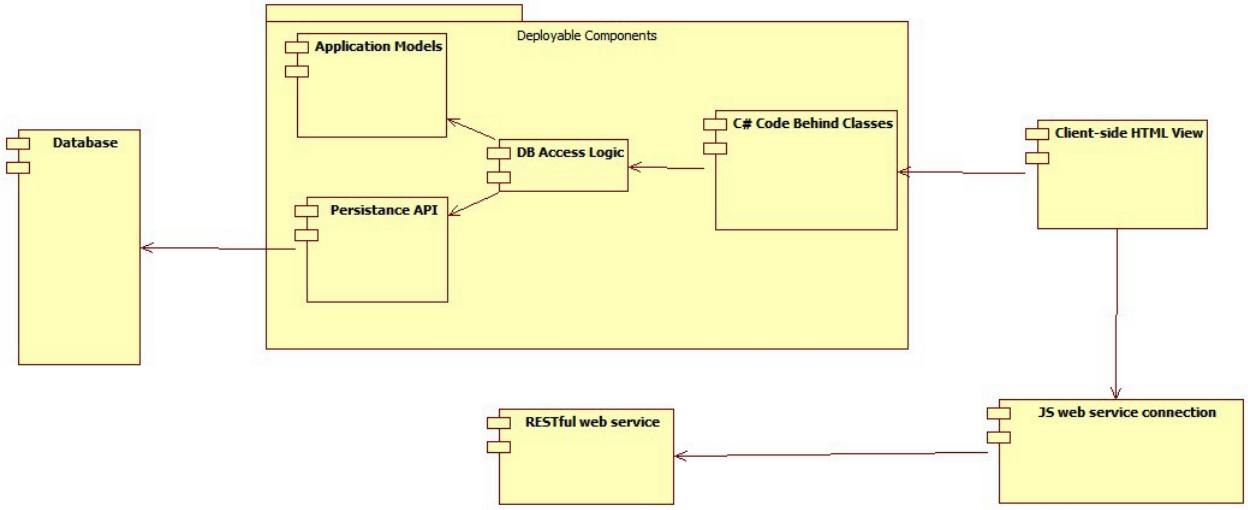
Distributed Systems – assign 2

-Stefan Prisca, group 30441-

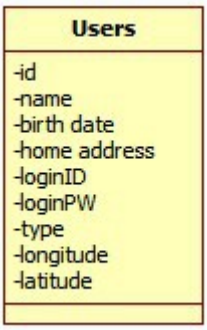
Conceptual Diagram



In the above diagram, you can clearly see all the important parts of this application and how they interect with each other:

- The Database: represents the storage used for the application.
- The Persistence API: represents the API used to connect with the database and to ease data persistence.
  For this application, the Entity framework was chosen as the persistence API. It is easy to configure, and can be used to create code-first databases as well as import existing databases and generate the model classes.
- Application Models: These are in-memory representations of the data models from the database.
- C# Code-Behind classes: are the C# server-side classes that stand behind the HTML views. The application uses ASP.NET WebPages technology, which is based on Micrisoft's "Code-Behind" design pattern.
  Each of this Code-Behind classes contain the data and event handlers that are required for the HTML page to display it's information, and interact with the user.
- RESTful Web Service: this is the web service used for identifying the current location and timezone of the user.
  For this application, Google Maps Timezone API was used. It provides an easy way to determine the location through longitude and latitude.
- RESTful service client connection: this represents the client connection used to communicate with the web service.
  It consists of two javascript functions that run on the client-side.
- Client-side HTML View: contains the web pages that will be displayed in the browser. These were build using ASP.NET WebPages technology.
  Each of these pages has a code-behind server side c# class.

