UNIVERSITY OF ABERDEEN

ABERDEEN
CITY COUNCIL

ABERDEEN CITY COUNCIL DATA OBSERVATORY MAINTENANCE GUIDE

Prepared by Team Alpha

Alga Abilesh
Dimitar Bakardzhiev
Lachezar Kostov
Maksim Lilchev
Real Bakhit
Samuel Kwakye

Table of Contents

Introduction

User Maintenance Guide Introduction

This User Maintenance Guide is designed to provide the admin users of the ACCDO with instructions how to install, run and maintain the application. The Maintenance Guide provides instruction for local and hosted installation. As the application was designed for internal use in ACC, the instructions for hosted installation are for Windows Server environment.

System Description

The application is developed in C# using Visual Studio .NET 2017 framework. It is implemented as .NET MVC 4.6 and Web API 2.0 application. The database is created using Entity Framework 6.2 Code First Migrations. The user interface is implemented using HTML, CSS, JavaScript and jQuery. The UI uses AJAX calls to consume the API.

Installation instructions

Local installation

To install and run the application locally follow the steps below:

1. Install Visual Studio .NET on your system.
2. In Visual Studio .NET go to File >> Open >> Project/Solution.
3. Navigate to the application folder and open the ACCDV.sln file.
4. In Visual Studio, navigate to Tools >> NuGet Package Manager >> Manage NuGet Packages for solution. Update the packages for the solution and that will install any missing packages. Agree to any installation agreements and proceed with the installation.
5. Open the Solution Explorer from the View menu in Visual Studio.
6. In Solution Explorer find the Web.config file in the ACCDV project folder and open it.
7. In the Web.config file find <connectionStrings> element and change the path to the ACCDV-DB.mdf file in the AttachDbFilename in the connectionString attribute of the DefaultConnection with the path to the ACCDV-DB.mdf in your system.
8. Run the application using the running IIS Express option.
9. The application will open in a web browser.

Hosted installation (Windows Server)

Once installed locally, the application can be packaged and deployed to a hosted server environment. To deploy and run the application on Windows Server follow the steps below:

1. Navigate to the project folder and open ACCDV.sln file in Visual Studio.
2. Open the Solution Explorer from the View menu.

3. In the Solution Explorer, right click on the project folder and select Publish.
4. In the Publish window, go to the Publish tab, “publish your app to Azure or another host”, and click on Start.
5. On the new window select “IIS, FTP, etc” then select “configure” and press Create Profile.
6. On the new window select “Connection” and for publish method select “File System”. In target location pick a directory (or create a new one) on your system and click Next.
7. On the Settings window select “Release” under the Configuration and select Save.
8. Press “Publish” in the publish window.
9. Upload the directory containing the published application to your Windows Server.
10. On the Windows Server’s pool make sure that .NET 4.5 or is installed.
11. On the Windows Server, go to ISS, right click on it and select “Add Application”.
12. In the Add Application Wizard enter alias name for your application and select the folder that contains the application and the server pool, and then create.
13. Install SQL Server or SQL Express and SQL Server Management Studio if not already installed.
14. Open SQL Server Management Studio, expand the SQL server directories, right click on “Databases” and select “Attach”.
15. In the new window select “Add” and navigate to the application folder, open the App-Data and select the database file (.mdf).
16. Change the path of the .mdf db file in the Web.config to point to the file in the application directory.

Third-party software

The application uses third-party JavaScript libraries to perform some of the functionalities in the user interface. To render charts the application uses Chart.js, which can be downloaded at https://www.chartjs.org/, and HighCharts available at https://www.highcharts.com/. For styling the interface we use Bootstrap CSS theme from https://getbootstrap.com/. To display animated spinner while the data is loading on the page, we use Spin.js, available at https://spin.js.org/. To format the currency values in the housing tables the app uses Numeral.js, available at http://numeraljs.com/.
All the third-party libraries are bundled in the BundleConfig.cs file inside the App_Start folder.

The app uses an embedded Google Map from https://cloud.google.com/maps-platform/, with a unique access token for the application.

File system and packages

The file system has the typical structure of a .Net MVC5 application. It has separate folders for Models, Views and Controllers. There is a Migrations folder, created by the Entity Framework, containing all the migrations and configuration files to create the database schema.

There are two additional files. The first one is Filters and it contains the AjaxOnlyAttribute.cs file with the AjaxOnlyAttribute class, responsible for filtering the requests to the API. The second additional folder is CSV and it contains all the csv files with the data needed to populate the database tables. The files in the CSV folder are used by the Seed method of the Configuration.cs file from the Migrations folder to seed the database when it is initially created.

The application uses NuGet packages, which can be found in the main project folder in a folder named Packages. The packages can be managed using the NuGet Package Manager in the Visual Studio. They can also be configured in the packages.config file.

Scaling and extending the application

The application is designed with scalability and extendibility in mind. Using the Entity Framework code first migrations, new tables with data can be easily added to the database without the need to drop and recreate the schema. To add a table to the database, write the appropriate Model class in the Models folder, generate a migration using the NuGet Package Manager Console and update the database. The table can be seeded with data using the seed method in the Configuration.cs file in the Migrations folder.

The application architecture includes a Web API. New API controllers and endpoints can be easily added to the pipeline. The use of API decouples the interface from the rest of the application and provides the flexibility to replace the interface if desired or consume the data from different interfaces simultaneously.

The interface is structured in different sections, containing different graphical representation of the data, and it can be easily extended to display information from additional datasets.

Using Web API allows the current architecture of the application to be further broken down into microservices architecture if desired.