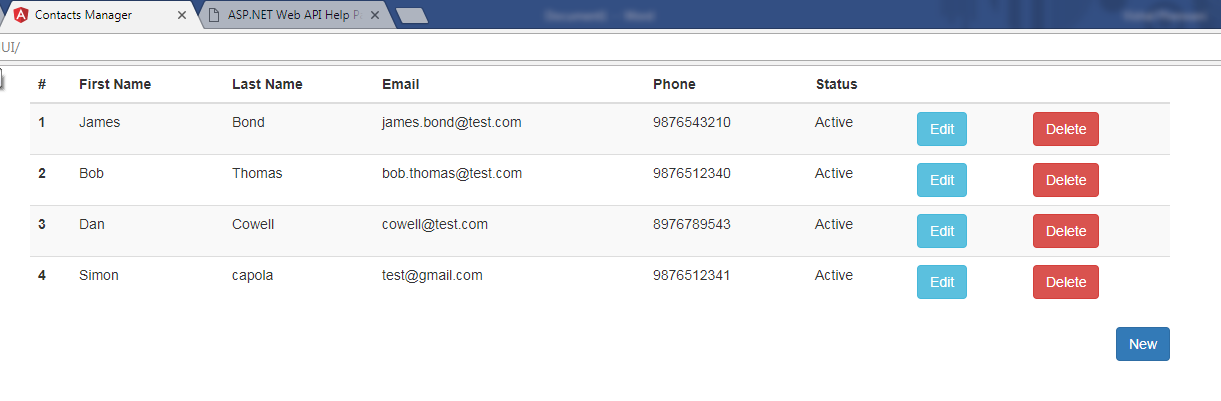
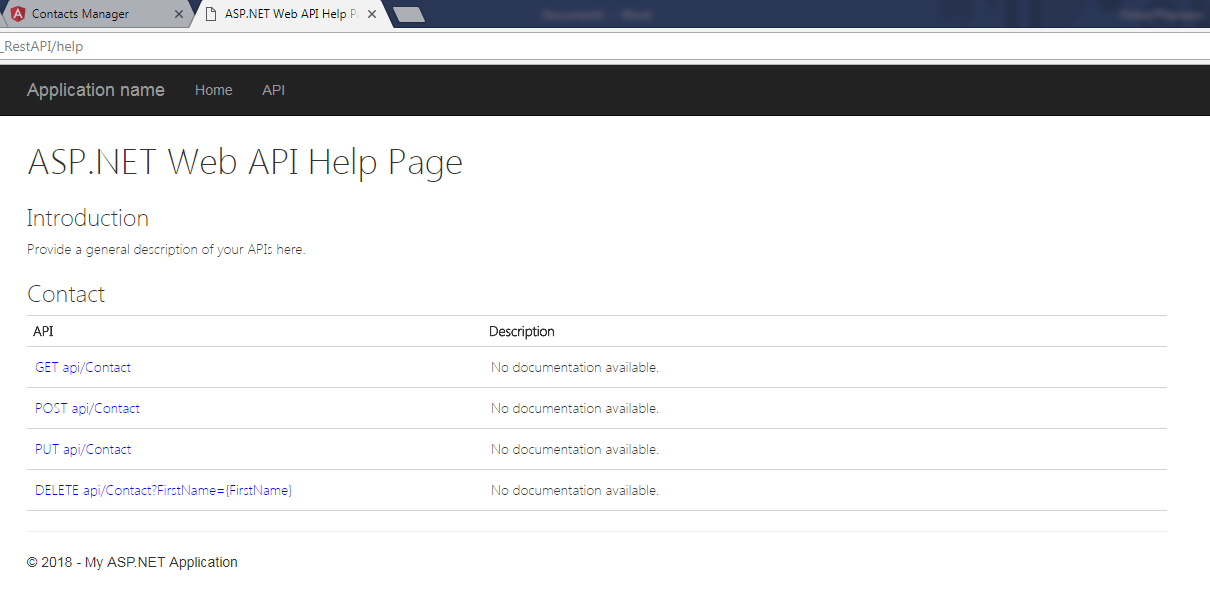
# Contact Manager Solution

