**Rich Internet Applications Project**

**Msc in Web Technologies**

**A Single Page Application for a Swimming Pool Enterprise**

**Created with AngularJS, HTML5 and CSS3**

**Author: Venkat Reddy Gaddam (X11102071)**

**Lecturer: Vikas Sahni**

**Repository:** <https://github.com/indiag/aria>

**Deployed at:** <http://indiag.github.io/aria/app/#/view1>



**Table of Contents**

1. Introduction
2. Motivation

3.1 Project Scope

3.2 Area of Contribution

4. State of Art Review

4.1 Technologies Used

4.2 Angular JS

4.3 Protractor

4.4 Skel JS

4.5 CanvasJS

4.6 Google Calendar API

4.7 Google Map API

4.8 Single Page Applications in Angular JS

4.8.1 Asynchronous Loading of Pages

4.8.2 HTML

4.8.3 Controllers

4.8.4 Routes

4.8.5 Partials

4.9 Critical Review of Strategy Adopted

4.9.1 Possibilities Approached

4.9.2 Approach Adopted

4.9.3 Methods and Strategy

4.9.4 Critical Analysis

5. Interface Design

6. Architecture

6.1 Application Architecture

6.2 Security

6.2.1 Skel and Angular as a Security Risk

6.2.2 Angular Defence against Security Risks

6.3 Data Transfer Strategies

6.4 Evaluation and Testing

7. Conclusion

8. References

**1. Introduction:**

The main Aim of this Project was to create a Rich Internet Application for Swimming Enterprise which would eventually be available Online, Viewed on any Device when connected to the Internet for instance Mobile Phone, Tablet etc.

I created a Single Page Application using primarily Angular JS, HTM5 and CSS3. The code has been deposited on Github (<https://github.com/indiag/aria> ) and deployed on github pages (<http://indiag.github.io/aria/app/#/view1> ). The Code was tested using Protractor (link) an End to End Testing Framework for Angular JS.

Here I have designed the Rich Internet Application in the Recreation Environment. The main purpose of this Website is that it allows all types of Swimmers to use our Website and book their respective Swimming Courses rather than coming to our Pool location by wasting lot of time to book their Swimming Courses. The name of our Website is “**Pool App**”. Here in our Website there are many Swimming Courses offered such as Back stroke, Butterfly Stoke, Side Stroke, Free Style Stroke, Breast Stroke. We also provide other services in our Swimming Pool such as Aerobics, Gymnasium for the Swimmers in our Pool. Here in this Website I mainly concentrated on creating a Single Page Application using Angular JS Framework. The main Goal was to create a Single Page Application (SPA) which had the “Look and Feel” of a Desktop Application which uses only recent Open Source Technologies. We are actually targeting all type of users who want to swim mainly. It also provides the principles of the Responsive Design and Test Driven Development. It is purely a Rich Internet Application with all the Recent Technologies involved in it. The main idea to create this Web application is keeping 2020 year in mind. We have mainly focussed on implementing this Website by using all the recent technologies which would be can be used easily till 2020 year as these technologies used are booming and will be the future in next decade too.

**2. Motivation:**

**2.1 Project Scope:**

This Application was created by using a Single Page Application with Angular JS Framework, HTML5, CSS3 and JavaScript. The Responsive Design was achieved by using the SkelJS framework. The Testing part was done by using Protractor which is one End to End Test Framework for Angular JS and Testing Framework used by Protractor was Jasmine (link). The Version Control was with Github and the Application was deployed using Github Pages.

This Application was created using Model View Controller (MVC) Architecture where the Presentation Layer and all the Business Logic lies on Client.

**2.2 The Area of Contribution:**

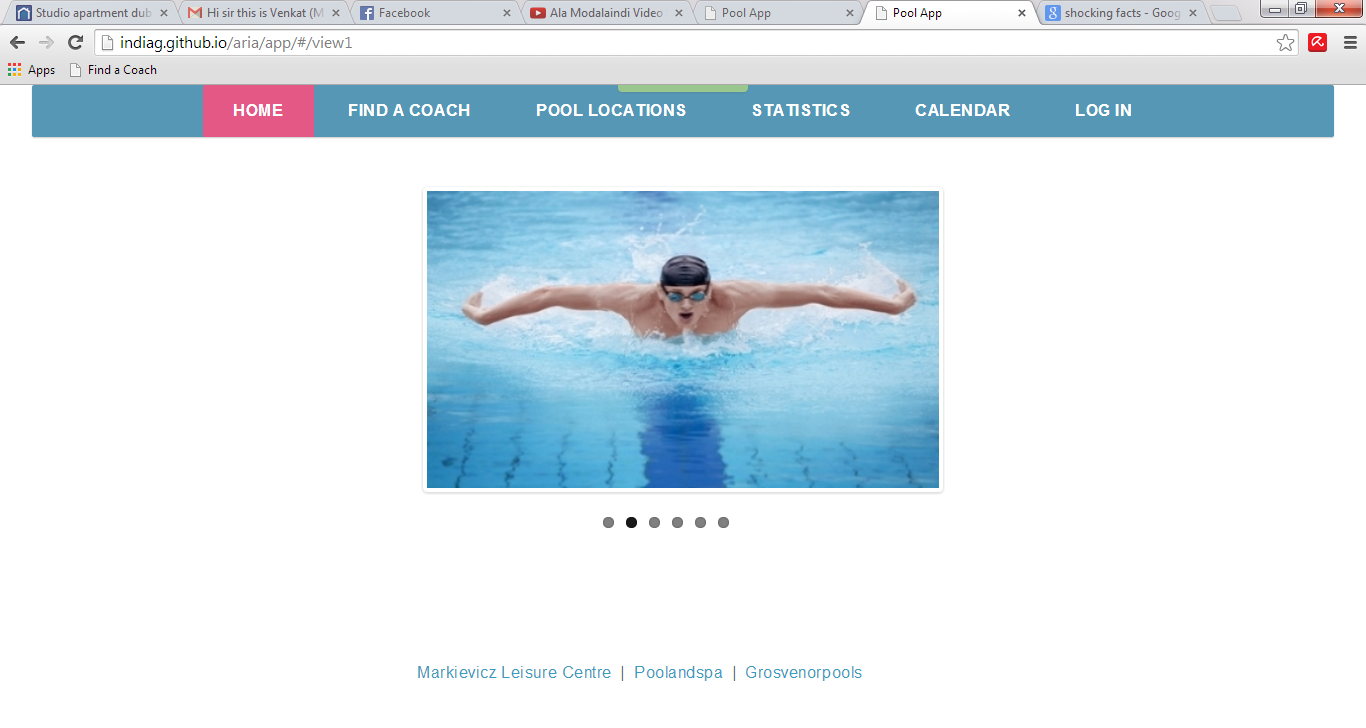
It was actually decided to create a Rich Internet Application for Irish Pool Enterprise where we have incorporated the following Functionalities.They are as follows:

1. A Home Page
2. Dynamic Search Functionality
3. Find Pool location by using Google Maps
4. View the Workout Statistics
5. Calendar
6. Log in

A main feature of this Application is that the Page load times are minimized a lot, as the user would be expected to browse quickly to the functionality and the Information required.

By using Angular JS Framework there is a great benefit like all the Code is incorporated into a Single Page where the Partial HTML files are loaded asynchronously and injected into Single HTML page i.e. Index.html.

Screen shot of Homepage is shown in figure 1.a below:



s

Figure 1 a

**3. State of the Art Review**:

**3.1 Technologies Used:**

The main Technologies used were as follows:

1. Angular JS
2. Skel JS
3. Canvas JS
4. Protractor
5. Google Map API
6. Google Calendar API
7. Angular Seed
8. JQuery
9. Ajax
10. Html5
11. Css3
12. JSon
13. Visual Studio 2012
14. Zurb Foundation
15. JQuery Flex Slider
16. Date Picker

Let us discuss some of them in Detail.

**3.2 Angular JS:**

The Angular JS is an Open Source JavaScript Framework which was created by Google and it was released in May 2011 by Hevery and Green. It actually allows the Dynamic functionality to be added to HTML by using a very powerful Syntax and MVC Architecture. It provides Data Binding and Dependency Injection. By using Angular it helps to express the Behaviour in a Clean Readable form without updating the DOM.

**Data Binding** is actually a method of Changing the View whenever the Model changes. Here the Controllers are written in JavaScript and controls the Behaviour of DOM elements.

The Web Application normally built in Angular JS are usually Testable, Well-structured and maintained for the Front End Applications.

**Advantages of Angular JS:**

1. It actually uses the MVC Architecture and **Separation of Concerns** and it is easy to implement.
2. It actually allows automatic Synchronization of both Models and Views.
3. It’s a lightweight, Open Source framework which allows Dynamic functionality to be incorporated into HTML does with Syntax.
4. It’s easy to Deploy.
5. It’s easy to test the Functionality.
6. It’s widely used and documented well.

**Disadvantages of Angular JS**

1. The Angular Controls are actually incorporated into HTML Mark up and it violates the principle of Unobtrusive Jscript.
2. The Angular Controllers are not Compatible with JQuery.[1] [8]

**3.3** **Protractor:**

The Protractor is actually a very powerful Testing Framework used for Angular JS which allows the End to End Testing. It is built on the Top Web DriverJS and then tests an App by running it in a Browser. Here the Jasmine Framework and Syntax may be used to write the Tests.

**Advantages of using Protractor JS:**

1. The Tests may be written in Jasmine Syntax.

2. Here any piece of Functionality is tested.

3. It actually allows the main Functional Website to be tested.

**Disadvantages of using Protractor JS:**

1. This could be Slow. It might take several minutes for an App to be Tested.[7]

**3.4 Skel.Js**

The Skel JS is actually a Lightweight Frontend Framework for building Responsive Sites and Apps. All of the Code is actually contained in a Single JavaScript File which for the present Release is actually around 20 Kb.

There are some 4 Essential Components for SKEL.JS:

**1. The Responsive Handler**:

**CSS Media Queries** may be used here to assign a different Style sheet depending on Browser Window Style.

**2. CSS Shortcuts:**

It uses CSS Shortcuts for common CSS tasks.

**3. Plugin System:**

SkelJS actually provides a Platform for installing Plugins without actually compromising on Lightweight Status

**Advantages of Using Skel.js**

1. It’s Light Weight in nature.
2. It’s an Open Source
3. It allows the Responsive Design functionality which could be easily into an Application.[4]

**3.5.Canvas.js:**

It’s an HTML/JS Library which is built on the top of the HTML5 Canvas Element. It actually allows Dynamic Charts and Graphs to be constructed. Its main features include Preloaded Themes, High Performance and Cross platform Capability.

**Advantages of using Canvas.js :**

1. It’s free to Non Profit Organizations but paid to Commercial Apps.
2. It actually renders 105 times Data points in less than hundred mts.