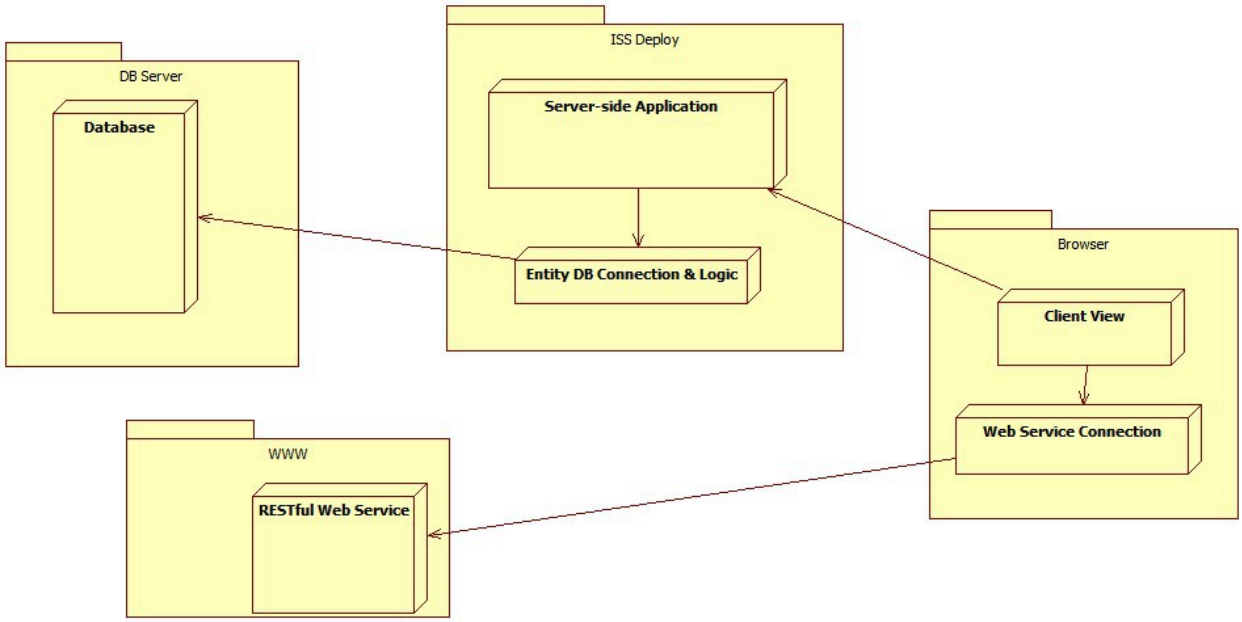
Data Design



The data design is quite simple for this application.
It consists of only one table that contains all the user information that is required for the application to run:

- id: database identifier for the current user
- name: name of the user
- birth date: the birth date of the user
- home address: the home address of the user.
- LoginID: the id the user is using to log in the application
- LoginPW: the password used to log in to the application
- type: the type of the user: administrator or regular user.
- logitude and latitude: world coordinates for the user to determine timezone.

Deployment diagram



The above diagram clearly separates the server-side, client side and web service components of this application.
You can see what runs of each of the three:

1. Server side:
    1. Database will be stored on the server, the application having direct access to it through the Entity framework persistence API.

    2. Server-side application: contains the Entity persistence API, the applicaiton models and the Code Behind classes.
       This is basically the application parts that run on the server.
2. On the WWW: the RESTful web service will reside on the WWW. This is the google maps timezone api.
   The server-side RESTful client will access this service through a *get* request to the server, and will receive the responce in a XML format.
3. Client side: this contains the WebPages HTML pages that will be diplayed on the user browser, as well as the js scripts that will call the external web service.

Opening and Running the application with Visual Studio

Prerequirements:

Besides the packages for ASP.NET WebPages API, the application uses the following NuGet packages:

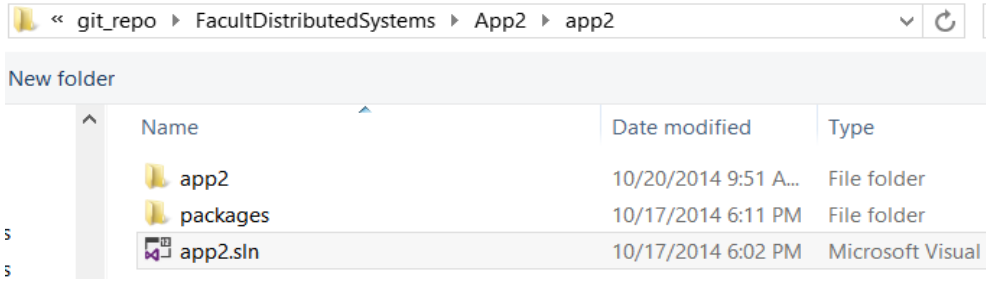
- EntityFramework v6.0.0
- jQuerry v1.10.2
- Bootstrap v3.0.0

You can get this packages using the NuGet Package manager from VS.

