Table of Contents

[Installation Guide 3](#_Toc493812710)

[Build the back-end solution 3](#_Toc493812711)

[Build the front-end solution 4](#_Toc493812712)

[Task 1 - Race Dashboard 5](#_Toc493812713)

[Race status 5](#_Toc493812714)

[Total amount of money for each race 5](#_Toc493812715)

[All horses per race with name, bet count and winning prize 5](#_Toc493812716)

[Task 2 - Additional Endpoints 6](#_Toc493812717)

[List of customers 6](#_Toc493812718)

[Total bet count per customer 6](#_Toc493812719)

[Total bet amount per customer 6](#_Toc493812720)

[Total bet amount for all customer 6](#_Toc493812721)

[At risk customers (with bets more than $200) 6](#_Toc493812722)

[Solution Architecture 7](#_Toc493812723)

[Hosts 7](#_Toc493812724)

[TechChallenge.ApiHost 7](#_Toc493812725)

[TechChallenge.Spa 7](#_Toc493812726)

[Business 8](#_Toc493812727)

[Engines 8](#_Toc493812728)

[Strongly-typed configurations 8](#_Toc493812729)

[Services 9](#_Toc493812730)

[Tests 9](#_Toc493812731)

[TechChallenge.Tests.Unit 9](#_Toc493812732)

[Strategies used 10](#_Toc493812733)

[BaseClasses 10](#_Toc493812734)

[IoC/DI using MEF 10](#_Toc493812735)

[Logging– NLog and Application Insights 10](#_Toc493812736)

[Async/Await 10](#_Toc493812737)

[Linq via Fluent Api 10](#_Toc493812738)

[Generics 10](#_Toc493812739)

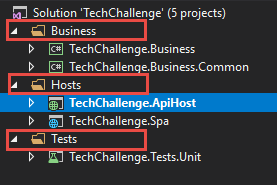
[Mediator 10](#_Toc493812740)

[ActionFilterAttribute – ExplicitDisposeAttribute and IDisposeAware 10](#_Toc493812741)

[NuGets 10](#_Toc493812742)

Solution Architecture Overview

In bird’s eye view, the solution is separated into 3 different areas namely: **Business**, **Hosts** and **Tests.**



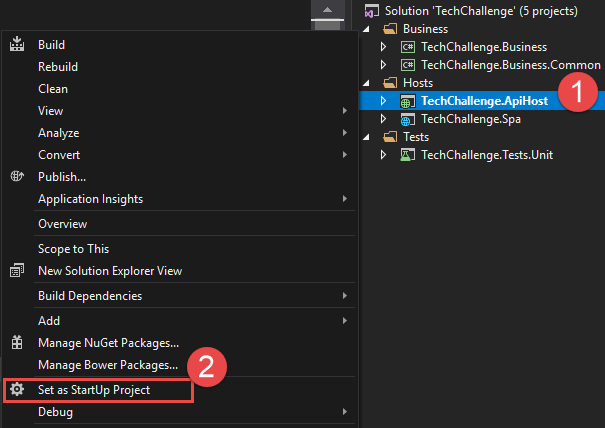
The solution is built using Visual Studio 2017

– with .Net 4.6.2 and .NetCore 1.1 (*for TechChallenge.Spa only*)