**3.6 Google Map API:**

It offers a wide range of API’S that actually allows Google Maps to be embedded into an Application. This actually includes a Google Map (JS API)

**Key Features of the Google Map API are:**

1. It’s a Free API key from Google
2. The Google Map overlays has option to add a Marker in the Google Map.
3. Google Map Controls by default sets Zoom, Map type and Street View options.

Actually there are 4 types of Google Maps available. They are:

1. Road Map
2. Satellite
3. Hybrid
4. Terrain

**Advantages of Google Map API**:

1. API’s mostly are Free or very cheap.
2. By leveraging the Resources of the web it could have saved a lot of time in terms of Development Time.

**Disadvantages :**

1. The Reliability of the System actually depends upon the provider. Since it is Free, the provider does not have the Responsibility to keep the App Operational.[2]

**3.7** **Google Calender API:**

This Google Calendar API actually allows to incorporate into a Web app where the user can create some new events, edit and delete the Existing events.

**Features of Google Calendar API:**

1. The Events can have the information’s like Title of Event, Start and End Times and Attendees.
2. A feature which actually shows that you are Free or Busy in terms of Calendars.

**3.8 Single Page Applications in Angular.js:**

Here in this Web application we mainly concentrated on the Single Page Application feature .In this Single Page Application **(SPA**) the code here is retrieved in a Single Page load and all the Actions are carried out on a Single Page. The idea of moving to separate page is created by the Dynamic communication with a Server and mainly by using JavaScript.

The main Benefit of the Application is that it actually decreases the number of HTTP Requests and the User may browse the site without any Delays associated with Page Load times. Actually the Angular JS makes it easy to create a SPA. Here Skeleton SPA was created using Angular Seed Framework.

In this Framework **SPA** is created using MVC Architecture. The Dependencies are firstly loaded asynchronously and the Single Page is Index.html. The Partial Html files are ‘injected’ into Index.html in Response to the User Interaction where each Partial is under the control of a Separate Controller. [6]

**3.8.1Asynchronous Loading of Dependencies:**

Here in this Web application the Dependencies and Partials are loaded asynchronously. The file index.html could be updated at any time.

**3.8.2 Html:**

The Relevant HTML is shown in the following Code Snippet (Index.html). The Index.html is ‘hooked up’ to a Module and defined in app/js/controller.js by using the Angular ng-app directive and the Contents of each Partial are ‘Infected’ into a Diva by using the ng-view directive.

Coding provided here:

<!doctype html>

<html lang="en" ng-app="myApp">

<head>

<meta http-equiv="content-type" content="text/html" charset="utf-8">

<title>Pool App</title>

<link href="http://fonts.googleapis.com/css?family=Source+Sans+Pro:300,300italic,600|Source+Code+Pro"

rel="stylesheet"

/>

<script src="js/vendor/jquery-1.10.2.min.js"></script>

<script type="text/javascript" src="js/canvasjs.min.js"></script>

<script type="text/javascript" src="js/main.js"></script>

<script type="text/javascript" src="js/vendor/skel.min.js"></script>

<link rel="stylesheet" href="css/app.css" />

<link rel="stylesheet" href="css/main.css"/>

<link rel="stylesheet" type="text/css" href="css/angular.css" />

<link rel="stylesheet" type="text/css" href="css/angular2.css" />

<link rel="stylesheet" href="css/foundation.css" />

<script type="text/javascript" src="http://maps.googleapis.com/maps/api/js?sensor=false"></script>

</head>

<body>

<div class="container">

<div id="main" ng-app>

<!-- Header -->

<div class="row flush header">

<div id="irish pool logo.jpg"></div>

</div>

<nav class="**{{**active**}}**">

<a href="#/app/view1" ng-click="active='home'" class="home">Home</a>

<a href="#/view2" ng-click="active='partialTwo'" class="partialTwo">Find a Coach</a>

<a href="#/view3" ng-click="active='partialThree'" class="partialThree">Pool Locations</a>

<a href="#/view4" ng-click="active='partialFour'" class="partialFour">Workout Statistics</a>

<a href="#/view5" ng-click="active='partialFive'" class="partialFive">Calendar</a>

<a href="#/view6" ng-click="active='partialSix'" class="partialSix">Log in</a>

</nav>

<!-- Hero -->

<section>

<div ng-view></div>

</section>

<div id="loginid" >You have Logged in successfully</div>

<!-- Footer -->

<div class="row">

<div class="-4u 9u">

<div id="footer">

<ul>

<li class ="ulfooter"><a href="http://www.dublincity.ie/RecreationandCulture/SportsFacilities/LeisureCentres/Locations/markievicz/pages/markieviczleisurecentre.aspx">Markievicz Leisure Centre</a>

&nbsp;| &nbsp; </li>

<li class ="ulfooter"><a href="http://poolandspa.ie/swimming\_pool\_suppliers/index.htm"> Poolandspa</a> &nbsp;| &nbsp;</li>

<li class ="ulfooter"><a href="http://www.grosvenorpools.com/index.html"</a>Grosvenorpools</a></li>

</ul>

</div>

</div>

<script src="js/main.js"></script>

<script src="js/vendor/skel.min.js"></script>

<script src="js/angular.js"></script>

<script src="js/angular-route.js"></script>

<script src="js/app.js"></script>

<script src="js/services.js"></script>

<script src="js/controllers.js"></script>

<script src="js/filters.js"></script>

<script src="js/directives.js"></script>

<script src="js/vendor/jquery.js"></script>

<script src="js/vendor/ui-utils.js"></script>

<script src="js/googlemap.js"></script>

<script src="js/location.js"></script>

<script src="js/angular-sanitize.min.js"></script>

</div>

</body>

</html>

**3.8.3** **Controller:**

Here we now have each partial which is under the Control of a Separate Controller. These are actually defined in /app.js/controller.js. The Code Snippet is shown below.

