Rest APIs Application

Project Requirements Document

CNLjunggren

<https://github.com/CNLjunggren>

Revision 0.47

Date: 3/31/2020

**Table of Contents:**

[**Document Changes/Documentation History:** 2](#_Toc36544963)

[Document Change Log 2](#_Toc36544964)

[**Use Cases:** 3](#_Toc36544965)

[User: 3](#_Toc36544966)

[Admin: 4](#_Toc36544967)

[**System Design:** 5](#_Toc36544968)

[Top-Down Approach: 5](#_Toc36544969)

[Legend for Diagram: 5](#_Toc36544970)

[**Technical Requirements:** 6](#_Toc36544971)

[System Technical Requirements 6](#_Toc36544972)

[Software/Hardware Requirements 6](#_Toc36544973)

[**System Logical Model:** 8](#_Toc36544974)

[**Reports:** 9](#_Toc36544975)

[User Results 9](#_Toc36544976)

[Weather Results 9](#_Toc36544977)

[Transaction Results 10](#_Toc36544978)

[**Screen Definitions and Layouts:** 11](#_Toc36544979)

[Home Page: 11](#_Toc36544980)

[When not logged in: 11](#_Toc36544981)

[User/Admin view: 12](#_Toc36544982)

[Dropdown Menus (Admin View): 12](#_Toc36544983)

[Registration Page: 13](#_Toc36544984)

[Login Page: 14](#_Toc36544985)

[User Management Page (Admin-Only): 15](#_Toc36544986)

[OpenWeather – Weather Request Page: 16](#_Toc36544987)

[OpenWeather – Weather Request Results Page: 17](#_Toc36544988)

[Current Weather Results: 17](#_Toc36544989)

[Weather Results Error: 17](#_Toc36544990)

[Weather Forecast Results: 18](#_Toc36544991)

[OpenWeather – Weather History Page: 19](#_Toc36544992)

[User view: 19](#_Toc36544993)

[Admin view: 20](#_Toc36544994)

[: 21](#_Toc36544995)

[**Security:** 22](#_Toc36544996)

[Security Matrix 22](#_Toc36544997)

[**Other (As dictated by the context and scope of the project):** 23](#_Toc36544998)

[**Sources:** 24](#_Toc36544999)

# **Document Changes/Documentation History:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Document Change Log | | | | |
| **ID** | **Change Description** | **Document Status** | **Date Entered** | **Document Revision #** |
| 1 | Original upload to repository. Included completed Use Cases, System Design, and Technical Requirements sections. | Incomplete | 2/29/2019 | 0.3 |
| 2 | Update to this document with the Reports section completed as well as the Wireframe views for the Home, Login, and Registration pages for the Screen definitions and Layouts section. | Incomplete | 3/8/2019 | 0.45 |
| 3 | Added the missing wireframe views for the OpenWeather pages and revised the OpenWeather Results page for both current and forecast related results. | Incomplete | 3/9/2019 | 0.46 |
| 4 | Updated the documentation to include the security section, listing known security issues the application would bring and what solutions will be in place to combat them. Also included a security matrix | Incomplete | 3/31/2019 | 0.47 |
|  |  |  |  |  |

