MyCompany Tour – Description and Setup

This document provides a global overview of each application in MY COMPANY, while showing you how to run it, the different functional areas, user credentials, covered technologies, etc.

Contents

[1. Please Read 5](#_Toc371687461)

[2. Requirements 5](#_Toc371687462)

[3. What is MyCompany? 6](#_Toc371687463)

[4. Covered technologies 7](#_Toc371687464)

[Services technologies 7](#_Toc371687465)

[Web Application technologies 7](#_Toc371687466)

[Security technologies 8](#_Toc371687467)

[Windows Apps technologies 8](#_Toc371687468)

[Office 365 technologies 8](#_Toc371687469)

[5. Solutions summary 9](#_Toc371687470)

[Visitors 9](#_Toc371687471)

[Expenses 9](#_Toc371687472)

[Travel 10](#_Toc371687473)

[Staff 10](#_Toc371687474)

[Vacation 10](#_Toc371687475)

[Web project URIs 10](#_Toc371687476)

[6. Test mode 11](#_Toc371687477)

[Web Support 11](#_Toc371687478)

[Windows Store Support 11](#_Toc371687479)

[Desktop Support 11](#_Toc371687480)

[7. Quick Start to run MyCompany from Visual Studio in test mode 12](#_Toc371687481)

[Travel 12](#_Toc371687482)

[Web 12](#_Toc371687483)

[Desktop (WPF) 13](#_Toc371687484)

[Vacation 13](#_Toc371687485)

[Visitor 13](#_Toc371687486)

[Web 13](#_Toc371687487)

[Windows Store 14](#_Toc371687488)

[Expenses 14](#_Toc371687489)

[Web 14](#_Toc371687490)

[Windows Store 14](#_Toc371687491)

[Staff 15](#_Toc371687492)

[Vacation with SharePoint 15](#_Toc371687493)

[Staff with SharePoint 15](#_Toc371687494)

[8. Windows Azure Active Directory 18](#_Toc371687495)

[Create new tenant 18](#_Toc371687496)

[Create demo users 22](#_Toc371687497)

[Update the tenantname in all MyCompany projects 25](#_Toc371687498)

[Travel 25](#_Toc371687499)

[Register WAAD applications 26](#_Toc371687500)

[Travel Web 26](#_Toc371687501)

[Register MyCompany Web Apps 28](#_Toc371687502)

[Register MyCompany Native Apps 29](#_Toc371687503)

[9. Visitors Tour 31](#_Toc371687504)

[Goals 31](#_Toc371687505)

[Technologies used 31](#_Toc371687506)

[Web application 32](#_Toc371687507)

[Windows store application 39](#_Toc371687508)

[Windows Phone Application 46](#_Toc371687509)

[10. Expenses Tour 48](#_Toc371687510)

[Goals 48](#_Toc371687511)

[Technologies being used 48](#_Toc371687512)

[Windows Store Application 49](#_Toc371687513)

[11. Travel Tour 53](#_Toc371687514)

[Goals 53](#_Toc371687515)

[Technologies being used 53](#_Toc371687516)

[Web application 54](#_Toc371687517)

[Desktop application 66](#_Toc371687518)

[12. Staff Tour 69](#_Toc371687519)

[Goals 69](#_Toc371687520)

[Technologies being used 69](#_Toc371687521)

[13. Vacation Tour 80](#_Toc371687522)

[Technologies being used 80](#_Toc371687523)

[14. Deploy MyCompany to local IIS 85](#_Toc371687524)

[Premilinary steps 85](#_Toc371687525)

[Internet Information Server 85](#_Toc371687526)

[SQL Server 2012 85](#_Toc371687527)

[Edit host file 85](#_Toc371687528)

[Emails directory 85](#_Toc371687529)

[Web Deployment Tool 85](#_Toc371687530)

[Travel Web 87](#_Toc371687531)

[Vacation Web 87](#_Toc371687532)

[Visitor Web 87](#_Toc371687533)

[Expenses Web 88](#_Toc371687534)

[Create IIS applications for Web Apps 88](#_Toc371687535)

[Staff 89](#_Toc371687536)

[Visitors Windows Store 93](#_Toc371687537)

[Expenses Windows Store 93](#_Toc371687538)

[Travel Desktop (WPF) 94](#_Toc371687539)

# Please Read

The database with sample data for the demos will be created the first time the application is started, so it´s important to start the application at least once before doing the demo. This operation takes a few minutes.

