## Credentials

|  |  |  |  |
| --- | --- | --- | --- |
| **Email Address** | **Username/UserId** | **Password (Encrypt)** | **Purpose** |
| dribbly.dev@google.com |  | dribbly | For development purposes   * Google Maps Api |
| dribbly.dev@google.com | devdribbly | smarterasp | Test Web Client (<https://member3-1.smarterasp.net/account/loginform?go=1>) |
|  |  |  |  |

## Keys

|  |  |
| --- | --- |
| Google Map API Key | AIzaSyCQwPkj7HcSjORBr6z8ZGf56e4uXNPHUuY |

## Terms and Definitions

|  |  |  |
| --- | --- | --- |
| Term | Definition | References |
| OWIN | An interface that breaks the tight coupling between web servers (such as IIS) and .Net web applications. It allows .Net web application to be hosted on any OWIN-compliant server (not just IIS) |  |
| Asp.Net Identity | Authentication system for ASP.NET applications. It implements authentication not just using Username and Password, but also Facebook and Google authentication and some more advance features | <https://docs.microsoft.com/en-us/aspnet/identity/overview/getting-started/introduction-to-aspnet-identity> |
| OAuth | An authorization pattern which uses tokens instead of Username/Password combination | <https://developer.okta.com/blog/2017/06/21/what-the-heck-is-oauth> |

## Source Code

### Projects

#### Dribbly.Authentication

##### Dependencies:

* Entity Framework 6.4.0

## Creating the project

### Tools used:

* Visual Studio 2017

### Steps:

1. Open Visual Studio
2. Under **File** menu, select **New** > **Project**
3. Copy the options below, then hit OK.



1. On the next screen select the options below, then hit OK.

## Requirements for Phase 1

* Landing page
* Main Navigator
* Authentication
  + Photo upload
* Courts
  + Registration
  + Photo uploading
  + Details Viewing
  + Updating of details
  + Searching
  + Scheduling
* Player profiles
  + Creation
  + Search
* Team Management
* Games management

## App Structure

## Types of Pages

### Search Page

Page used for searching players, courts, games, etc

### Browser Page

Page used for viewing player details, court details, game details

## Rules

### DateTime Objects

* Always expect dates from Web API in **UTC** timezone and in **ISO** format (ie. 2020-03-01T17:59:23Z). If value from WEB API does not have ‘Z’ suffix, add right upon receiving it.

## Conventions

### Dribbly

* Prefix components, directives, filters with ‘drbbly’
* Add a newline character at the end of every files if allowed

### Git

Branch name prefix

|  |  |
| --- | --- |
| **Prefix** | **Purpose** |
| feat | For new components/functionalities |
| improv | For improvement of existing components/functionalities |
| fix | For fixing error in existing components/functionalities |
| setup | For tasks that are not directly related to the app (e.g. compiler setup, publish profile changes, git setup updates) |
| clean | For removing unused code/files |
| docu | For adding/updating comments |
| design | For prototype design changes (usually photoshop files) |
| refactor | For changes in code that do not affect any functionalities (e.g. renaming variables, making code more organized) |

## Guides

### Adding new NPM dependencies

* Install npm module
* Update the **noteLibs** array in gulpfile.js (**Note:** reference required files only)
* Add reference the files in \_SiteLayout.cshtml
* Inject the module into siteModule

### Adding new component files

* Add reference the files in \_SiteLayout.cshtml

## Pages

### Courts List

Features:

* Viewing of list of courts
* Filtering list of courts
* Registering a court

## Modules

### SiteModule

* The “foundation” module
* Should only contain components that will be used both by **authModule** and the **mainModules**
* All 3rd party modules expect those that are only for a specific mainModule should be injected here

### AuthModule

* Should only contain services and components used for authenticating user

### AppModule

* Should only contain components and services that are shared among the **mainModules**

### MainModule

* Should only contain components and services for itself
* **Dribbly** is currently the only mainModule. Dribbly is focused in Basketball. We may add other mainModule for other sports and/or a **store** later