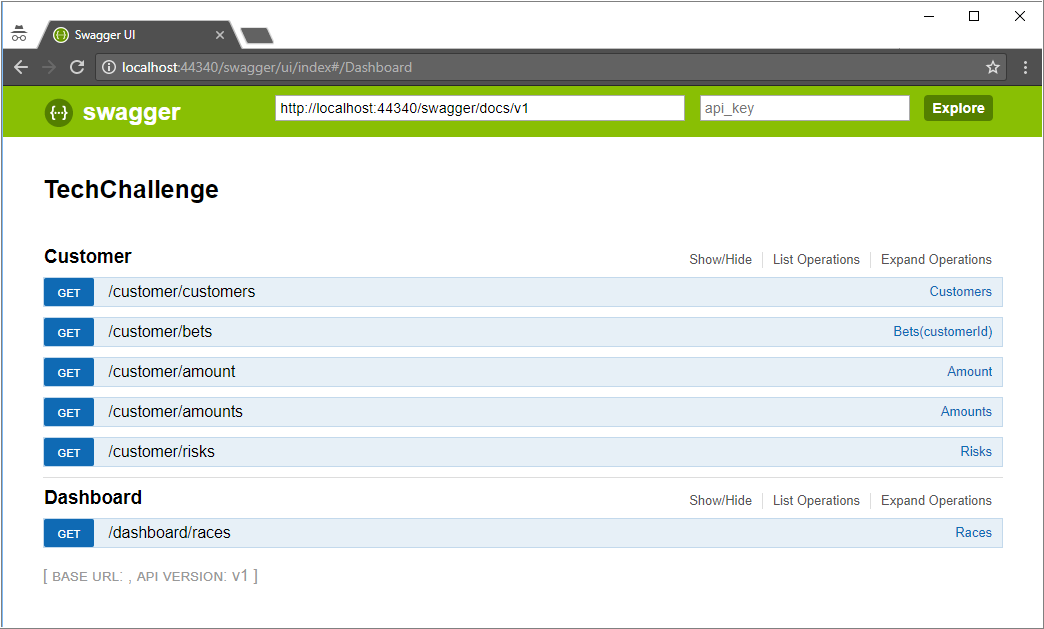
# Installation Guide

## Build the back-end solution

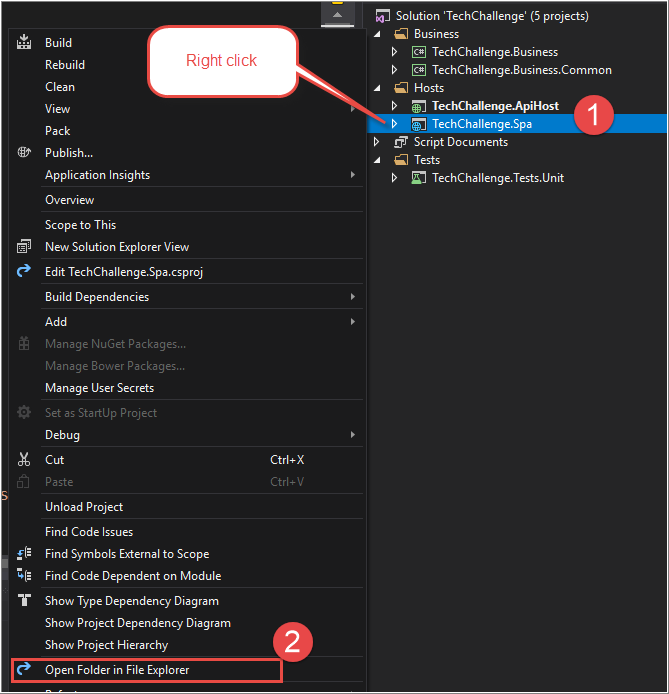
Set start-up Project

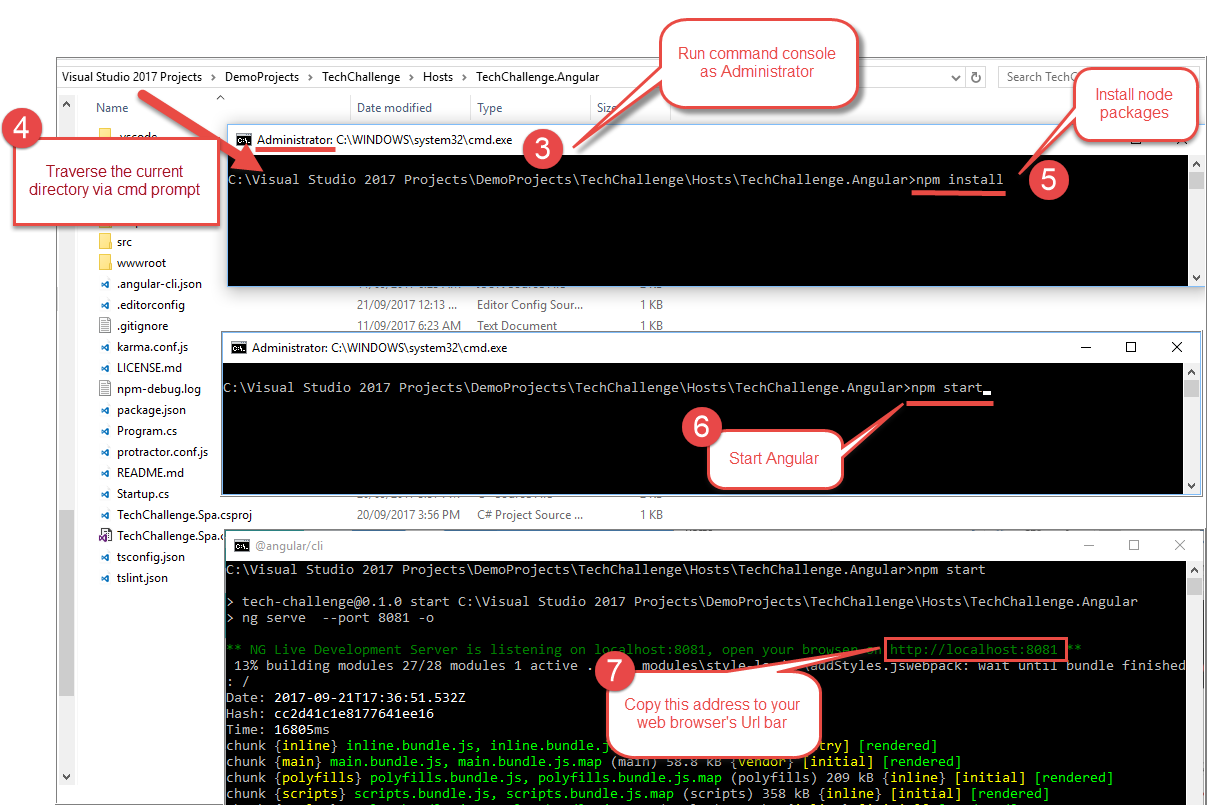