'use strict';

/\* Controllers \*/

angular.module('myApp.controllers', ['ngSanitize'])

.controller('MyCtrl1', function ($scope) {

//Partial One code Goes Here

//alert('hello from partial One')

})

.controller('MyCtrl2', function ($scope, $http) {

$http.get('./Coaches.json').success(function (data) {

$scope.coachList = data;

});

$scope.orderProp = 'name';

$scope.HTMLalpha = "Alphabetical";

$scope.HTMLlowestPrice = "Lowest Price";

$scope.HTMLhighestPrice = "Highest Price";

$scope.HTMLCategory = "Category";

$scope.HTMLemail = "Email";

$scope.HTMLcreated\_at = "Created At";

$scope.HTMLupdatedAt = "Updated At";

$scope.HTMLid = "Id";

//$scope.orderPropAlt = 'email';

$scope.myFirstName = function (string) {

return string.split(' ')[0]

}

})

Here the Code in the Controller 2(ng Ctl2) actually illustrates the syntax of Angular.js. Firstly, the $http Angular JS Service retrieves the Data from a JSON file and then it assigns it to the variable Data.Secondly, the function myFirstName is defined where $ Scope allows this function to be called from within an HTML file by using Data Binding.

**3.8.4** **Routes**:

Here the Route for each Partial is set in the app/js/app.js and it is shown in the Code below and its very easy to implement it. The Display of the Routes is shown below in Figure 3.

'use strict';

// Declare app level module which depends on filters, and services

angular.module('myApp', [

'ngRoute',

'myApp.filters',

'myApp.services',

'myApp.directives',

'myApp.controllers'

]).

config(['$routeProvider', function ($routeProvider) {

$routeProvider.when('/view1', { templateUrl: 'partials/partial1.html', controller: 'MyCtrl1' });

$routeProvider.when('/view2', { templateUrl: 'partials/partial2.html', controller: 'MyCtrl2' });

$routeProvider.when('/view3', { templateUrl: 'partials/partial3.html', controller: 'MyCtrl3' });

$routeProvider.when('/view4', { templateUrl: 'partials/partial4.html', controller: 'MyCtrl4' });

$routeProvider.when('/view5', { templateUrl: 'partials/partial5.html', controller: 'MyCtrl5' });

$routeProvider.when('/view6', { templateUrl: 'partials/partial6.html', controller: 'MyCtrl6' });

$routeProvider.otherwise({ redirectTo: '/view1' });

}]);

**Figure 3**

**3.8.5 Partials:**

The Partial actually illustrates 3 main points. They are

**1.Repeater Functionality:**

The ng-repeat Directive is actually used to loop once through the JSON file and the Data Binding may be used to abstract the Relevant Data into a HTML Unordered List.

**2. Dynamic Searching**:

The Dynamic Search functionality may implement by Binding the input box to the $Scope object Query properly by using the ng-model Directive.

**3. Function Call and the Data Binding:**

The Function myfirsrtname is actually defined in the Controller to may be called from within the HTML as {{{myFirstName(coach.name)}}}

<div class="row">

<div class="-2u 10u">

<!--Table of Products-->

<ul>

<li ng-repeat="coach in coachList | filter:query | orderBy:orderProp">

<span class ="myspan2"><img ng-src="./img/**{{**coach.image\_url**}}** "></span>

<span id = "myFirstName" class ="myspan">**{{**myFirstName(coach.name)**}}**</span>

<span class ="myspan"> **{{**coach.category**}}**</span>

<span class ="myspan">**{{**coach.price | nfcurrency:"&euro;" **}}**/hr</span>

<span class ="myspan"> **{{**coach.coachemail**}}**</span>

</li>

</ul>

</div>

</div>

</div>

**3.9** **Critical Review of the Strategy adopted**:

**3.9.1 Possibilities Considered:**

Actually at starting stages of the Project there were various possibilities considered. They are:

1. To use Microsoft Visual Studio 2013 for Backend with a Rich Internet Application on Front End.
2. To use PHP for the ‘Back End’ and develop the Client Side with Html5, JavaScript and JQuery.
3. To build a Single Page App by using Angular where the Details to be read from the JSON files.

**3.9.2** **Approach Adopted**:

Actually regarding Ruby on Rails we are not allowed by our lecturer to do it in Ruby on Rails and also it is very difficult to Deploy so that is the main Criticism for Ruby on Rails. At initial attempt to use PHP within Angular JS App it leads to many difficulties and then that’s the main reason that the approach was banned.

Then after trying many attempts it was the developed to “try something new” and then attempt to build a Single page Application by using Angular JS. It’s an Open Source Framework that actually uses to extend a programming language mainly JavaScript. Actually I had no experience in the part of using MVC within the Presentation layer and it was felt that it could be a Great Challenge. Now, Angular is also extremely popular, Easy to Test, Easy to Deploy and its Lightweight.

**3.9.3** **Methods and Strategy:**

Coming to the Methods and Strategies the following is a Summary of Development process involved.

Firstly a Github Account was established for the Version Control.