It is recommended to use Visual Studio 2013 for this application, with the above-mentioned versions of the packages.

Opening the application

1. Clone the repository on your local machine.
2. From VS 2013:
   1. Go to File -> Open Project
   2. Go to the repository folder (named *FacultDistributedSystems)*, and select the *App2/app2/app2.sln* solution file:



   3. Click on Open.
3. The solution should now be opened in your Visual Studio IDE.

After the application is opened, you'll have to configure the connection string for the db connection. I do not provide my database in the repository.
The best way to do this is to allow Entity to create a db and configure the connection string for you from the Model/APPLICATIONUSER.cs class.
See this tutorial on how to configure code-first databases with Entity framework.

Running the Application

In order to run the application, after you opened it in VS, simply select a browser to open the application in, and click the run button from VS toolbar:

Using the application.

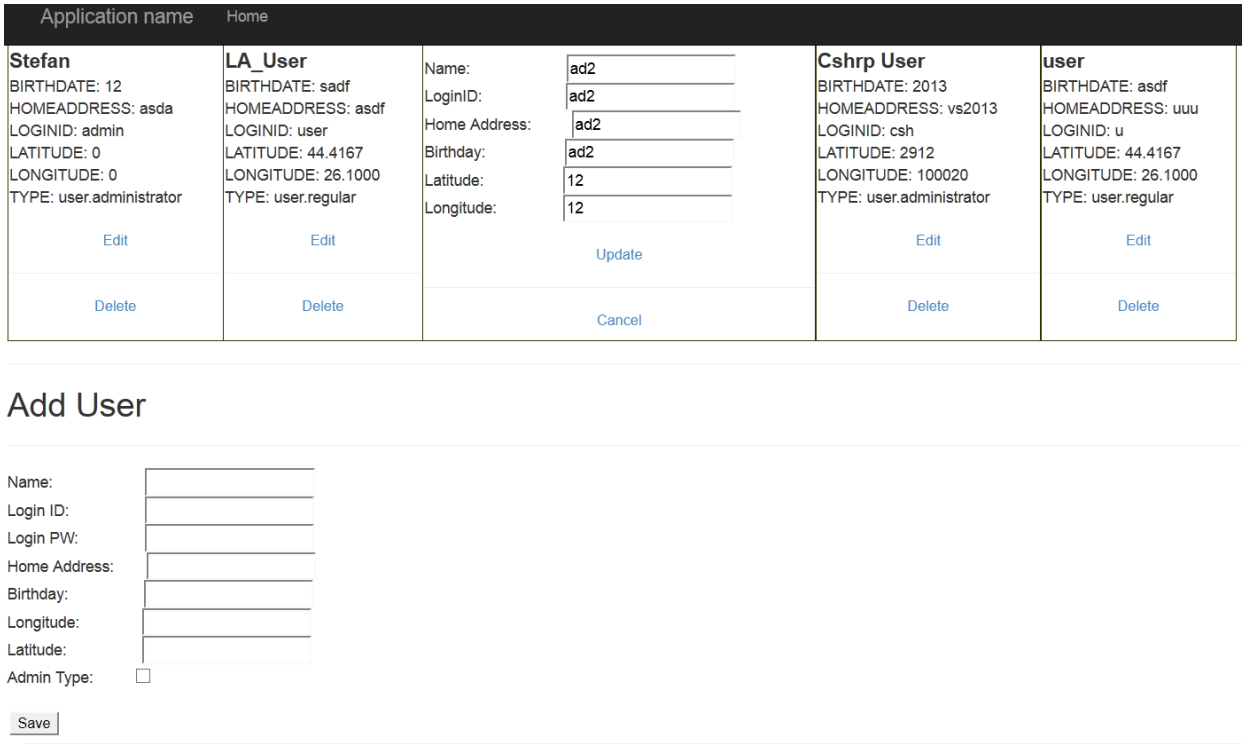
After you start the application using VS, and you access the application through the browser, you'll have to log in.
Note that since there is no DB provided in the repository, you'll have to manually insert some account in the db you create.