Press F5 to view the WebApi documentation via swagger



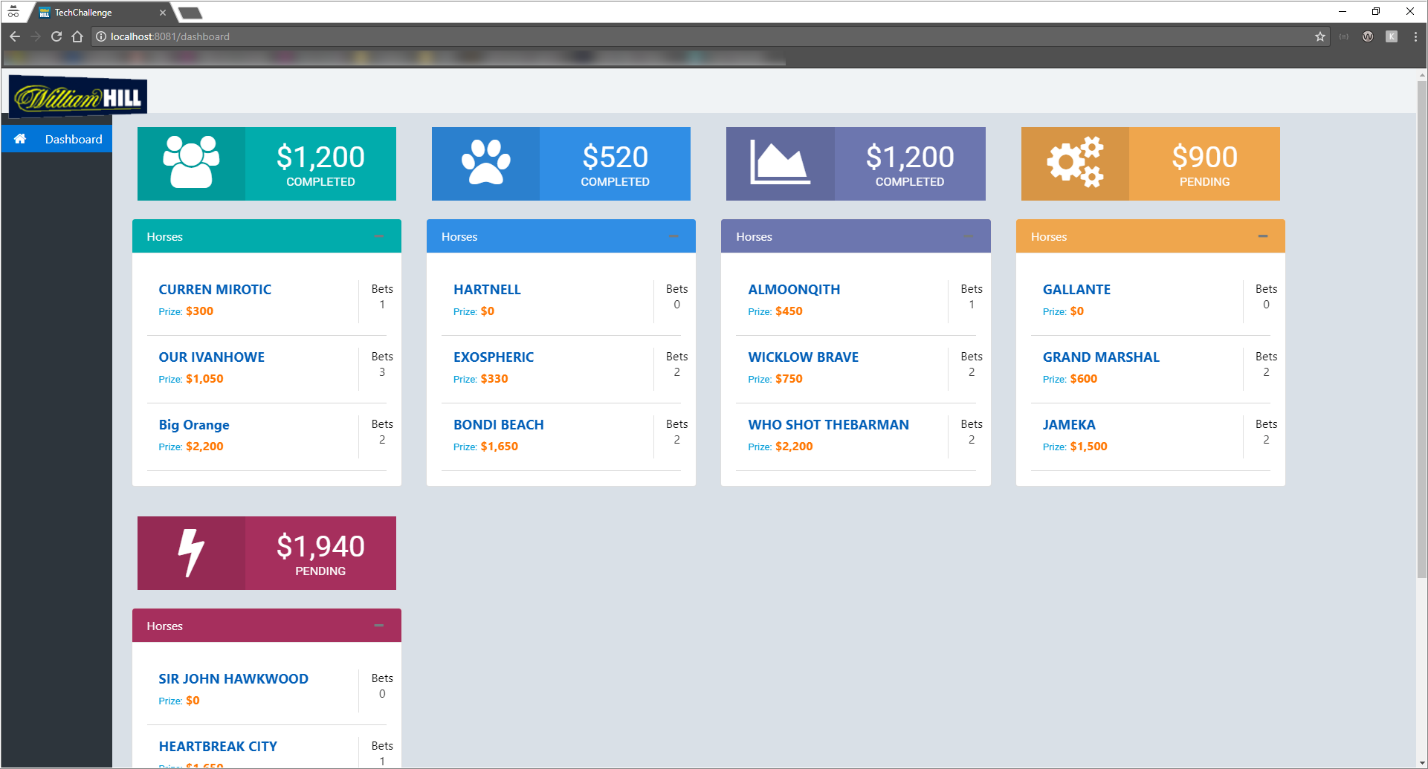
## Build the front-end solution





**Note**: Keep the WebApi solution running while browsing the front-end.

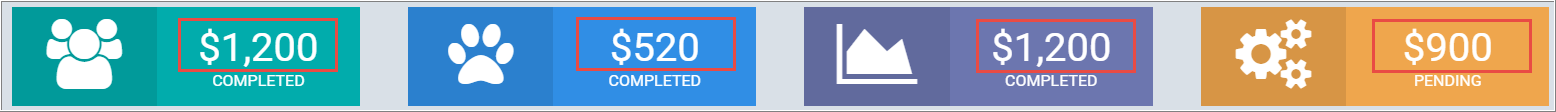
