**Dotnet Core 2.0 Web API with MongoDB**

**Configuring MongoDB Server**

1. To set up the mongodb server and create a user to work on. Create a mongod.conf file in the mongodb root folder and open it in the editor. Add the following lines of code in the file.

# mongod.conf

# Where and how to store data.

storage:

dbPath: C:\mongodb\data

journal:

enabled: true

# network interfaces

net:

port: 27017

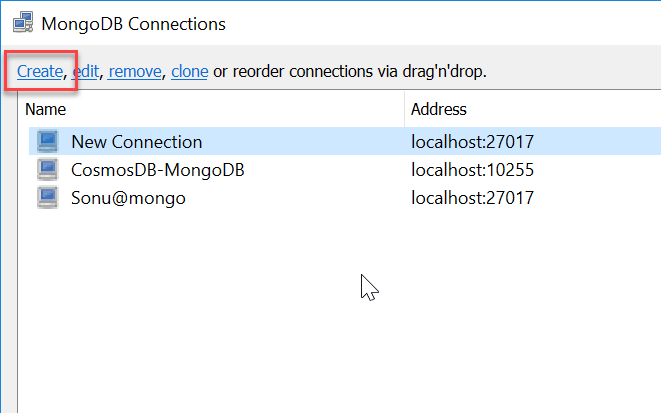
bindIp: 127.0.0.1 # Listen to local interface only, comment to listen on all interfaces.

1. Create a batch\_file with the name mongo\_start.cmd to start mongodb server. Open the file in text editor and add the following code.

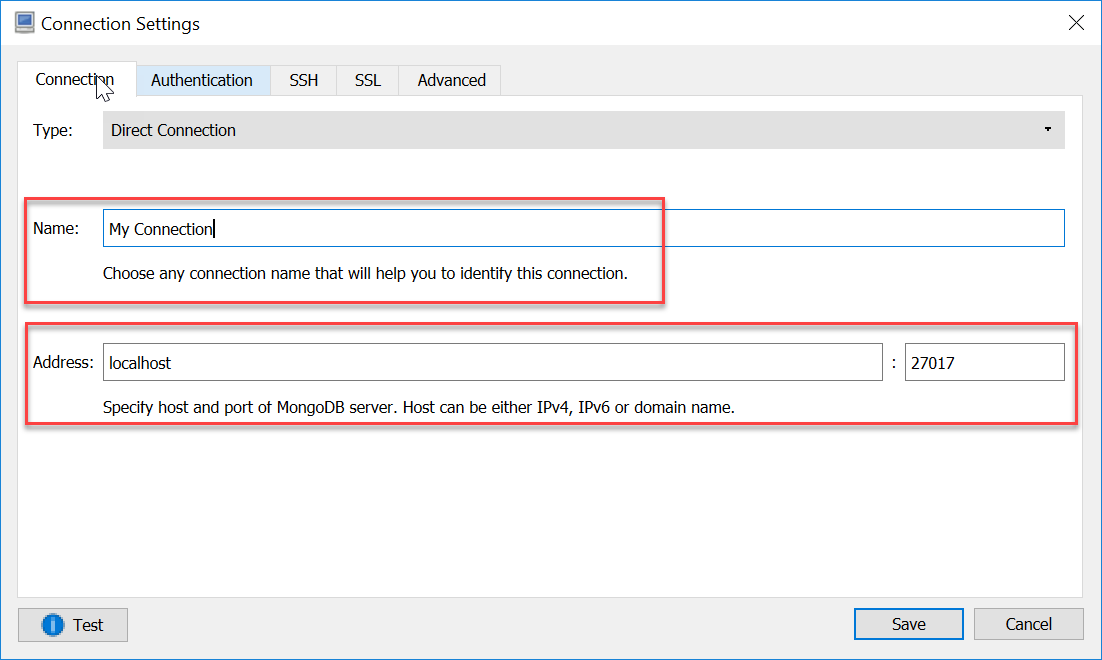
**c:\mongodb\bin\mongod --config "C:\mongodb\mongod.conf"**