All the products uses **LocalDB**. LocalDB is created specifically for developers. It is very easy to install and requires no management, yet it offers the same T-SQL language, programming surface and client-side providers as the regular SQL Server Express. In effect, the developers that target SQL Server no longer have to install and manage a full instance of SQL Server Express on their laptops and other development machines.

You can run all the projects in Debug Mode/Any CPU; it´s the default configuration.

Most of the apps in the suite use Windows Azure Active Directory for authentication. This products also could avoid this dependency, for example, for demos where there is not Internet connection. This mechanism is called “**test mode**” where the app works without authentication.

# Requirements

* Windows 8.1 RTM (Recommended: Professional SKU or higher)
* Visual Studio 2013 RTM (Recommended: Ultimate or Premium SKU)
* Windows Azure SDK 2.2 or higher
* Windows Azure subscription
* Office 365 online account and developer site

# What is MyCompany?

My Company is a set of sample applications comprised of typical enterprise/business modules: Travel, Staff, Vacation, Visitors and Expenses.

Each product suite is developed by different teams, and they use different technologies and solutions to fulfill the different requirements.

MyCompany is offered as a Cloud-based solution supporting a single client deployment. All web solutions are deployed to Windows Azure, and use Windows Azure Active Directory to support Single Sign On between all the products in the suite.

The companies can synchronize the directory service with their on-premises directories (Active Directory) to support SSO using the on-premises corporate credentials, or they can integrate the apps with Office 365.

Different technologies are used depending on the application: Web Apps, Windows Store, Windows Phone, WPF, etc. The demo script uses the most appropriate for each scenario.

# Covered technologies

## Services technologies

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Staff | Expenses | Visitors | Travel | Vacation |
| ASP.NET Web API 2 |  | X | X | X | X |
| Attribute Routes |  | X | X | X | X |
| CORS |  | X |  |  |  |
| OWIN-Katana | X | X | X | X | X |
| OWIN self-hosting |  | X |  |  |  |
| Web API OData |  |  |  |  | X |
| LightSwitch OData Services | X |  |  |  |  |
| SignalR 2.0 | X |  | X | X | X |
| Entity Framework 6 |  | X | X | X | X |
| Email notification |  |  |  | X | X |
| Windows Azure Service Bus | X | X | X | X | X |

## Web Application technologies

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Staff | Expenses | Visitors | Travel | Vacation |
| ASP.NET MVC 5 |  |  | X | X | X |
| SPA Web application |  |  | X | X | X |
| Durandal |  |  | X | X | X |
| Knockout.js |  |  | X | X | X |
| Angular.js |  |  |  |  | X |
| ASP.NET Scaffolding |  |  | X |  |  |
| Web Mobile View |  |  |  | X |  |
| JQuery Mobile |  | X |  |  |  |
| Bootstrap |  |  | X |  |  |
| LightSwitch Web Application | X |  |  |  |  |
| Office 365 Cloud Business Application | X |  |  |  |  |
| App for SharePoint |  |  |  |  | X |
| App for Office 365 |  |  |  |  | X |

## Security technologies

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Staff | Expenses | Visitors | Travel | Vacation |
| Windows Azure Active Directory |  | X | X | X | X |
| ASP.NET Identity | X |  | X |  |  |
| SharePoint integrated security | X |  |  |  | X |