# Task 1 - Race Dashboard



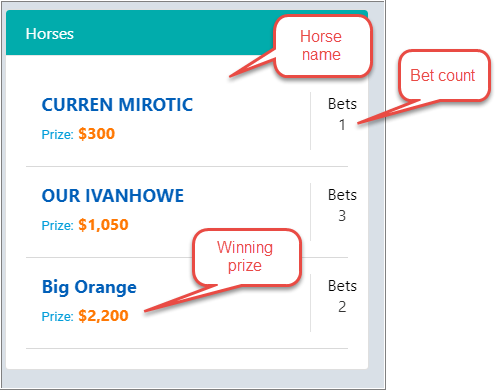
## Race status



## Total amount of money for each race



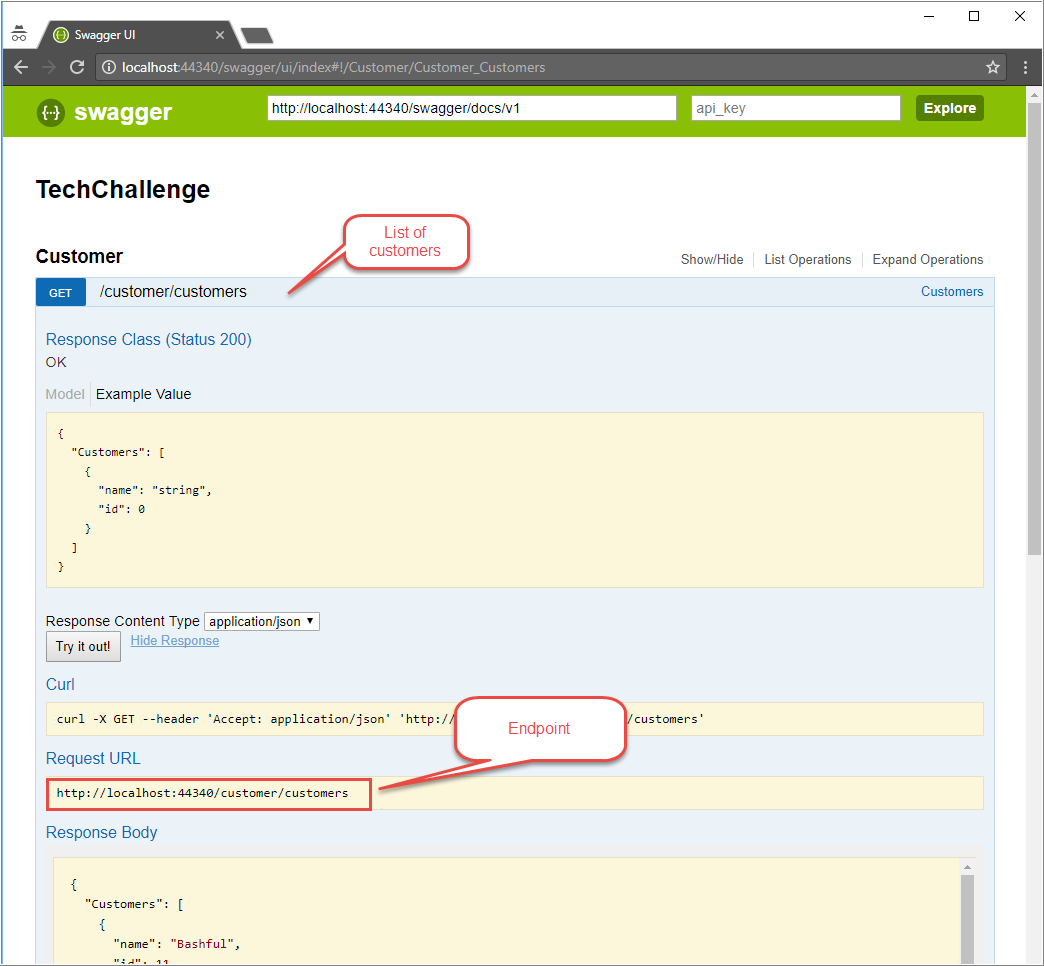
## All horses per race with name, bet count and winning prize