## Modules Hierarchy

Site Module

Auth Module

App Module

Store Module

Main Module

## Publishing

### Local IIS Server

#### Publishing

Pre-requisites:

* Make sure .NET Core Hosting Bundle is installed on the machine (This is required for .NET Core applications to run on local IIS Server)
* Open IIS Manager



* Add Website



* Click OK to create the website
* Run Visual Studio as admin, then open the solution
* In Visual Studio, right-click on the project then select Publish
* Click on New Profile…
* Select “IIS, FTP, etc”
* click on “Create Profile”



* Click Next
* For now, set Configuration to Debug, until the website is configured for release (minify files, etc.)



* Click Save.

Helpful article: <https://docs.microsoft.com/en-us/aspnet/core/host-and-deploy/iis/index?tabs=aspnetcore2x&view=aspnetcore-2.2>

#### Accessing local IIS Instance from another machine via IP address

1. Add Website a binding

* Open IIS Manager, select the website, and then click on “Bindings”



* Click “Add”
* Set type to “http”, select the IP from the dropdown, then enter the port. Leave Host Name empty. (Note: IPs will only show in the dropdown if the network is already connected to the network. Otherwise, type in the IP address. We can also enter \* to allow it to be accessed whatever the current IP address) Click OK



1. Add Firewall rule

* Open Firewall advanced settings window
* Select “Inbound Rules” then click on “New Rule…”



* Select “Port”, then click Next



* Select TCP, then enter the same port number used in the previous step. Click Next



* Select “Allow Connection”. Click Next



* Check all options. Click next



* Enter a name for the rule, then click Finish



* Any device on the same network should now be able to access the website using the IP address and port (example: http//:192.168.43.179:9010)

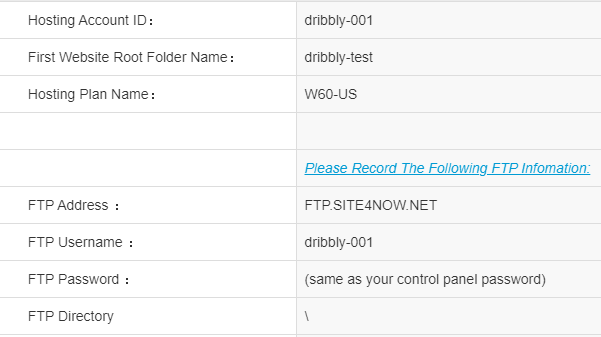
### Test Server (myasp.net)

|  |  |
| --- | --- |
| Email address used | [rjpablo16@yahoo.com](mailto:rjpablo16@yahoo.com) |
| Credentials | Rjpablo112/encrypt(myasp) |
| Site name | <http://rjpablo112-001-site1.dtempurl.com/> |
| FTP Addresses/path | [FTP.SITE4NOW.NET/drbbly-web](ftp://FTP.SITE4NOW.NET/drbbly-web) |
| FTP User Credential | rjpablo112-001/encrypt(myasp) |



### MyWindowsHosting

|  |  |
| --- | --- |
| Username | dribbly |
| Email | [dribbly.test@gmail.com](mailto:dribbly.test@gmail.com) |
| Password | enypt(password) |
| File Management page | <https://member3.mywindowshosting.com/cp/filemanager?pop=1&d=h%3A%5Croot%5Chome%5Cdribbly%2D001%5Cwww%5Cdribbly%2Dtest> |



## Random Notes

Max width for mobile design is **575px** (bootstrap’s breakpoint for sm (small) screens)

## App Flow

Under Courts show limited number of Featured Courts, New Courts, etc. A “more…” link at the bottom of each category would load more items for that category or show all the items in a modal with pagination

## References

* Password Reset <https://code-maze.com/password-reset-aspnet-core-identity/>

## Ideas

* Each player should have the option to vote for one mvp. The player who get the most mvp votes becomes the Game MVP. All MVP votes received by a player will reflect on his profile, in addition to how many times he became the Game MVP. Players who selects an MVP will receive points as reward
* Upon publish, register public basketball courts as initial Featured content