Link provided here : <https://github.com/indiag/aria>

1. A Skeleton Single Page Application was then developed and deployed on Github Pages
2. SkelsJS functionality was then added.
3. Search, Google Maps API, Google Calendar API, Canvas JS was then developed and incorporated into the Apps.
4. It allows the Data to read from the JSON Files.
5. Then the Tests were written to allow the End to End Testing with Protractor.
6. An attractive UI was then developed.
7. The Application was finally developed on Githubpages.

Link provided here: <http://indiag.github.io/aria/app/#/view1>

**3.9.4 Critical Analysis:**

Our initial learning Curve had over then the Application Development with Angular which was very successful which allowed the expression in Code of the Ideas and Aspiration. Actually all the Technogies used were integrated properly. There were no Conflicts at all. Only one Exception is that JQuery which is not very compatible with Angular JS is that it was generally felt that this would be a Disadvantage.

The Angular JS Implementation of MVC Architecture is allowed for easy Code Development and Integration. Here each Partial has its own Separate Controller and each member of the Team could concentrate on a particular Area which is governed by single Partial and Controller. It was now very easy to integrate everything into full functional Application.

The main benefit of Angular is that Single Page Application was built without having the Developers to implement any Ajax Functionality. Infect the Deployment to the Github pages was actually very easy and it required no Special Modification to the Code Base. All that it required is only a separate Github Branch which is created by git checkout-origin gh-pages command. Now add the Code and finally ‘push’ it by git push origin gh-pages command.

This aim is to actually a main Benefit of Angular JS Approach. The only major Drawback is that the powerful ‘BackEnd’ Functionality provided by Ruby on rails Application is absent.

**4. User Interface Design:**

The main Goal of Designing the User Interface for the project is to make the User Interaction to be very simple and efficient. This describes the process employ to arrive at the Design of the User Interface.

Actually the formulation of the User Interface Design was implemented by many factors. At the Start of the Project I discussed the Requirements of the potential Users but due to the tight time Constraints and less Budget the project Requirements were finally finalized based on the Research rather than full scale functionality Required gathering Techniques such as Interviews or Questionaries. This was agreed that the Web application must have to be simple and functional and it must deliver the specifc Pool information to its members outside of the people. This list mainly included a list of Pool locations nearby the Class by locations Schedules, Workout Statistics etc. Users actually tend to search for the nearby Pool locations when travelling check the Class schedule when they are Back home and check their own Working Statistics when on Train. In the Growing Era the popularity of Smart Phones and are easy to the Internet, the importance of Mobile friendly Design cannot be understated. The solution was actually to implement a Responsive Design which would provide a User Experience throughout the Website across Multiple Devices and for a number of Screen Sizes.

A Responsive Design could be implemented quite efficiently as any of the JavaScript Frameworks such as Skel JS .But due to the level of Skills and Experience recently acquired I have decided to use **Skel JS.**

Actually **SkelJS** is characterized by a 12 Coloumn CSS Grid System which actually is proportional to the Scale of User Interface Elements and it includes a Break point handler which replaces the CSS Queries depending on the type of Media the Application is displayed on. Actually SkelJS is using an easy way to be implemented into the App.

Actually Skel Js is a Single JS file which weight around 20Kb is too light as compared with other libraries. It is a Single Page Application with the Desktop like behaviour can be implemented into Web Based Apps and are rapidly growing in the Popularity and will slowly become the New Standard.So, in order to agree with the Current expectations, the project has been designed and implemented as SPA Conf by using AngularJS. AngularJS is a powerful JS framework which applies MVC structure for the Projects Front End Business logic and it provides a number of features such as Custom, Html Templates, Two Way Binding. The implementation here is in the form of Interface as it only impressed the User Interface but when selecting the Menu Tabs the users are taken from one page to another without the Page load waiting time.

So based on the established Requirements I have concentrated on forming the Information Architecture. So due to this simple nature of App it is decided that the Navigational structure will be very flat and it will have five Tabs in which each is corresponding to the separate Section.

One of the main Goals of User Interface Design was that it implements both Usability and Accessibility principles into the Application. By combing both the Usability and Accessibility it is meant to be creating and satisfying the Disabled people friendly applications. Here the Simplicity of the Structure and the Design is mainly meant to enhance both Accessibility and Usability of the Application.

These are the following features involved:

1. Images should actually contain the Alt Values.
2. Minimal Amount of Colour to prevent the Difficulties for people with Colour Blindness.
3. It should have Consistency in the Design.
4. It should actually use a Large Font and use a high Contrasting Colours such as Black and White on the Text and Background to help the people with the poor Eyesight.

Here I have designed the User Interface the Best Principle and Standards and it provides a high aesthetic appeal of the Design at the same Time Balancing.

**5.Architecture:**

**5.1 Application Architecture:**

Architecture of all the Advanced Rich Internet Applications Project which is actually a Set of a layered Technologies wrapped inside the top layer which produces a Stable Responsive User Interface.

The layers here are designed in such a way that it gives as much as Code Control to

The Programmer. All the Layers are shown below in Figure 1.

SKEL

Angular

Partial1

Partial2

Partial3

Partial4

Partial5

Partial6

Partial....n

Ctrl1

Ctrl2

Ctrl3

Ctrl4

Ctrl5

Ctrl6

Ctrl....n

JavaScript Functions

CSS

Html Page

*return*

Figure 1

1. AngularJS

2. SkelJS

3 .Angular JS Partials

4.. Angualar Controllers

5. Canvas JS

6. Html and Css Integration

7. JavaScript Control within the Angular

The layers are actually linked or wrapped in the depending on which Technology it depends on one another. The Diagram with all these structures is illustrated below:

Here actually all the layers above has a predefined Action which is defined by Arrows:

The SkelJS is a Framwork that actually reacts to the Size and with the Container that it is defined and here in this Case the Browser is actually the Container so that the Design of the Web Application is mainly the based on a Responsive Site that is actually compatible with the Mobile as a Mobile App.

