Hospital simulator has three components:

1. Web Service
2. Web Client
3. Data Service

The web service is a RESTful API that communicates with a data service through Windows Communication Framework (WCF). The web client provides an interface for another application to request a functionality from the web service. Patient data service models the data objects and implements the functionality of the simulator.

WCF REST Web Service:

WCF is a framework for message-based communication between clients and servers done through DataContracts and ServiceContracts. WCF with WebHttp behavior and WebHttpBinding configuration is used to implement the RESTful API web service for the Hospital Simulator. The elements of CRUD can be implemented using HTTP verbs (GET, PUT, POST, DELETE), and serialization is done using JSON format (XML is also supported but not used in this project). REST requires that URIs represent resources which is done by using UriTemplates tag. in WebGet and WebInvoke attributes. These attributes of WCF service allows to define the logic on our RESTful API. Since WCF is heavily configurable, many aspects of it can be managed using the Web.Config, e.g. to specify that the client select the format of the response to a web request, a switch called automaticFormatSelectionEnable is activated. Any WCF service should be hosted to be accessed via client proxies. The Hospital Simulator web service is hosted in IIS Express, as it is conveniently integrated into MS Visual Studio.

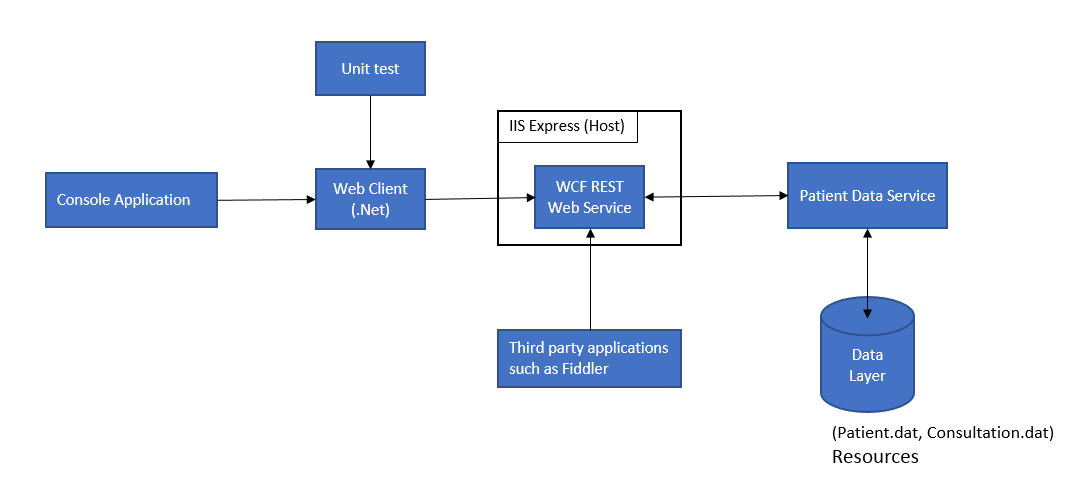
Web Client:

The web client makes all the interactions with the WebService. The client will take care of all the serialization, deserialization, network calls and creating transport objects. Hospital simulator web client uses built-in .Net classes to perform all service calls. Since client is requesting data from server in JSON format, a JSON string data is downloaded from the web service endpoint and desterilized to create the required objects. Because WebClient only uses Http GET verb to make a request to server, to specify another Http verb, HttpWebRequest should be used. To perform that, the client is using a factory method, HttpWebRequest.Create, to request a specific Http verb into the server endpoint.

PatientDataService:

PatientDataService provides the main functionality for patient registration and prepares the data for consumers. It also implements a simplified data layer which is mainly used for unit test purposes. Moreover, it contains the definition for all the data objects. The data objects are all part of the data contract that WCF web service uses to transport data. PatientDataSevice uses a stream reader to read the hospital resources from a JSON file and parse them to their corresponding data objects.

The figure below illustrates the architecture of the Hospital Simulator:



Console application is a simple command line program that tests the main functionality of the web service. It is not designed as the main interface for the web service. But it can be used to list the patients and consultations.

Unit test perform several tests for different aspects of the service. It is focused on the consultation aspect of the simulator since that is the main workflow.

Third party applications such as Fiddler or Postman can be used to directly interact with the web service and inspect the response.

Instruction of how to build:

The Hospital Simulator has been developed using MS Visual Studio 2013. It uses one nuget package from nuget.org (Newtonsoft.Json.11.0.1) and is configured to download that package automatically during build. All other libraries are internal to .Net 4.5 and should be available during build. The application has been tested for debug or release builds and the best way to test it, is to use the unit test.

Please note that as the main source of training, I have used PluralSight course “10 Ways to Build a Web Service in .Net”.