I usally use the following accounts for testing purposes:

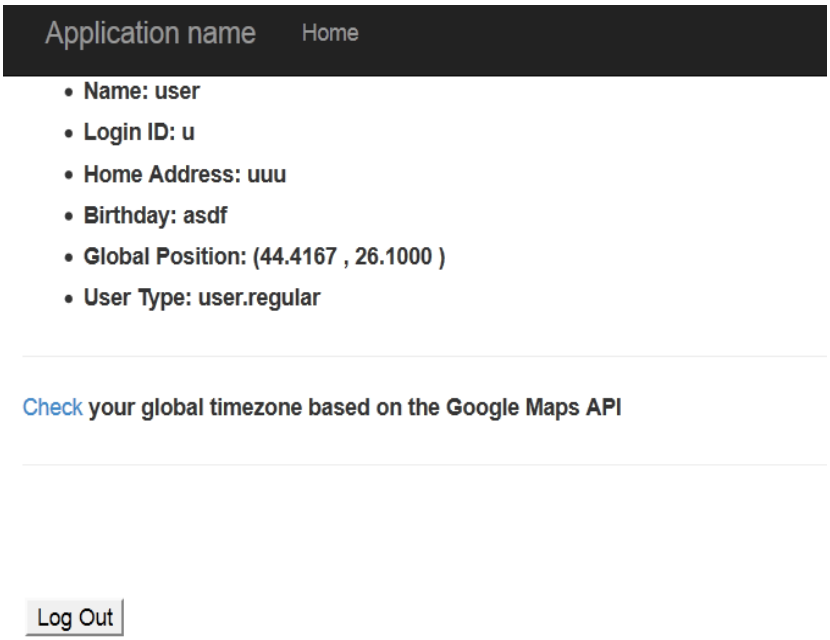
1. administrator account: id = admin; pw = admin
   Use this account to test the administrator functions
2. regular user account: id = user; pw = user
   Use this acount to test the regular user functions.

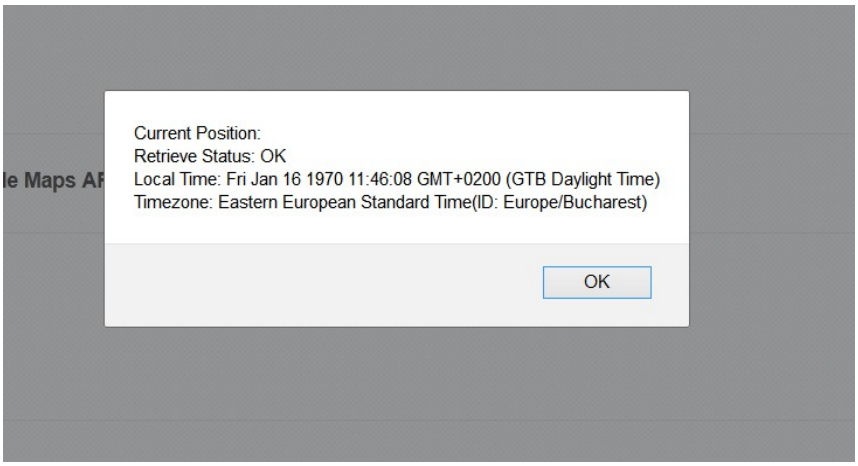
Depending on your log in credentials, you will be redirected to the administrator page or to the regular user page.
Note: If you are not logged in as an administrator, and you try to access the administrator page, you will be redirected to the log in page. Only logged in administrators can access this page!

1. Administrator page:
   Here you will see a table with all the user records from the database.
   You can edit users information, delete users or add aditional users to the data base.
   Note that after you edit user entries, you'll have to press the *Update* button in order to persist the changes in the data base.
   If you edit the longitude and latitude fields, it is highly recommended to use real world coordinates for them. This will allow you to further test the Timezone recognition functionality.



2. Regular user page:
   Here you will see basic information about the regular user.
   The page will display the user record from the database, and the user's current global position and timezone, based on the longitude and latitude fields.





References

- ASP.NET WebPages: <http://www.asp.net/web-pages>
- Entity Framework: <http://msdn.microsoft.com/en-us/data/ef.aspx>
- Visual Studio 2013 Express: <http://www.visualstudio.com/downloads/download-visual-studio-vs>
- Project github repo: <https://github.com/stefanprisca/FacultDistributedSystems>