Here in this Web Application ARIA project the SkelJS Framework actually acts as a Wrapper around the Angular Toolset so that there would be a Development of the Angular Framework and it will behave responsively because it has the SkelJS Container in it.

SkelJS is the Topmost layer and hence all the Angular Code and Dependency Toolset Code which means that the Partial Html, Partial JavaScript etc. act very Responsively with the Top SkelJS Layers.[4]

**5.2 Security:**

Skel and Angular a Security Risk: Here now a Days Internet is a wide open network so that so Security will be the major issue with the Client Side Scripting.

The major Drawback for the Security is that the JavaScript does not actually supports the Read/Write to the Local files so it can list the files on a Client Personal Computer. Actually Angular and SkelJS have a Control Dependency on Java Script so that they actually do not pose a Security Risk. Here JavaScript cannot use Clients Computer as a Platform for attacking or attempting a Personal Crack on the other Network.

**5.3 The Angular Defence with against Security Risks**

Even though there are some basic level of Security Issues which are adopted by JavaScript there will infect be more Issues that are still existing

Here the Angular Toolset is actually programmed to be a Single Page Application so that it can have layers of the Partial Html Pages within the main Angular Page. Here actually each Partial Html Page is called from the main Angular Menu System and it is then displayed as one Web page of Information.

The Partial Pages will have a Dependency on Angular Contollers which are a Composite of both Java Script and J Query. These Scripts uses all the Controllers for sending or Calling the functions and by accepting a Return value.

Here actually the Angular System Web pages uses a REST Type Metholodogy for the Navigation by using a designated Naming Convention such as the /View 1../View 2….etc.

Here there is a Typical Url format and its shown below:

Protocol/Host/Path/#/View1

The Partial loads as the View 1 or whatever Designated Description it is assigned to the Partial within the Code.

See figure 2

<!-- Header -->

<div class="row flush header">

<div id="irish pool logo.jpg"></div>

</div>

<nav class="**{{**active**}}**">

<a href="#/app/view1" ng-click="active='home'" class="home">Home</a>

<a href="#/view2" ng-click="active='partialTwo'" class="partialTwo">Find a Coach</a>

<a href="#/view3" ng-click="active='partialThree'" class="partialThree">Pool Locations</a>

<a href="#/view4" ng-click="active='partialFour'" class="partialFour">Workout Statistics</a>

<a href="#/view5" ng-click="active='partialFive'" class="partialFive">Calendar</a>

<a href="#/view6" ng-click="active='partialSix'" class="partialSix">Log in</a>

</nav>

<!-- Hero -->

<section>

<div ng-view></div>

</section>

<div id="loginid" >You have Logged in successfully</div>

<!-- Footer -->

Figure 2

Actually the main Idea is to stop the JavaScript programs from exporting the Private Information about the User. Here in this Case the Angular is implements on a different level in which the AJAX is based on the Client Framework. By this Reason the User by passes all the Security measures that has really implemented on the Client Side in Real Time. So hence all the Security is been implemented on the Server Side so that any Response will be Authorized.

So here actually the Data Rendering is not exactly done on the Client Side because it is a Security Precaution.

**5.4 Disadvantages of Angular with Security Issues:**

1. The Object is considered as Global in a Browser.
2. The Ajax and JavaScript Openness.
3. The XSS Attacks.
4. Scripts can be interacted only with the same Objects.[3] [8]

**5.5 Data Transfer Strategies:**

The Data actually was read directly from the JSON files from the Angular Controller (/app/compilers.js)

For Instance, take the Data in the Coaches.json available to the Application which is the Partial 2 an Http Request to the File is made from within the Relevant Controller (my Ctrl2) as the following Code

.controller('MyCtrl2', function ($scope, $http) {

$http.get('./sCoaches.json').success(function (data) {

$scope.coachList = data;

});

$scope.orderProp = 'name';

$scope.HTMLalpha = "Alphabetical";

$scope.HTMLlowestPrice = "Lowest Price";

$scope.HTMLhighestPrice = "Highest Price";

$scope.HTMLCategory = "Category";

$scope.HTMLemail = "Email";

$scope.HTMLcreated\_at = "Created At";

$scope.HTMLupdatedAt = "Updated At";

$scope.HTMLid = "Id";

//$scope.orderPropAlt = 'email';

$scope.myFirstName = function (string) {

return string.split(' ')[0]

}

})

The Data will again be available and may be Incorporated into the Html as below:

<li ng-repeat="coach in coachList | filter:query | orderBy:orderProp">

<span class ="myspan2">

<a href="#/view7" ng-click="active='partialSeven'" class="partialSeven"><img ng-src="./img/**{{**coach.image\_url**}}** "></a>

</span>

<span id = "myFirstName" class ="myspan">**{{**myFirstName(coach.name)**}}**</span>

<span class ="myspan"> **{{**coach.category**}}**</span>

<span class ="myspan">**{{**coach.price | nfcurrency:"&euro;" **}}**/hr</span>

<span class ="myspan"> **{{**coach.coachemail**}}**</span>

**5.6 Evaluation and Testing:**

The Testing was actually done by using the Protractor which is an End to End Test Framework for the Angular JS. Link provided Here: (<https://github.com/angular/protractor>). By this is actually allows the End to End Testing of the App where the Tests are Run on the Actual Website itself.

The Framework here actually requires the Node.js (<http://nodejs.org/>) and the Selenium Web Driver is installed. Link provided here <http://docs.seleniumhq.org/download/>.

Here actually in Partial 3.html there is a Button that actually calls a Function called as Update fn(). Here the Update Function is defined within a Controller 3 and a specific Pool location is displayed dynamically depending upon the View chosen by the User in a Drop down Menu.

In order to actually run the Test from the Ubuntu System only the Following Commands should be needed to be executed.

1. Npm start
2. Protractor. /test/protractor-conf.js

**Some sample Testing Scenarios are shown below**:

'use strict';

describe('my app', function () {

browser.get('index.html');

it('should automatically redirect to /view1 when location hash/fragment is empty', function () {

expect(browser.getLocationAbsUrl()).toMatch('view1');

});

describe('view2', function () {

beforeEach(function () {

browser.get('index.html#/view2');

});

it('should render view2 when user navigates to /view2'

, function () {

expect(element.all(by.css('[ng-view] h4')).first().getText()).

toMatch(/Find a Pool Coach/);

});

});

describe('view3', function () {

beforeEach(function () {

browser.get('index.html#/view3');

});

it('should render view3 when user navigates to /view3', function () {

expect(element.all(by.css('[ng-view] h1')).first().getText()).

toMatch(/Find Our Pool Locations/);

});

});

describe('view4', function () {

beforeEach(function () {

browser.get('index.html#/view4');

});

it('should render view4 when user navigates to /view4', function () {

expect(element.all(by.css('[ng-view] h1')).first().getText()).

toMatch(/Workout Data/);

});

});

describe('view5', function () {

beforeEach(function () {

browser.get('index.html#/view5');

});

it('should render view5 when user navigates to /view5', function () {

expect(element.all(by.css('[ng-view] iframe')).first().getText()).

toMatch('');

});

});

describe('Gymnasium App Title', function () {

beforeEach(function () {

browser.get('index.html');

});

describe("index", function () {

it('should display the correct title', function () {

expect(browser.getTitle()).toBe('Pool App')

});

});

});

describe('ng-bind-html', function () {

beforeEach(function () {

browser.get('index.html#/view2');

});

describe("sort options", function () {

it('should check ng-bind-html', function () {

expect(element(by.binding('HTMLalpha')).getText()).toBe(

'Alphabetical');

expect(element(by.binding('HTMLlowestPrice')).getText()).toBe(

'Lowest Price');

expect(element(by.binding('HTMLhighestPrice')).getText()).toBe(

'Highest Price');

expect(element(by.binding('HTMLCategory')).getText()).toBe(

'Category');

expect(element(by.binding('HTMLemail')).getText()).toBe(

'Email');

expect(element(by.binding('HTMLcreated\_at')).getText()).toBe(

'Created At');

expect(element(by.binding('HTMLupdatedAt')).getText()).toBe(

'Updated At');

expect(element(by.binding('HTMLemail')).getText()).toBe(

'Email');

});

});

});

describe('Repeater Function', function () {

beforeEach(function () {

browser.get('index.html#/view2');

});

describe('Number of items', function () {

it('should have twelve items', function() {

var elems = element.all(by.repeater('coach in coachList'));

expect(elems.count()).toBe(12);

});

});

});

describe('ng-click functions', function () {

beforeEach(function () {

browser.get('index.html#/view3');

});

describe("Google maps function", function () {

it('should show correct text for pool location ', function () {

var ptor = protractor.getInstance();

//This will not get the option required

expect(ptor.element(by.css('#l1')).getText()).toMatch('Parnell Street');

ptor.findElement(protractor.By.css('#addressinput option:nth-child(2)')).click();

ptor.findElement(protractor.By.css('#Button1')).click();

expect(ptor.findElement(protractor.By.css('#l1')).getText()).toMatch('Arjun Nagar, Agra, India');

ptor.findElement(protractor.By.css('#addressinput option:nth-child(3)')).click();

ptor.findElement(protractor.By.css('#Button1')).click();

expect(ptor.findElement(protractor.By.css('#l1')).getText()).toMatch('Malakpet,Hyderbad, India');

});

});

});

describe('Search Functionality', function () {

beforeEach(function () {

browser.get('index.html#/view2');

});

describe('Dynamic Search Box', function () {

it('should return first name', function () {

var ptor = protractor.getInstance();

//This will not get the option required

ptor.findElements(protractor.By.repeater("coach in coachList"));

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('Balaji');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('Balaji O Shea');

});

it('should return the correct sort-by result', function () {

var ptor = protractor.getInstance();

ptor.findElement(protractor.By.css('#sortBy option:nth-child(2)')).click();

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('James');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('James Mason');

ptor.findElement(protractor.By.css('#sortBy option:nth-child(3)')).click();

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('Henry');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('Hungry Henry');

ptor.findElement(protractor.By.css('#sortBy option:nth-child(4)')).click();

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('Balaji');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('Balaji Murphy');

ptor.findElement(protractor.By.css('#sortBy option:nth-child(5)')).click();

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('Balaji');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('Enda Kenny');

ptor.findElement(protractor.By.css('#sortBy option:nth-child(6)')).click();

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('karthik');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('karthik Jones');