# Task 2 - Additional Endpoints

## List of customers

(http://localhost:44340/customer/customers)



## Total bet count per customer

(http://localhost:44340/customer/bets)



## Total bet amount per customer

(http://localhost:44340/customer/amounts)



## Total bet amount for all customer

(http://localhost:44340/customer/amount)



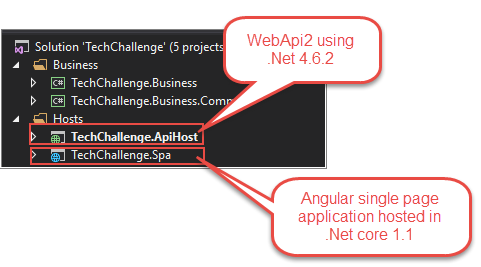
## At risk customers (with bets more than $200)

(http://localhost:44340/customer/risks)



# Solution Architecture

## Hosts



### TechChallenge.ApiHost

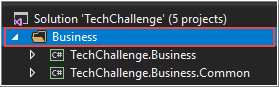
The back-end project using WebApi2.

### TechChallenge.Spa

Front end project using Angular4, ES6, Typescript and WebPack.

## Business

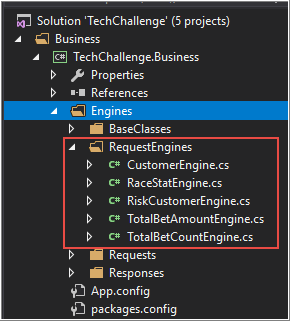
The purpose of this separate project is to capture the ‘business use cases’ into one location for reusability and maintainability. This is where most of the business logic is found.



### 

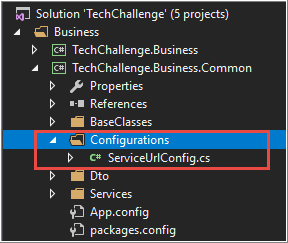
### Engines

Engines are the code representation of specific business use-case. This is where the request/response pattern are implemented, glued together by the Mediator class (see NuGet section). The engine’s “single responsibility” characteristic provides modularity and better testability.



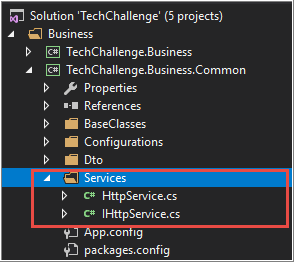
### Strongly-typed configurations

Uses ConfigParser to produce strongly types values. (see NuGet section).



### Services

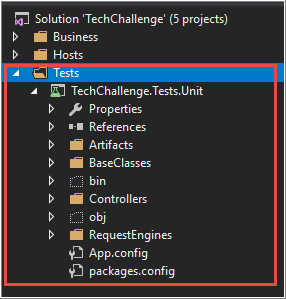
A general-purpose class used to retrieve data from external endpoints.



## Tests

### TechChallenge.Tests.Unit

Contains all the unit tests for the web api.



## Strategies used

### BaseClasses

### IoC/DI using MEF

### Logging– NLog and Application Insights

### Async/Await

### Linq via Fluent Api

### Generics

### Mediator

#### Request/Response pattern

#### Abstract class factory pattern

#### Command pattern

### ActionFilterAttribute – ExplicitDisposeAttribute and IDisposeAware

## NuGets

Below are my NuGet packages installed in this exercise:

