**Willow Tree Test Project**

1. **Project Requirement:**

Create an API that consumes the Willow Tree Employee data from the below URL and makes it possible to implement a full-featured game on top of it. The client should be very simple, and all of the logic for the game (for example, is this guess right?) should be implemented in the server. Providing a client implementation is optional.

Willow Tree Employee Data Url: <https://www.willowtreeapps.com/api/v1.0/profiles>

1. **System Design**

IDE: **Visual Studio 2017**

Technologies: .Net Framework 4.6, Web API 2.0, Html, JQuery, Bootstrap etc.

1. Create a Willow Tree Backend Library project that include Business Control, business Model and Common Utilities etc.

The EmployeeBControl business control will get employee data from Willow Tree Api async and uses the MemeoryCache to catch the data for 10 minutes.

Also create come common utilities classes like Configuration Utility, LogToFile Class, and Custom Exception Filter etc.

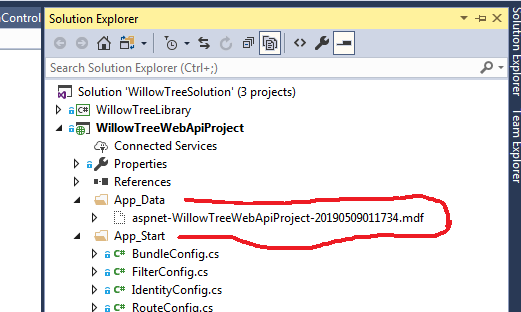
1. Create a Willow Tree Web API project that will consume the Backend Library and generate the API methods that can be consumed by front end project.
   1. For the testing propose, do not need create the separate Front End project, it will include the Front End pages inside the Web API project. The front end pages will use Jquery Ajax to consume Web API.
   2. Make Web API have the complete cross origin features with below setting:

*EnableCorsAttribute cors = new EnableCorsAttribute("\*", "\*", "\*");*

*config.EnableCors(cors);*

So that the separate front end project can also consume the Web API project in the future.

* 1. The Web APIshall the build in Individual Authentication to security API Project. The Front end pages include Register, Login page to create account and get Security access token. The Security database is local database and stays inside in below folder like:



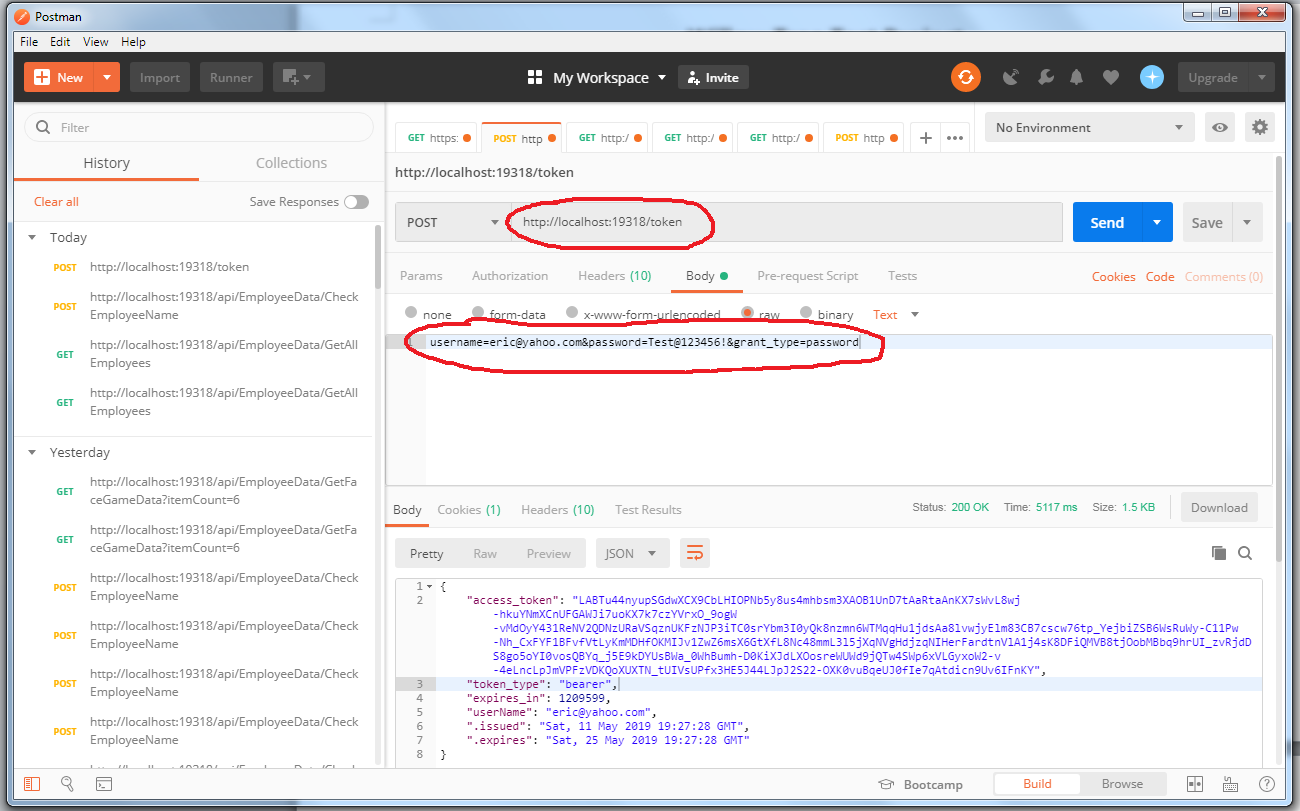
We can move the security db to sql server for real project.