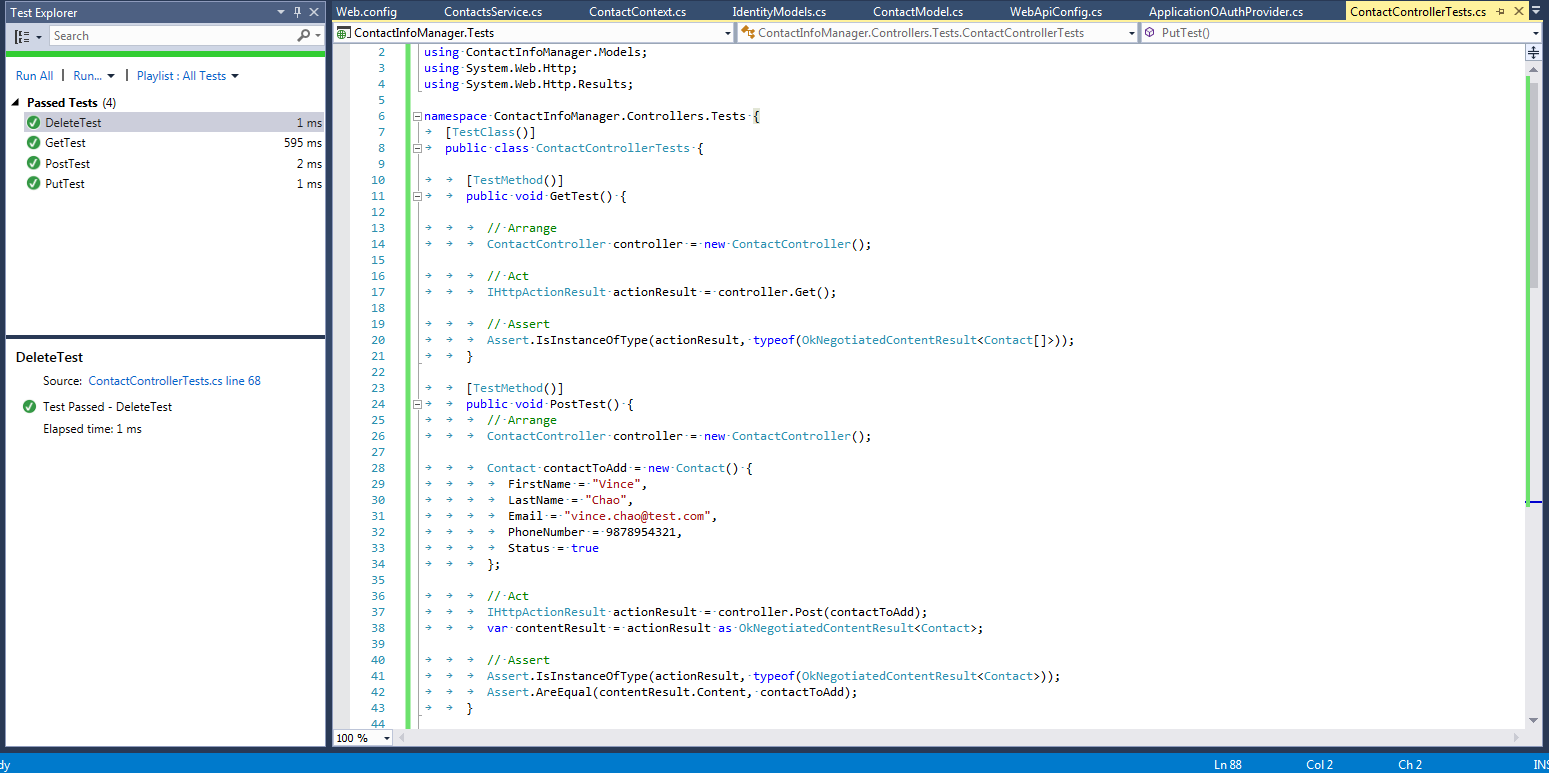
1. The UI application: it is an angular application consuming the .NET REST API –



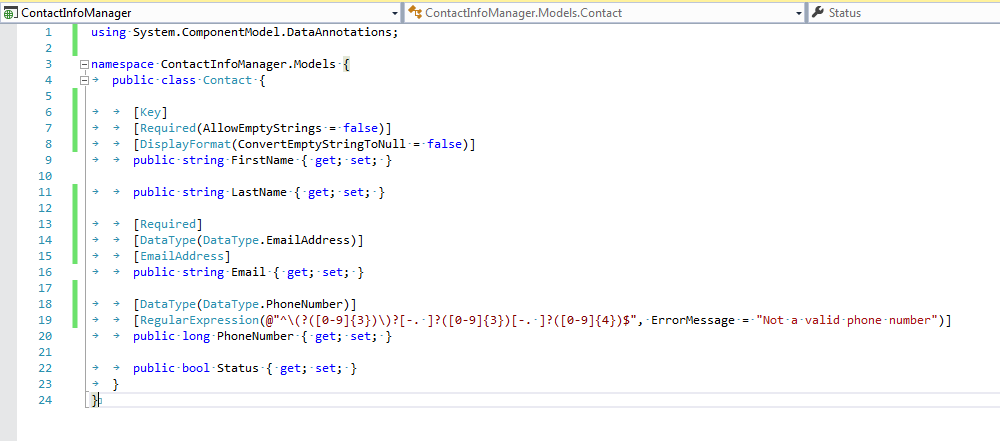
1. The .NET REST api is having 2 option to store data, for demo purpose I am using the global cache that is in memory option but I have also implemented the code for using the entity framework via code first model, the help page for the REST api looks like below and consists of 4 methods – GET, POST, PUT, DELETE