**pause>nul**

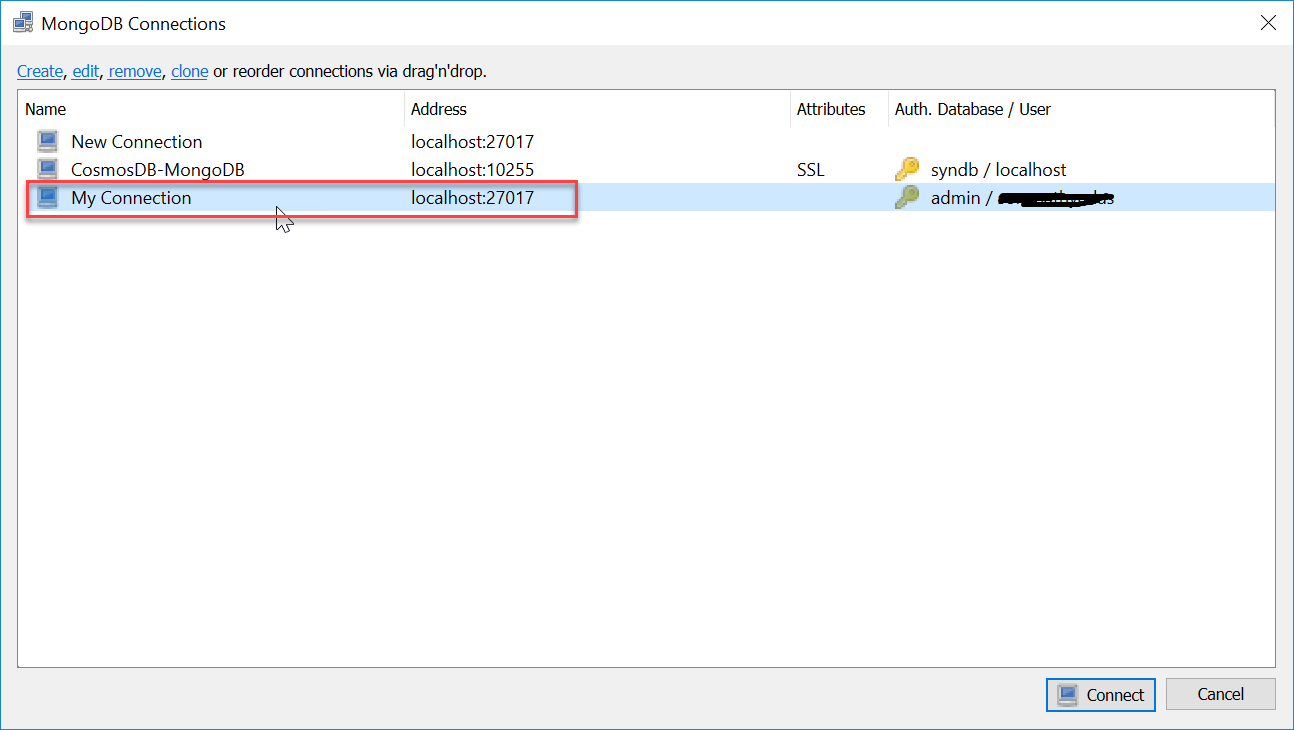
1. Save and close the file.
2. Double click the file to execute the bat file. It starts the mongodb server.
3. Open Rob3T and click on the connect button.



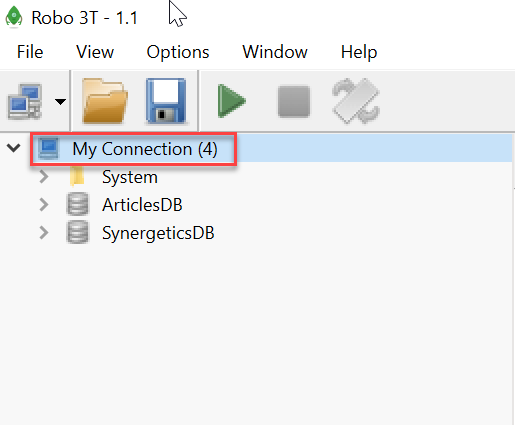
1. Click on the Create link which opens a new dialog window, Under “Connection” tab, specify the connection name and address. Address can be given as “localhost” and port number should be 27017.