* 1. The front end Game page needs to have Access Token to play the game.
  2. The Web API includes the LogToFile feature, also includes the custom WillowTreeExceptionFilter class to catch all unhandled exceptions for API Control.

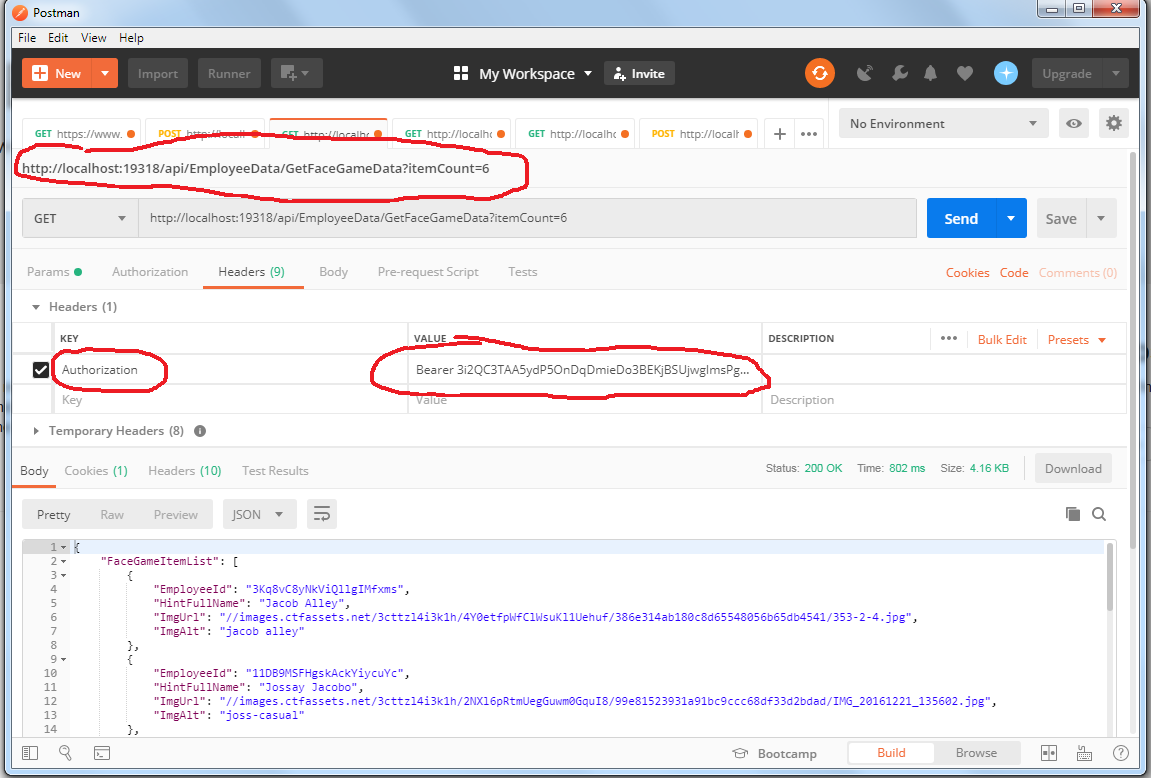
1. Create a Test Project that can test all controls and back end business control.

The test project has the features to test async methods.

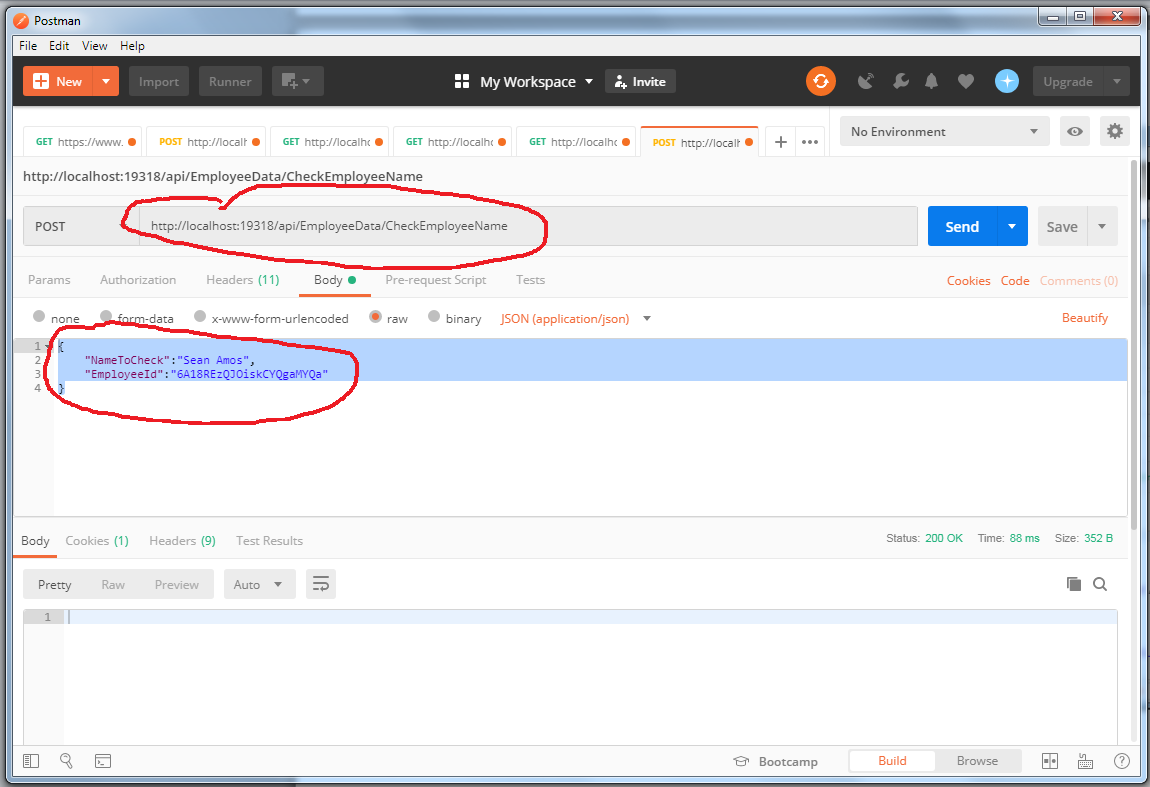
1. **Postman Testing:**
   1. Please run application and click the Register menu to register a new user.
   2. Test getting access token as below:



* 1. Test getting Game Data from Web API (Make sure providing the Bearer access token):



* 1. Test Check Name API:



* 1. Also can test GetAllEmployees and GetEmployee API, please check the source code for details.
  2. Also developer can test the API Methods from the Testing Project.

Please comment out the [Authorize] when test in Test Project.

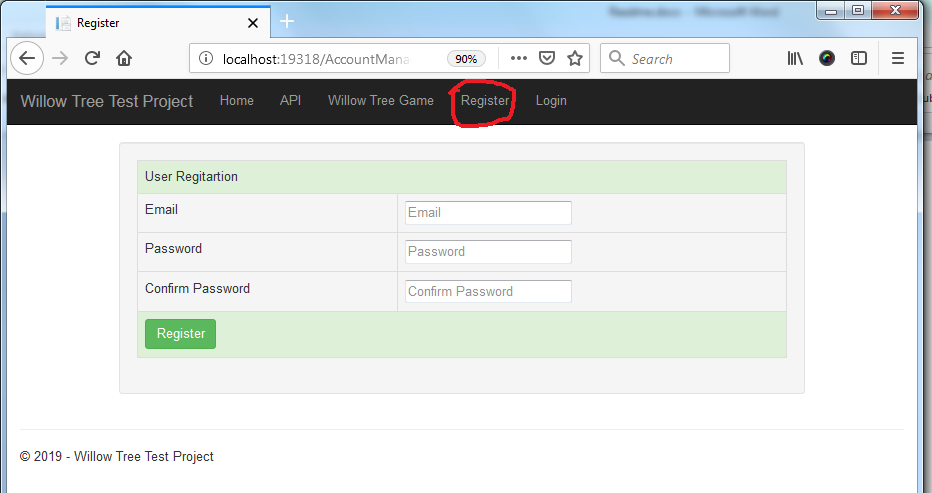
1. **Face Game Test:**
2. Please make sure register your account first. Or you can test by using below testing account if you build the solution from the zip file I sent to you since it already have my local database:

User Name: [eric@yahoo.com](mailto:eric@yahoo.com)

Password: Test@123456!

But if you get source code from Github, then please run application and register first since it has not local database, you must run register to generate local database at first time

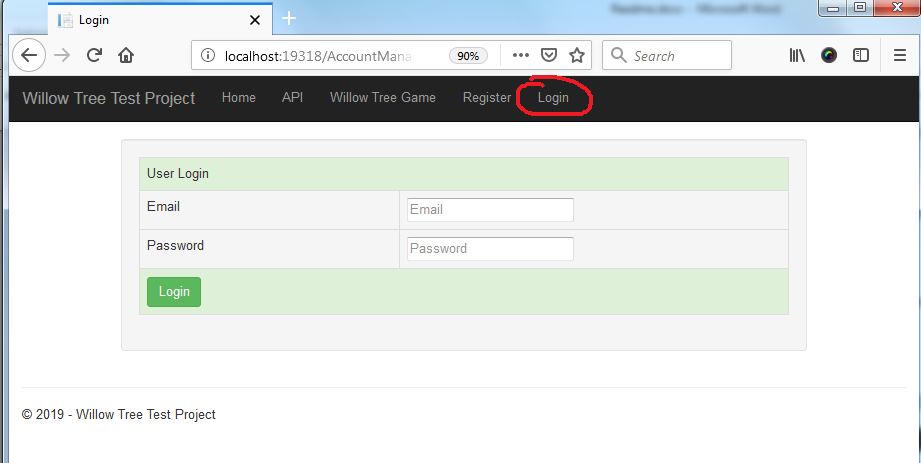
Please click Register menu to run register:



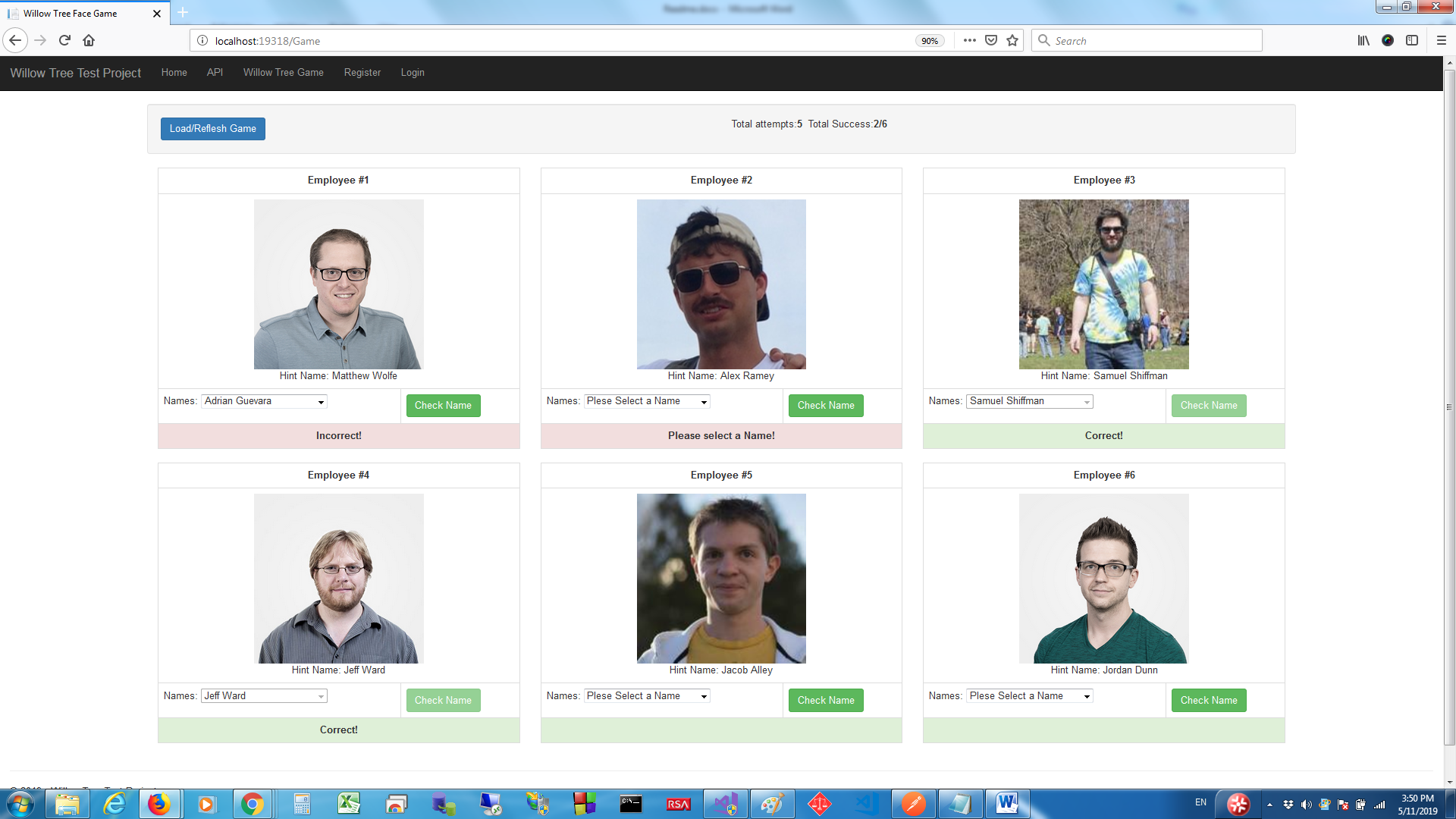
Notes:

After registration, please click Login menu to log in.

Login:



1. After Login, it shall automatically show Game page, please Click the Load/Refresh Game button, you can see below face game:



Notes:

For easily testing, I show the Hint name for each employee.

Player can select name from dropdown list and click the Check Name button to call Web API (*http://localhost:19318/api/EmployeeData/CheckEmployeeName*) to check the selected Name. If the API return correct (200 OK), then it will show “Correct” message and the Check Name button and Dropdown list will be disable.

The game will also show how many attempts and how many finished etc.