# **Use Cases:**

## User:

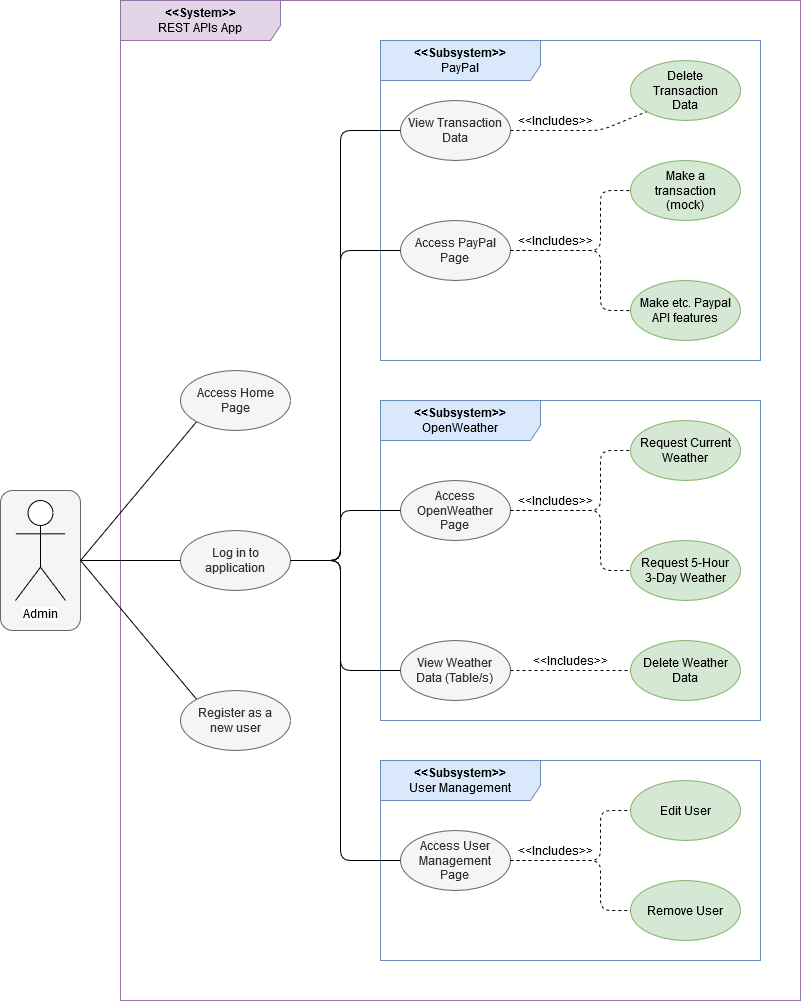
A screenshot of a cell phone

Description automatically generated

This application utilizes two roles that make up the functions and uses for this site. A description of both roles and their available actions is provided below;

* User: The standard role of the application that can log into the application, register as a new user, and access the home page. Once logged in, the User has access to the OpenWeather and PayPal API service pages. From here the user can:
  + Make a mock PayPal transaction.
  + Make requests for etc. PayPal APIs.
* Admin (Diagram on next page): The administrator tole of the application that shares all the actions of a User and more. The administrator actions that only an Admin can take are:
  + Access the User Management Page to view and edit/remove existing users from the database.
  + Access the Weather Management Page to view and remove weather data from the database.
  + Access the Transaction Management Page to view and remove weather data from the database.
    - Not all Transaction data will be visible for privacy/confidentiality reasons.

## Admin:



# **System Design:**

## Top-Down Approach:

A close up of a sign

Description automatically generated

## Legend for Diagram:

* Rounded Rectangles: Webpages/Views.
  + Blue: Standard User access level and higher.
  + Orange: Admin access level only.
* Rectangles: Functions/Methods groups.
  + Green: Standard User access level and higher.
  + Yellow: Admin access level only.

# **Technical Requirements:**

|  |  |
| --- | --- |
| System Technical Requirements | |
| Requirement | Description |
| Database | This project depends on a database that can store and allow for CRUD management of its entries for at least users, weather response data, and transaction data. |
| OpenWeather APIs Page | This web project requires at least one data validation supported form page for the OpenWeather APIs that allows for a user to select the type of weather call they want to request along with the location and measurement system used for the data returned. There will at least be one response page dedicated for displaying this data and error messages. |
| PayPal APIs Page | This web project requires at least one data validation supported form page for PayPay’s Sandbox REST API that allows for a user to Make and process a mock transaction. There will at least be one response page dedicated for displaying a confirmation or error message. |
| Data Management Pages | This web project requires at least three pages dedicated for database data viewing and modification through CRUD functionality. This includes a table and CRUD support each for users, transaction history, and weather data. Sensitive information will not be displayed or viewable by admins for security reasons. |