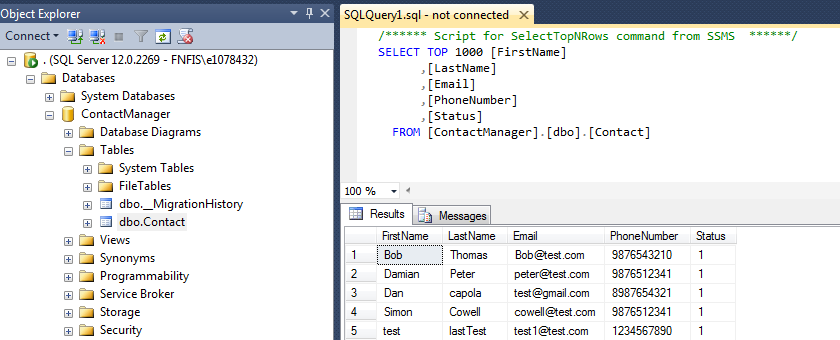
1. The api has been thoroughly unit tested by creating a unit test project in Visual studio



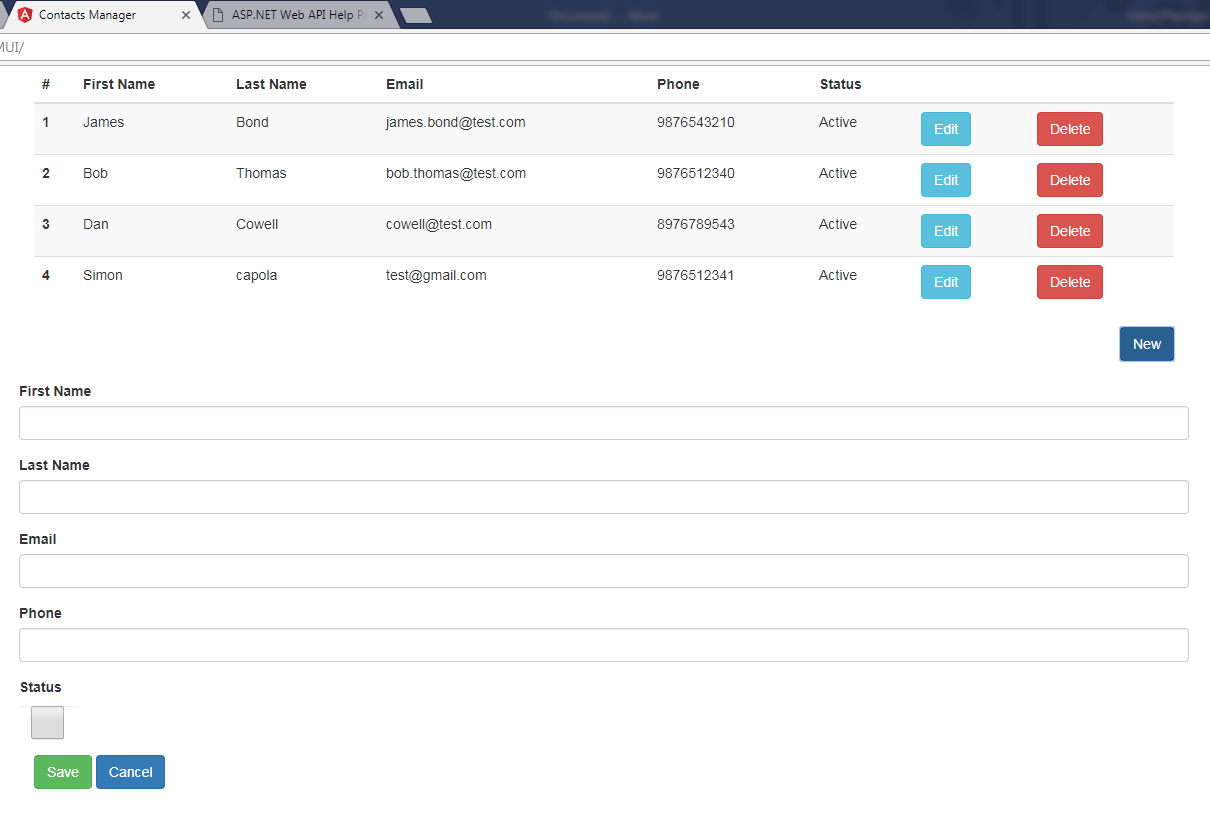
1. The validations have been put in place and are being validated using the Data Annotations, below the ContactModel having the validations in model –



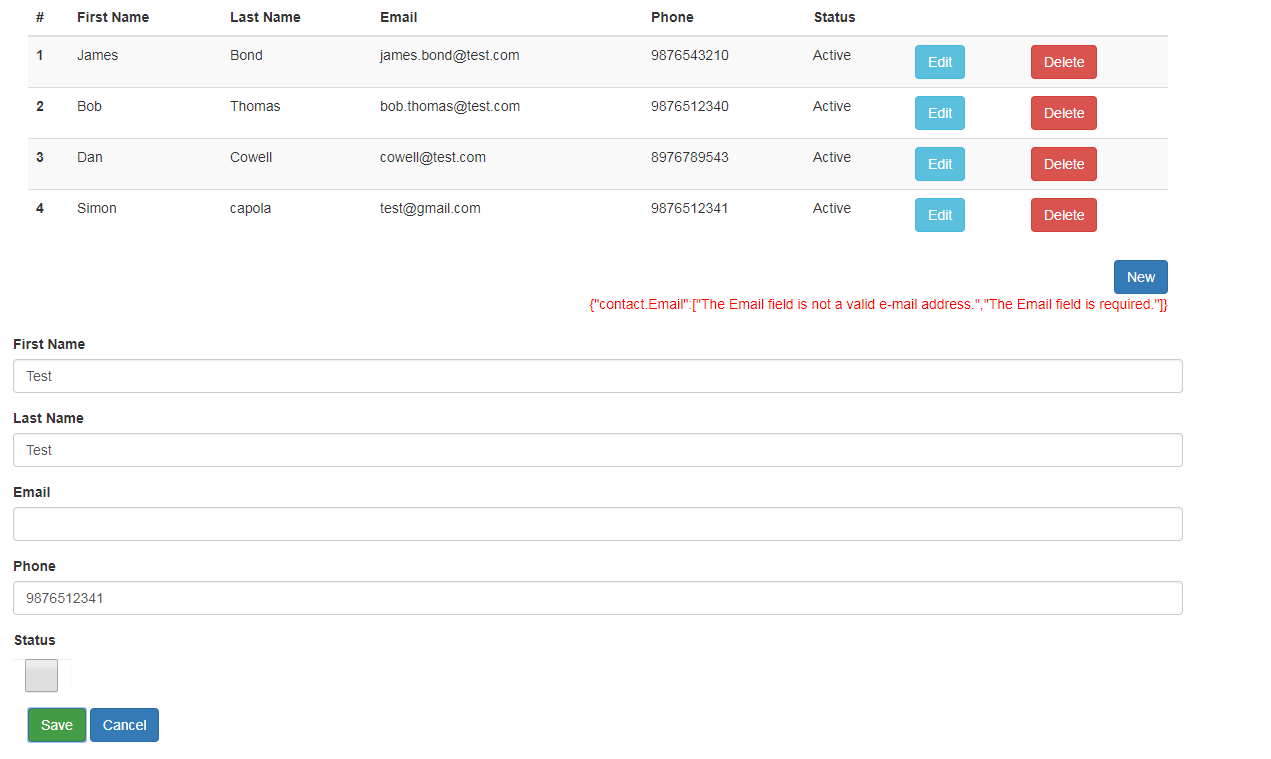
1. The database table Contact and the CRUD database related operations are performed using entity framework, below a snapshot from database table –