1. Click on the “Test” button to check the connectivity. On success it shows a success dialog box. Click on “save” button to save connection and go back to the previous window. Then select the connection which you have created and click “connect” button.



1. You can see the databases in the explorer



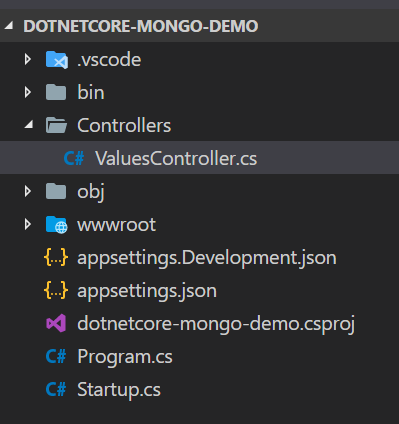
**Connecting MongoDB with ASP.NET Core 2.0**

You are now ready with the environment. You can start creating the .NET Core application.

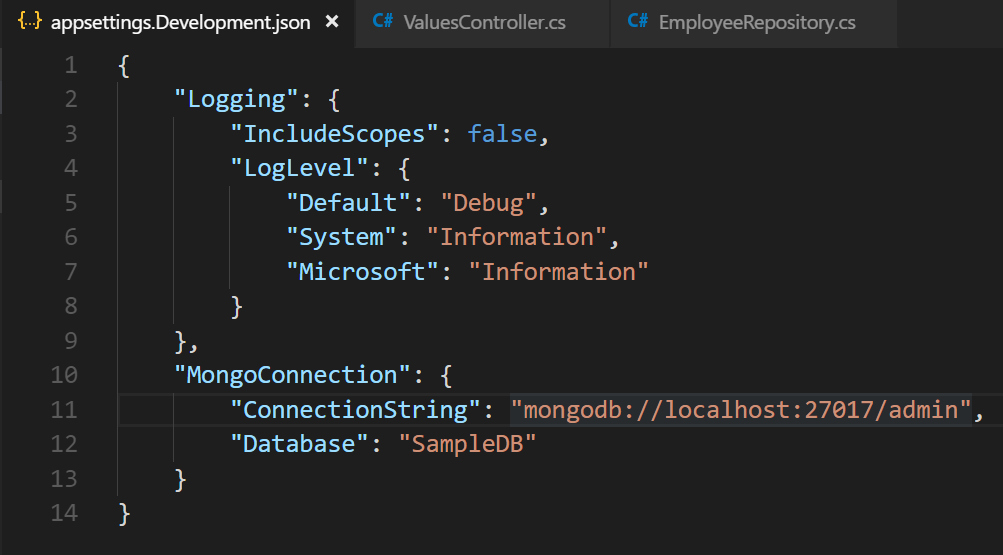
1. Open command prompt and run the following command to create a new project.

