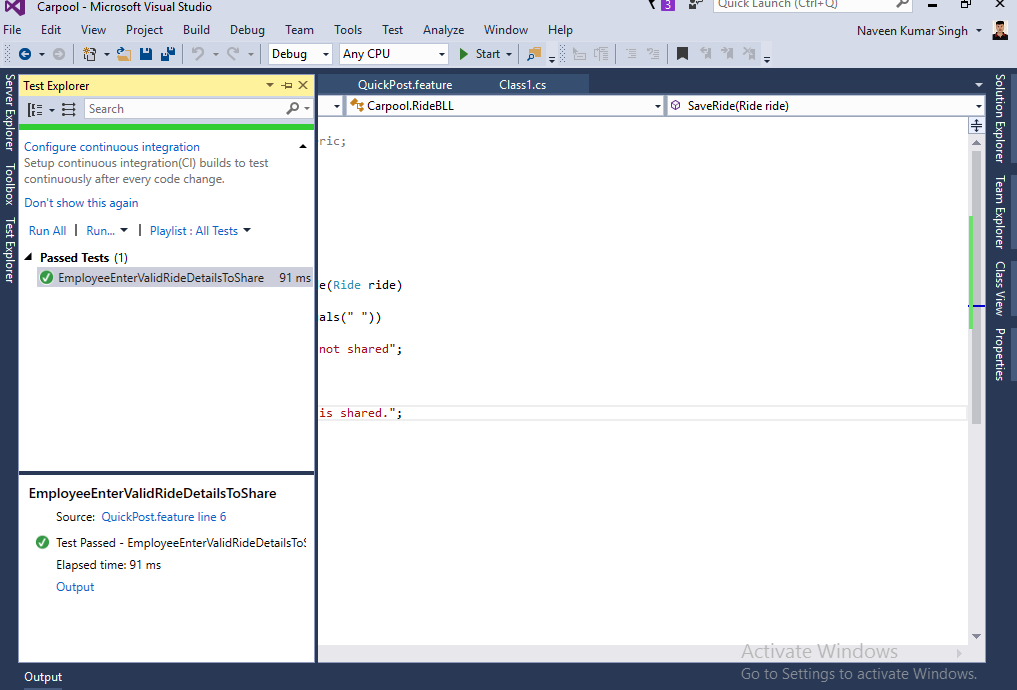
[](https://www.agilemania.com/wp-content/uploads/2017/01/ridebll-class.png)

Make changes in steps file like below: –

[](https://www.agilemania.com/wp-content/uploads/2017/01/steps-file.png)

**Step 7 – Run Tests again**

Build your solution again, go to test explorer and click on run all. Test will pass successfully like below.

[](https://www.agilemania.com/wp-content/uploads/2017/01/test-pass.png)