SSE developer manual

Requirements

Microsoft Visual Studio Community 2015 Version 14.0.25431.01 Update 3
ASP.NET Web Frameworks and Tools 2013  5.2.40314.0
Microsoft Visual C# 2015
ASP.NET and Web Tools 2015.1  14.1.20907.0
Common Azure Tools  1.8
Microsoft Azure Tools for Microsoft Visual Studio 2015 - v2.9.40923.2
SQL Server Data Tools  14.0.60519.0
Microsoft SQL Server Data Tools
Visual Studio Tools for Universal Windows Apps  14.0.25527.01

We recommend installing SQL Express if you intend to use a local database (see database schema) and the UWP SDK if you want to work on the UWP app. You could probably do some things without doing so, but might also run into problems.

We are currently hosting a Web role, web service, Database Server and a Database on the RDN Azure.

Information to access these services are given out by Karl Einarsson (karl@rdn.is).

The latest build connects to these services and requires the developer to have an firewall exception.

These are the tools used for the project and are known to work. Other .NET solutions might work but have not been tested and the UWP project is unlikely to work in another environment.
Sideload the UWP app

Before you can package the UWP app on your machine you must enable developer mode and have the windows 10 UWP SDK installed. If you’ve been supplied with a ready made package you should be able to skip ahead to the end of this chapter to install. Some versions of Windows 10 (1511 and earlier) prompt you when opening the UWP app in visual studio and you see this dialog:

![Developer Mode](image)

On other versions you might have to follow these steps:

1. Go to Settings. Choose Update & security, then choose For developers.
2. Choose Developer Mode.
3. Read the disclaimer for the setting you chose, then click Yes to accept the change.
Once you’ve made sure developer mode has been enabled you can package your app from visual studio.

In Solution Explorer, open the solution for your UWP app project.

Right-click the project and choose Store->Create App Packages.

The Create App Packages wizard appears.

Select No in the first dialog asking if you want to build packages to upload to the Windows Store, then click Next.
Choose the desired output location and architecture.

Now navigate to the output location. If this has been done before you might have more than one version in the location, make sure you select the right one. In the package folder right click the file `Add-AppDevPackage` and run with powershell.

Once powershell has finished the installation you will be able to find the SSEUWP app from the start menu.
**Account Controller**
Contains logic for handling account registration and user login.

**Estimation Controller**
Contains all the logic for Estimation pages, this includes the service calls for creating Estimations, Heads, Lines and uploading and downloading files created from Estimations.

**FAQ Controller**
Routes the user to our FAQ page and Help page

**Home Controller**
Routes the user to homepage containing links to other section of the site

**Manage Controller**
Contains logic allowing the user to change password.

**Team Controller**
Contains the routing logic for Team pages, also the service calls for joining, leaving and creating teams.

**User Controller**
Used for development purposes, routes to a site containing all the users in the system and has logic to change user information.
Database Schema

We use the same model for Estimations, heads and lines, difference is determined by what is in the IsLine property.

An Estimation always has its IsLine property set to 0. Head has a reference to the Estimation it belongs to via Estimation Id and a Line belonging to a specific Head has the Head Id in it’s IsLine property. Therefore we are able to upgrade Lines to Heads and Heads to Estimations by changing the IsLine property.

Description, min, exp and max are taken from the user and the service calculates values for median stddev and Pvalue.

Team table describes a team. It has a unique identifier, name and description.

TeamReg table is used to determine if a user belongs to a specific team. A team can also own Estimations. We are using the default ASP.net user table along with Default MVC user registration.

To use a local database rather than the one hosted on azure while working on either the service or a client you must publish through Visual Studio. Right click on the project inside the solution and select publish, the name of the database should be SSEDb. SQL express is recommended but might not be necessary.

You will have to edit the connection strings accordingly.
Service Descriptions

GET

Returns a list of all registered users.
List<UserDTO> GetAllUsers();

Returns UserDTO object containing ID, Email and username.
UserDTO GetUserDetails(string uid);

Returns TeamDTO list containing all registered teams
List<TeamDTO> GetAllTeams();

Returns single TeamDTO object containing Id, Name, Description and Owner
TeamDTO GetTeamDetails(string uid);

Returns a list of UserDTOs for containing all users in a team
List<UserDTO> GetUsersInTeam(int tid);

Returns a HeadDTO list containing entire database of estimation.
List<HeadDTO> GetAllHeads();

Returns HeadDTO object containing Id, Description, Min, Ex, Max, Median, StdDev, PValue, IsLine, Owner and TeamId
HeadDTO GetHeadDetails(int hid);

Returns a HeadDTO list containing all elements that have a parent
List<HeadDTO> GetAllLines();

Returns a list of Estimations belonging to specific user
List<HeadDTO> GetUsersEstimations(string userId);

Returns a list of heads belonging to a specific estimation
List<HeadDTO> GetEstimationHeads(int estId);

Returns a list of lines belonging to a specific head
List<HeadDTO> GetHeadLines(int headId);

Returns a list of users belonging to a specific team
List<TeamDTO> GetUsersTeams(string uId);

Returns an integer denoting the number of users in the team
int GetNumberOfUsersinTeams(int tid);

POST

Creates a new user
void CreateUser(UserDTO userDTO);

Creates a new team
int CreateTeam(TeamDTO teamDTO);

Adds a user to the team
void AddUserToTeam(int tid, string userId);

Returns true if user is in team
bool IsUserInTeam(int tid, string userId);

Removes the user from the team
void DeleteUserFromTeam(int tid, string userId);

**Creates the initial estimation**
int CreateHead(HeadDTO headDTO);

**Creates element belonging to specific estimation**
void AddLine(int parentID, HeadDTO headDTO);

**Changes database entry for given estimation**
void UpdateHead(HeadDTO headDTO);

**Recursively recalculates and changes status (if needed) of estimations ancestors**
void UpdateParent(HeadDTO headDTO);

**PUT**

**Changes database entry for given user**
void UpdateUser(UserDTO userDTO);

**Changes database entry for given team**
void UpdateTeam(TeamDTO teamDTO);

**DELETE**

**Removes the user from the system**
void DeleteUser(int uid);

**Removes the team from the system**
void DeleteTeam(int uid);

**Removes the Estimation/Head/Line from the system**
void DeleteHead(int hid);