**dotnet new webapi -o dotnetcore-mongo-demo**

1. Open the project folder in VS Code



1. Open the “appsettings.Development.json” file and add the following configuration to that.

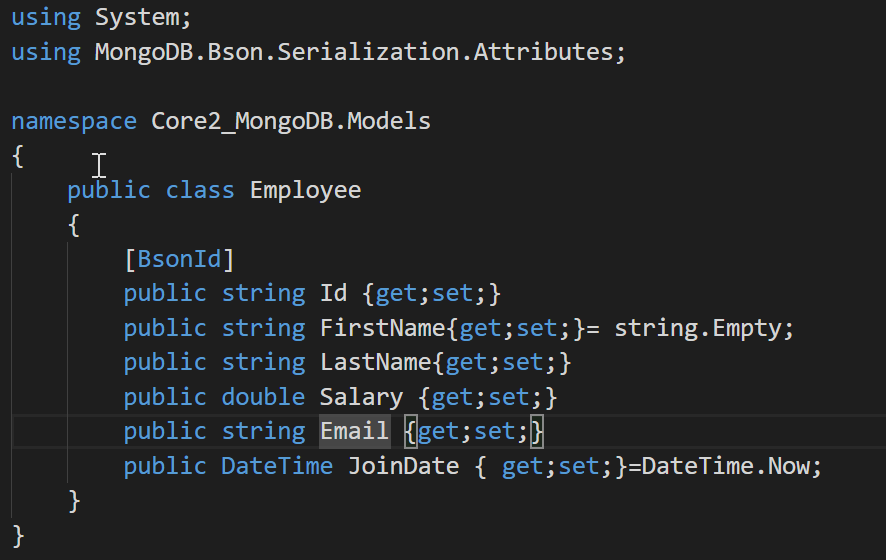


1. You need to install the mongodb driver for .NET Core. To install the nuget package for mongodb driver, Open the integrated terminal and execute the commands.

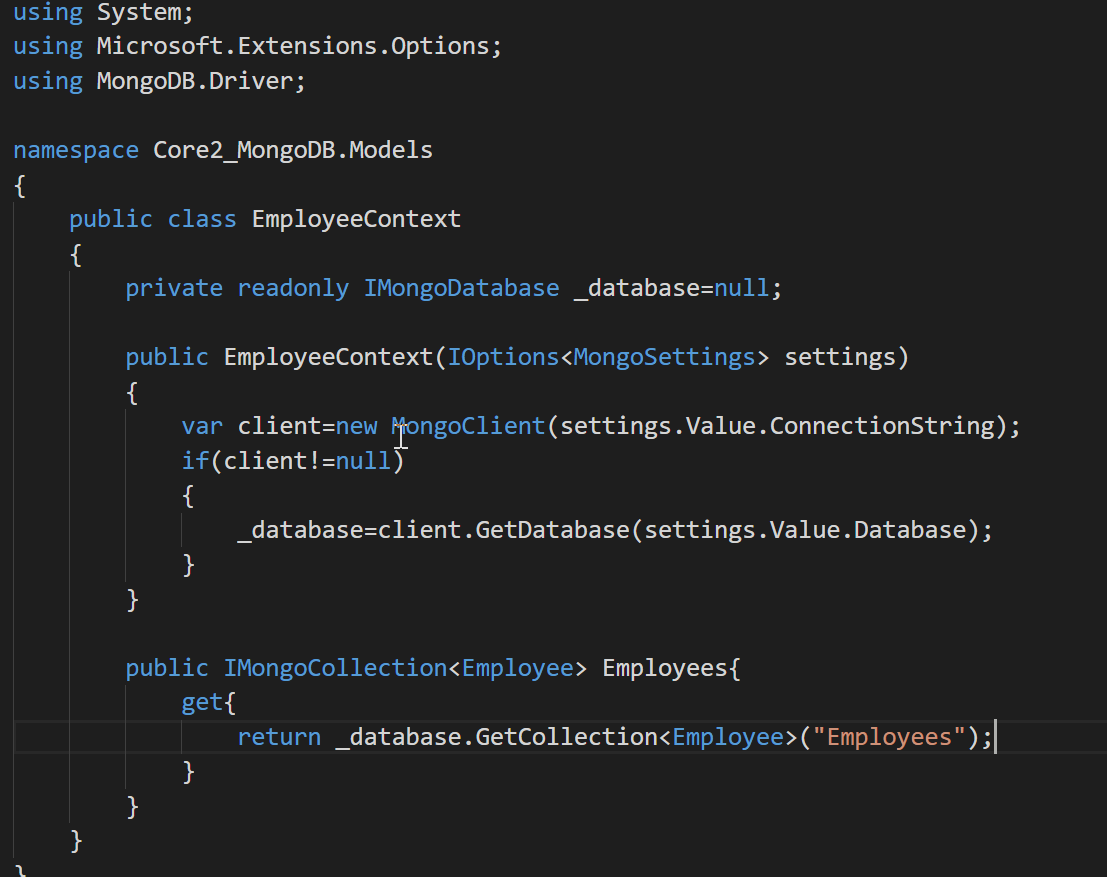
**dotnet add package MongoDB.Driver --version=”2.5.0”**