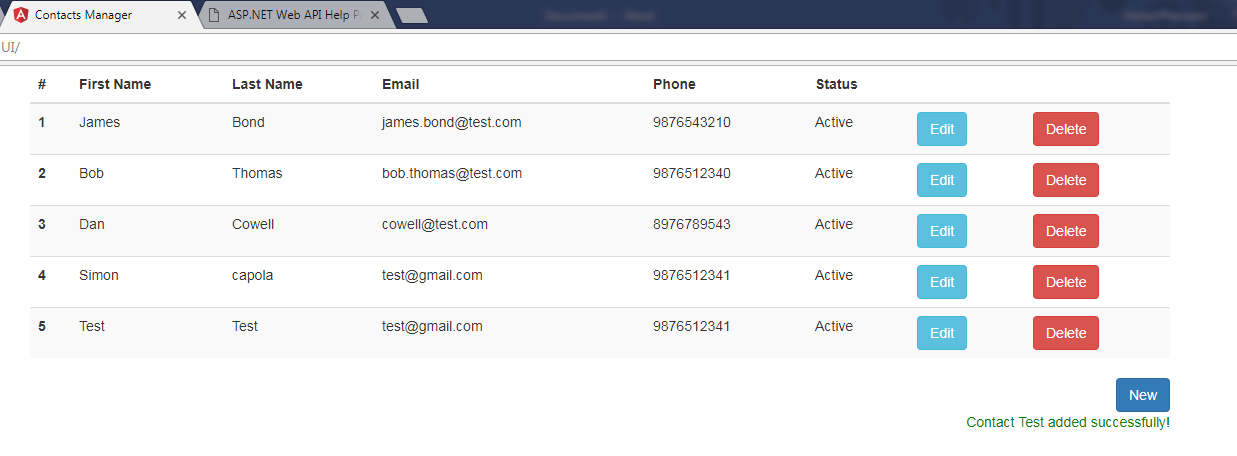
1. On the UI to add new contact click on New –



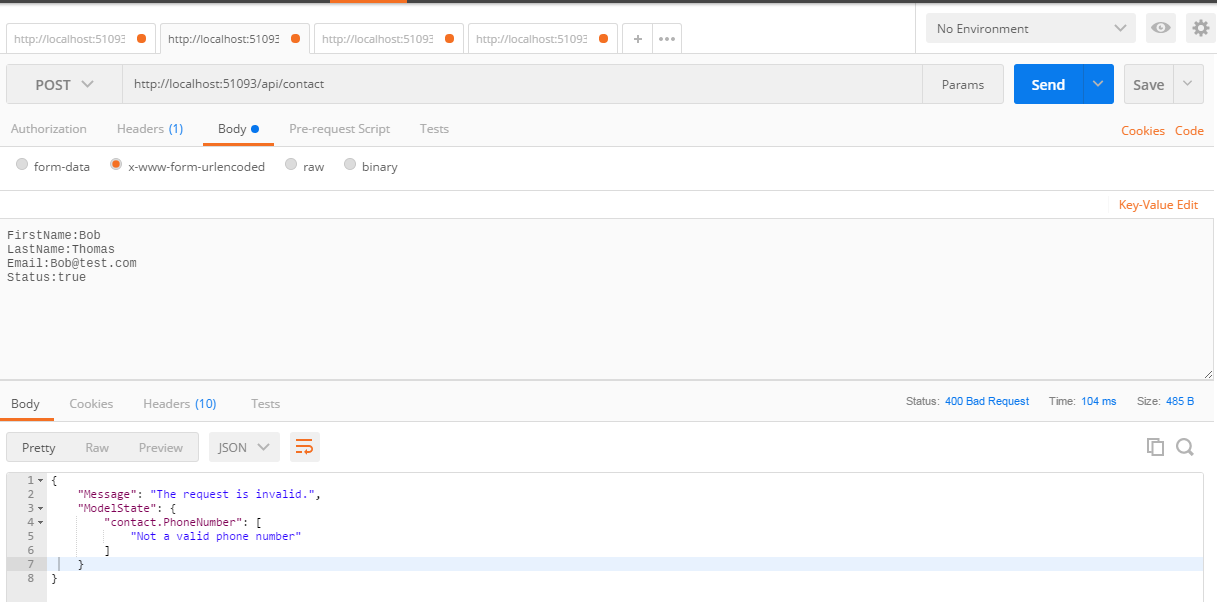
1. Appropriate error messages/success messages are being handled on all operations, example if email is missing –



1. After input of all valid information on save, success message –



1. Testing Model Validation using Post man –



1. Deployment procedure –
   1. The application api for demo purposes is using in memory cache for now so that one can have a look at the overview of the app without database, all CRUD operations are still supported in in-memory cache, but still the app is having an option to connect to database via entity framework and is production ready.
   2. Deploy the UI package to the IIS with name as CMUI
   3. Deploy the REST api package to the IIS with name as CM\_RestApi
   4. The appSettings file in the UI\_Package\assets contains the baseURI path which a configuration to point to the URL where the api is hosted and is accessible.

The application should be all set to run with the above procedure without database with all CRUD operations from in-memory cache.

Note: for the database set up using entity framework code first model, we just need to perform 1. Enable-Migrations 2. Add-Migration 3. Update-Database and enable to code in the contactService to use entity framework dbContext, this has also been tested thoroughly.

GIT Hub URL: <https://github.com/vishalpherwani/ContactInfoManager/>