## Windows Apps technologies

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Staff | Expenses | Visitors | Travel | Vacation |
| Windows 8.1 Store C#/XAML |  | X | X |  |  |
| Windows 8.1 Store HTML/JS |  | X |  |  |  |
| Windows 8.1 Push notifications |  | X |  |  |  |
| NFC communication |  |  | X |  |  |
| Windows Phone C#/XAML |  | X |  |  |  |
| Windows Phone Push notifications |  | X |  |  |  |
| .NET WPF |  |  |  | X |  |

## Office 365 technologies

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Staff | Expenses | Visitors | Travel | Vacation |
| Office 365 Cloud Business Application (aka. LightSwitch) | X |  |  |  |  |
| App for SharePoint – ASP.NET MVC |  |  |  |  | X |
| App for SharePoint – HTML/JS | X |  |  |  |  |
| App for SharePoint – BCS | X |  |  |  |  |
| App for Office 365 – Mail app |  |  |  |  | X |

# Solutions summary

## Visitors

All the projects and clients are inside the **MyCompany.Visitors.sln** solution.

|  |  |
| --- | --- |
| **MyCompany.Visitors.Web** | Main Web app with ASP.NET MVC 5 and ASP.NET Web API 2 |
| **MyCompany.Visitors.Web.Scaffolding** | Web app that uses ASP.NET Identity, BootStrap and ASP.NET Scaffolding |
| **MyCompany.Visitors.Client.WindowsStore** | Windows 8.1 Store app - C#/XAML |
| **MyCompany.Visitors.Client.WP** | Windows Phone app – C#/XAML – with NFC communication with Windows 8.1 Store App |

This folder also contains a MyCompany.Visitors.Client.WindowsStore.sln that loads only the Windows Store application, to open it with Blend.

## Expenses

|  |  |
| --- | --- |
| **MyCompany.Expenses.sln** | Main solution with all existing project |

**MyCompany.Expenses.sln** solution:

|  |  |
| --- | --- |
| **MyCompany.Expenses.Web** | Main ASP.NET Web API app |
| **MyCompany.Expenses.WebApiOwin** | ASP.NET Web API using OWIN self-hosting |
| **MyCompany.Expenses.Web.Mobile** | ASP.NET MVC using Visual Studio Mobile template (JQueryMobile) |
| **MyCompany.Expenses.Client.WindowsStore** | Windows 8.1 Store App – HTML/JS |
| **MyCompany.Expenses.Client.WP** | Windows Phone App – C#/XAML |

## Travel

All the projects and clients are inside the **MyCompany.Travel.sln** solution.

|  |  |
| --- | --- |
| **MyCompany.Travel.Web** | Main Web app with ASP.NET MVC 5 and ASP.NET Web API 2 |
| **MyCompany.Travel.Client.Desktop** | .NET WPF App |

This folder also contains a MyCompany.Travel.Client.Desktop.sln that loads only the WPF application, to open it with Blend.

## Staff

|  |  |
| --- | --- |
| **MyCompany.StaffAlone.sln** | LightSwitch project with ASP.NET Identity |
| **MyCompany.Staff.sln** | Office 365 Cloud Business Application (aka. LightSwitch App for SharePoint) |
| **MyCompany.Staff.Demo.sln** | Office 365 Cloud Business Application without some features that will be done during the demoscripts |
| **MyCompany.Staff.Web.SharePointODataECT.sln** | App for SharePoint that consume external OData Service and showing it in a regular SharePoint list |
| **MyCompanyStaffIssues.sln** | App for SharePoint (HTML/JS) that reads a SharePoint list. This project was done with Napa |

## Vacation

|  |  |
| --- | --- |
| **MyCompany.Vacation.sln** | ASP.NET MVC 5 and ASP.NET Web OData; SPA, Angular.js, OData. |
| **MyCompany.Vacation.SharePoint.sln** | App for SharePoint - ASP.NET MVC 5 and Web API 2; WAAD, SPA, Durandal, Knockout.js. |
| **MyCompany.Vacation.MailOfficeApp.sln** | App for Office 365; Mail app, OAuth. |
| **MyCompany.Vacation.MailOfficeAppBasic.sln** | App for Office 365; Mail app without security for demos |
| **MyCompany.Vacation.NoAuth.sln** | ASP.NET MVC 5 app without WAAD security for demos |