**dotnet add package MongoDB.Driver.Core --version=”2.5.0”**

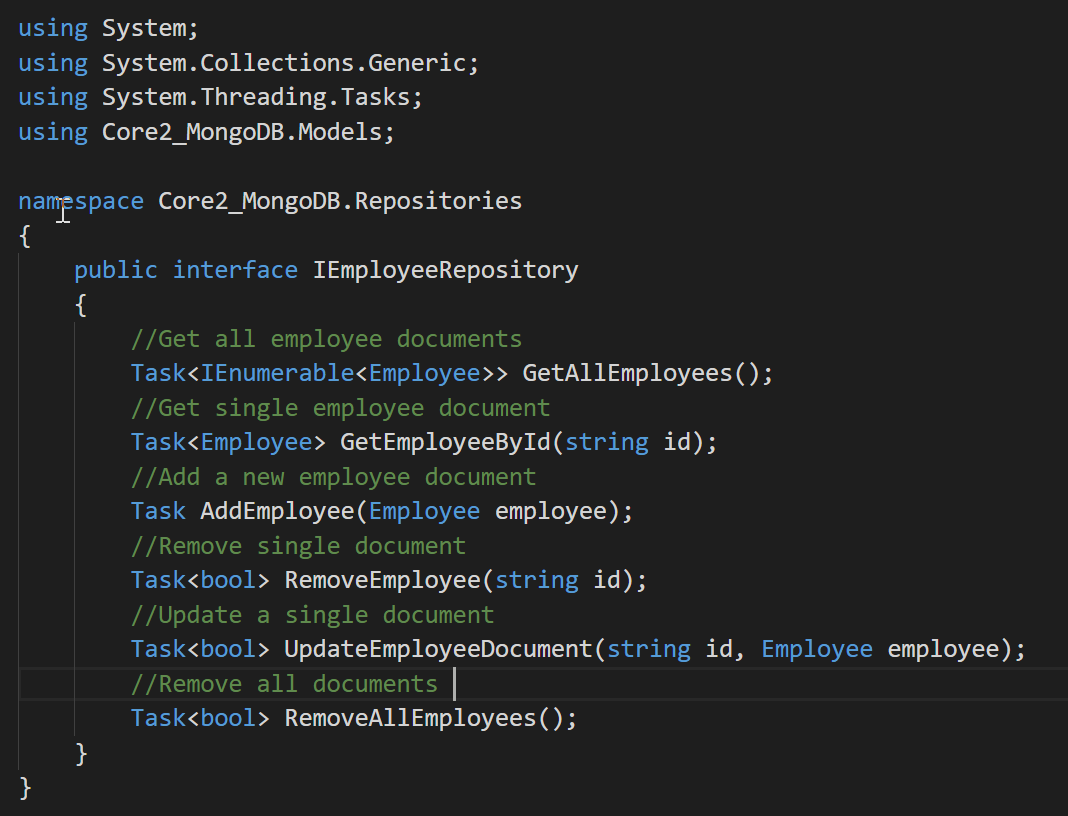
1. Create a MongoDB document model class. Create a new class in the Models folder with the name “Employee”.



1. Now, we need to set up the Context class. In order to keep the functions for accessing the database in a distinct place, we will add a EmployeeContext class. This will use the Settings defined above.



1. Using a repository interface, we will implement the functions needed to manage the Notes. These will also use Dependency Injection (DI) to be easily access from the application



1. The access to database will be asynchronous. We are using here the new driver, which offers a full async stack. Here is the full implementation, for all basic CRUD operations:
2. d