});

it('should return the correct search result', function () {

var ptor = protractor.getInstance();

ptor.findElement(protractor.By.id("ngsearch")).sendKeys('balaji');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('Balaji');

ptor.findElement(protractor.By.id("ngsearch")).clear();

ptor.findElement(protractor.By.id("ngsearch")).sendKeys('Jimmy');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('Jimmy');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('Jimmy Johnston');

ptor.findElement(protractor.By.id("ngsearch")).clear();

ptor.findElement(protractor.By.id("ngsearch")).sendKeys('Karthik');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('karthik');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('karthin');

ptor.findElement(protractor.By.id("ngsearch")).clear();

ptor.findElement(protractor.By.id("ngsearch")).sendKeys('Kiran');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('Kiran');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('Kieran');

ptor.findElement(protractor.By.id("ngsearch")).clear();

ptor.findElement(protractor.By.id("ngsearch")).sendKeys('Latif');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('Latif');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('Gatif');

ptor.findElement(protractor.By.id("ngsearch")).clear();

ptor.findElement(protractor.By.id("ngsearch")).sendKeys('Pavan');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('Pavan');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('Aavan');

ptor.findElement(protractor.By.id("ngsearch")).clear();

ptor.findElement(protractor.By.id("ngsearch")).sendKeys('Venkat');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('Venkat');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('Vikas');

ptor.findElement(protractor.By.id("ngsearch")).clear();

ptor.findElement(protractor.By.id("ngsearch")).sendKeys('Vivek');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).toMatch('Vivek');

expect(ptor.findElement(protractor.By.css('#myFirstName')).getText()).not.toMatch('Livek');

});

});

});

describe('nav bar functionality', function () {

beforeEach(function () {

browser.get('index.html#/view2');

});

describe('Click on menu links', function () {

it('should visit the correct page', function () {

var ptor = protractor.getInstance();

//This will not get the option required

ptor.findElement(protractor.By.css('.home')).click();

expect(browser.getLocationAbsUrl()).toMatch('view1');

expect(browser.getLocationAbsUrl()).not.toMatch('view2');

ptor.findElement(protractor.By.css('.partialTwo')).click();

expect(browser.getLocationAbsUrl()).toMatch('view2');

expect(browser.getLocationAbsUrl()).not.toMatch('view1');

ptor.findElement(protractor.By.css('.partialThree')).click()

expect(browser.getLocationAbsUrl()).toMatch('view3');

expect(browser.getLocationAbsUrl()).not.toMatch('view1');

ptor.findElement(protractor.By.css('.partialFour')).click()

expect(browser.getLocationAbsUrl()).toMatch('view4');

expect(browser.getLocationAbsUrl()).not.toMatch('view1');

ptor.findElement(protractor.By.css('.partialFive')).click()

expect(browser.getLocationAbsUrl()).toMatch('view5');

expect(browser.getLocationAbsUrl()).not.toMatch('view1');

});

});

});

});

Hence this is a great way to implement the Test Driven Development. The main Benefit is that all the Teats are being run on the Website. One Disadvantage is that is actually very difficult for sure to define if the Content such as Google Maps were loaded.

**6. Conclusion:**

So lastly here is a Single Page Application developed with a MVC Architecture which has been developed and deployed by using the Angular JS. This App is mainly known as the IrishPool Website with all the recent Technologies involved in it so that it can be used in the Future too. For this we developed the Responsive Website as now days many users are surfing Internet on Mobiles, Tablets. So in order to look good we make our Site Responsive with the help of Skel.js. Here actually the User might Search for the Pool Coach by using the Dynamic Search Functionality and also View the Location of a Pool by the Google Maps provided. Here the User can also View the Exercise and Workout Statistics or can even Record an Event by using the Google Calendar and finally this Site has been deployed on the Git Hub Pages.

<http://indiag.github.io/aria/app/#/view1>

**7 .References:**

[1]. AngularJS - Wikipedia, the free encyclopedia. 2014. AngularJS - Wikipedia, the free encyclopedia. [ONLINE]

Available at: http://en.wikipedia.org/wiki/AngularJS. [Accessed 30 May 2014]

[2]. Google Developers. 2014. Google Developers. [ONLINE] Available at: https://developers.google.com/maps/.

[3]. AngularJS Fundamentals In 60-ish Minutes - YouTube. 2014. AngularJS Fundamentals In 60-ish Minutes - YouTube. [ONLINE] Available at:https://www.youtube.com/watch?v=i9MHigUZKEM. [Accessed 30 May 2014].

[4]. skelJS : Documentation. 2014. skelJS : Documentation. [ONLINE] Available at: http://skeljs.org/docs. [Accessed 30 May 2014].

[5]. HTML Tutorials (thesitewizard.com). 2014. HTML Tutorials (thesitewizard.com). [ONLINE] Available at: http://www.thesitewizard.com/html-tutorial/index.shtml. [Accessed 30 May 2014].

[6]. Single-page application - Wikipedia, the free encyclopedia. 2014. Single-page application - Wikipedia, the free encyclopedia. [ONLINE] Available at: http://en.wikipedia.org/wiki/Single-page\_application. [Accessed 03 June 2014].

[7]. Practical End-to-End Testing with Protractor | ng-newsletter. 2014. Practical End-to-End Testing with Protractor | ng-newsletter. [ONLINE] Available at: http://www.ng-newsletter.com/posts/practical-protractor.html. [Accessed 06 June 2014].

[8]. 2014. . [ONLINE] Available at: https://docs.angularjs.org/guide/introduction. [Accessed 06 June 2014].