|  |  |
| --- | --- |
| Software/Hardware Requirements | |
| Requirement | Description |
| Microsoft Visual Studio  (C# & AspNet packages) | Microsoft Visual Studio is an IDE made by Microsoft that supports various languages and functionalities. For this project, I will be utilizing the C# language with AspNet’s .NET MVC framework. |
| Microsoft.AspNet.MVC (v5.2.4) | Microsoft .Net MVC is an MVC framework for C# that provides the required aspects for an MVC web application. This includes components such as Razor, web page functionality, and web optimization. |
| Microsoft.AspNet.Razor  (v3.2.4) | Razor is a portion of .Net MVC that is used to create the project’s website pages. |
| Microsoft.AspNet.WebPages  (v3.2.4) | WebPages is a portion of .Net MVC that that contains the core assemblies for .Net MVC web pages. |
| Microsoft.Bcl.AsyncInterfaces  (v1.1.0) | Provides the IAsyncEnumerable<T> and IAsyncDisposable interfaces and helper types for .NET Standard 2.0. |
| Microsoft.CodeDom.Providers.  DotNetCompilerPlatform  (v2.0.0) | Replacement CodeDOM providers that utilize the new .NET Compiler Platform (“Roslyn”) compiler as a service APIs. |
| Microsoft.Web.Infrastructure  (v1.0.0) | Microsoft Web infrastructure is an assembly that is used in web projects for dynamic HTTP module registering at run time. |
| System.Buffers (v4.5.0) | Provides resource pooling of any type for performance-critical applications that allocate and deallocate objects frequently. |
| System.Memory (v4.5.3) | Provides types for efficient representation and pooling of managed, stack, and native memory segments and sequences of such segments, along with primitives to parse and format UTF-8 encoded text stored in those memory segments. |
| System.Numerics.Vectors  (v4.5.0) | Provides hardware-accelerated numeric types, suitable for high-performance processing and graphics applications. |
| System.Runtime.  CompilerServices.Unsafe  (v4.7.0) | Provides the System.Runtime.CompilerServices.Unsafe class, which provides generic, low-level functionality for manipulating pointers. |
| Systems.Text.Encoding.Web  (v4.7.0) | Provides types for encoding and escaping strings for use in JavaScript, HyperText Markup Language (HTML), and uniform resource locators (URL). |
| System.Threading.Tasks.  Extensions (v4.5.3) | Provides additional types that simplify the work of writing concurrent and asynchronous code. |
| System.Value.Tuple (v4.5.0) | Provides the System.ValueTuple structs, which implement the underlying types for tuples in C# and Visual Basic. |
| Bootstrap (v4.4.1) | Bootstrap is going to be utilized alongside razor to create the visual appearances of the project’s website. Bootstrap is a library/framework for CSS. |
| popper.js (v1.16.0) | A requirement when installing version 4.4.1 of Bootstrap. Popper serves as a library that allows you to manage your poppers. |
| Modernizr (v2.6.2) | Modernizr is a JavaScript library used for CSS3 and HTML 5 technologies which is a big help in creating HTML web pages. |
| jQuery (v3.4.1) | jQuery is a JavaScript library used for various HTML, event handling, etc. simplification and ease of use applications. |
| jQuery Validation (v1.19.1) | The Validation jQuery plugin is used for data validation simplification and customization. |
| Microsoft.jQuery.Unobtrusive.  Validation (v3.2.11) | jQuery plugin that unobtrusively sets up jQuery.Validation. |
| Newtonsoft.Json (v12.0.3) | A high-performance JSON framework for use with the .NET frameworks. This framework will be used for JSON file deserialization from API responses, and possibly to serialize data into JSON files. |
| Microsoft SQL Server | SQL Server is going to be the database used for this project as it is a built-in database system in Visual Studio. |

# **System Logical Model:**

# **Reports:**

|  |  |  |
| --- | --- | --- |
| User Results | | |
| Display Value | Variable or Field | Format Notes |
| ID | User.Id | Number, Integer |
| Username | User.Username | String, 50 characters max |
| Password | User.Password | String, 50 characters max, at least one capital letter and #. (Censored for privacy) |
| Email Address | User.Email | Email (Data Validation enforced string?) |
| Role | User.Role | Number, Integer |
| Birth Date | User.BirthDate | Date format, “xx-xx-xxxx” |

|  |  |  |
| --- | --- | --- |
| Weather Results | | |
| Display Value | Variable or Field | Format Notes |
| ID | Weather.Id | Number, Integer |
| Longitude | Weather.CoordLon | Number, Double |
| Latitude | Weather.CoordLat | Number, Double |
| Weather Type | Weather.WeatherMain | String, 50 characters max |
| Weather Description | Weather.WeatherDesc | String, 50 characters max |
| Weather Icon | Weather.WeatherIcon | String, 50 characters max, partial URL of an OpenWeather icon. |
| Temperature | Weather.MainTemp | Number, Double |
| Feels Like | Weather.MainTempFeel | Number, Double |
| Pressure | Weather.MainPressure | Number, Integer |
| Humidity | Weather.MainHumid | Number, Integer |
| Temperature Low | Weather.MainTempMin | Number, Double |
| Temperature High | Weather.MainTempMax | Number, Double |
| Visibility | Weather.Visibility | Number, Integer |
| Wind Speed | Weather.WindSpeed | Number, Double |
| Wind Direction | Weather.WindDeg | Number, Integer |
| Cloud Coverage | Weather.CloudCover | Number, Integer |
| Rain: 1 Hour | Weather.Rain1H | Number, Double |
| Rain: 3 Hour | Weather.Rain3H | Number, Double |
| Snow: 1 Hour | Weather.Snow1H | Number, Double |
| Snow: 3 Hour | Weather.Snow3H | Number, Double |
| Time of Request | Weather.Dt | Number, Integer |
| Country Code | Weather.SysCountry | String, 50 characters max |
| Sunrise | Weather.SysSunrise | Number, Integer |
| Sunset | Weather.SysSunset | Number, Integer |
| City | Weather.Name | String, 50 characters max |
| Icon URL | Weather.IconURL | String, 50 characters max, URL of an OpenWeather icon. |

|  |  |  |
| --- | --- | --- |
| Transaction Results | | |
| Display Value | Variable or Field | Format Notes |
| ID | Transaction.Id | Number, Integer |
| Profile ID | Transaction.ProfileId | String, 50 characters max |
| Profile Name | Transaction.ProfileName | String, 50 characters max |
| Temporary? | Transaction.Temporary | Boolean (true or false) |
| Landing Page Type | Transaction.FlowPageType | String, 50 characters max |
| Bank Pending URL | Transaction.FlowPending | String, MAX characters (Censored for privacy) |
| Shipping? | Transaction.InputShipping | Boolean (true or false) |
| Address Override? | Transaction.InputAddress | Boolean (true or false) |
| Logo Image | Transaction.LogoUrl | String, 50 characters max |

# **Screen Definitions and Layouts:**

## Home Page:

### *When not logged in:*

A screenshot of a cell phone

Description automatically generated

### *User/Admin view:*

A screenshot of a cell phone

Description automatically generated

### *Dropdown Menus (Admin View):*

A screenshot of a cell phone

Description automatically generated

## Registration Page:

A screenshot of a cell phone

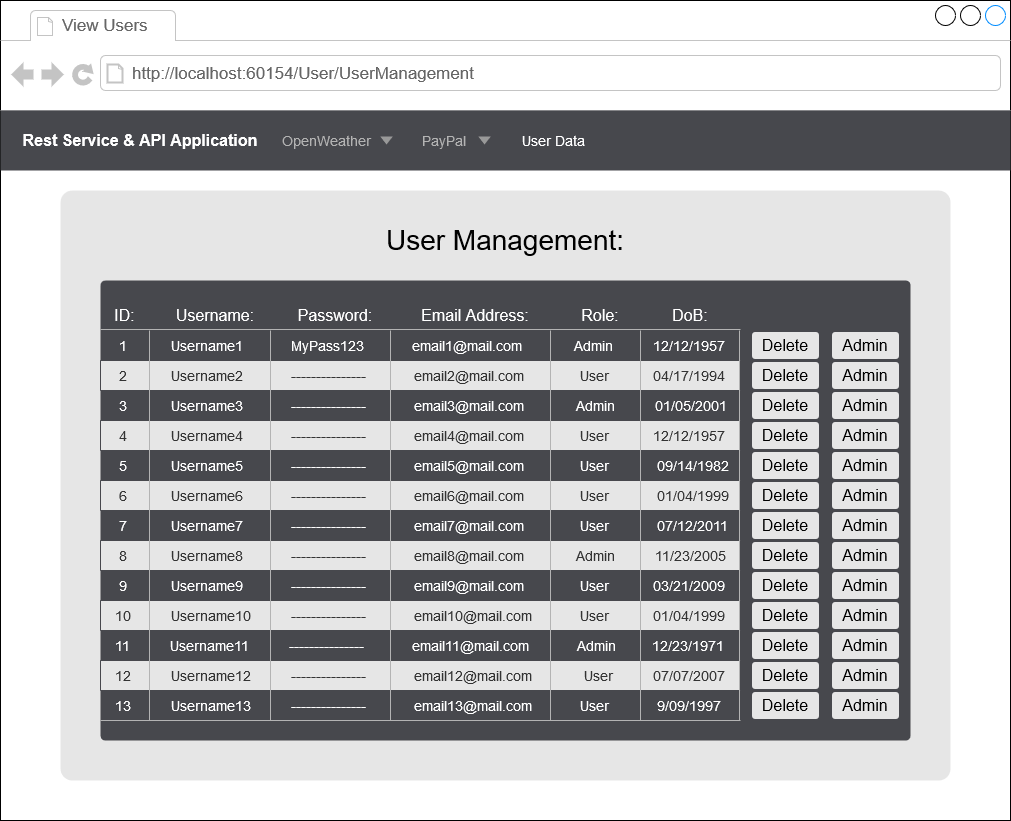
Description automatically generated

## Login Page:

A screenshot of a cell phone

Description automatically generated

## User Management Page (Admin-Only):



## OpenWeather – Weather Request Page:

A screenshot of a cell phone

Description automatically generated

## OpenWeather – Weather Request Results Page:

### *Current Weather Results:*

A screenshot of a cell phone

Description automatically generated

### *Weather Results Error:*

A screenshot of a cell phone

Description automatically generated

### *Weather Forecast Results:*

A screenshot of a cell phone

Description automatically generated

## OpenWeather – Weather History Page:

### *User view:*

A screenshot of a cell phone

Description automatically generated

### *Admin view:*

A screenshot of a cell phone

Description automatically generated

## :

# **Security:**

With this application relying on third party APIs and supporting user account functionality, there are several security concerns that could compromise the application. This includes:

* Users could attempt to bypass site security by entering the path URLs to webpages they don’t have access to.
  + Solution: Implement a call that can be reused throughout the entire application to check the user’s role before loading a page or attempting an admin-restricted process. Implemented.
* Users might attempt to register or login while already logged in, which could have unexpected results on the website.
  + Solution: Implement a call that can be reused throughout the entire application to check if the user is already logged in, forcing a logout in order to access the page. Implemented.
* Users can attempt to exploit the web application/database through form inputs such as through overflow attacks.
  + Solution: Utilize data validation to restrict the type of data users can enter, especially the length and types of characters that can be entered. Partially Implemented.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Security Matrix | Users | Nonuser | User | Admin |
|  |  |  |  |  |
| Access Home page |  | X | X | X |
| Access Login page. |  | X |  |  |
| - Log into the application as an existing user. |  | X |  |  |
| Access Registration page. |  | X |  |  |
| - Register to the site as a new user. |  | X |  |  |
| Access OpenWeather functionality. |  |  | X | X |
| - Request the current weather from a specific location. |  |  | X | X |
| - Request the weather forecast from a specific location. |  |  | X | X |
| Access OpenWeather admin functions. |  |  |  | X |
| - View weather request data. |  |  |  | X |
| - Delete a given weather request from the database. |  |  |  | X |
| Access admin User Management. |  |  |  | X |
| - View all users from the database. |  |  |  | X |
| - Change the role of an existing user. |  |  |  | X |
| - Delete a user (non-admin) from the database. |  |  |  | X |
|  |  |  |  |  |

# **Other (As dictated by the context and scope of the project):**

# **Sources:**