## Web project URIs

|  |  |
| --- | --- |
| MyCompany.Visitors.Web | <http://localhost:31330/> |
| MyCompany.Travel.Web | <http://localhost:31332/> |
| MyCompany.Expenses.Web | <http://localhost:31329/> |
| MyCompany.Expenses.Web.Mobile | <http://localhost:31333/> |
| MyCompany.Expenses.WebApiOwin | <http://localhost:31345/> |
| MyCompany.Vacation.sln | MyCompany.Vacation.Web | <http://localhost:31331/> |
| MyCompany.Vacation.SharePoint.sln | MyCompany.Vacation.Web | <http://localhost:31340/> |

# Test mode

Most of the apps use Windows Azure Active Directory for authentication. The app can avoid this dependency, for example, for demos where there is not Internet connection or not to configure WAAD. This mechanism is called “**test mode**” where the apps works without authentication.

If the test mode is activated the application will use the user Andrew Davis by default.

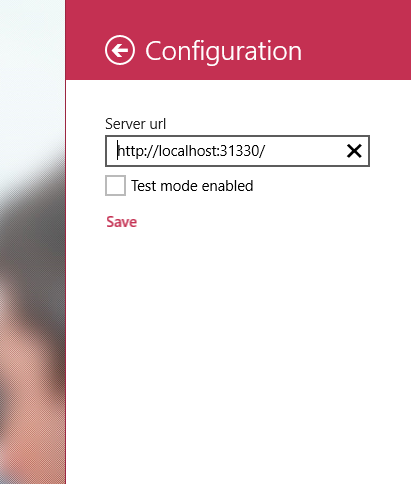
## Web Support

Adding the “NoAuth” word at the end of the URI the application will not ask for authentication.

* Travel => <http://mycompanyserver/travel/noauth>
* Visitors => <http://mycompanyserver/visitors/noauth>
* Vacation => <http://mycompanyserver/vacation/noauth>

## Windows Store Support

In the setting page of the windows Store Apps (Expenses and Visitors) it´s possible to activate the test mode. The application remembers these settings so the next time the application will start in test mode.



## Desktop Support

To activate the mode in the Desktop application (MyCompany.Travel.Client.Desktop) you need to change the test mode application setting inside the configuration file (app.config). Set this property to true to activate the test mode.

# Quick Start to run MyCompany from Visual Studio in test mode

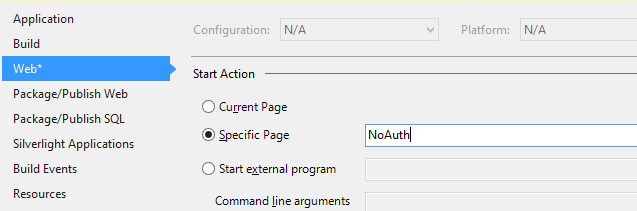
This steps shows how to rum MyCompany project without configure Windows Azure Active Directory, using fake users.

This steps only show how to run the main applications.

## Travel

### Web

1. Open solution MyCompany.Travel.sln
2. Select MyCompany.Travel.Web
3. Open the Web project and set NoAuth as specific page in the start action.



1. By running this project with F5, it´s possible to debug it. The first time it is run, the database is created.
2. The Web App must be open at <http://localhost:31332/>noauth
3. Click on Publish

### Desktop (WPF)

1. Open the MyCompany.Travel.sln solution.
2. Select MyCompany.Travel.Web and start it without debugging.
3. Select MyCompany.Travel.Client.Desktop.
4. Open the app.config file to check the configuration.
   1. The ServiceUrlBase property must be <http://localhost:31332/>
   2. The TestMode property must be “true”
5. Run the project with F5.

## Vacation

1. Open solution MyCompany.Vacation.sln
2. Select MyCompany.Vacation.Web
3. Open the Web project and set NoAuth as specific page in the start action.
4. By running this project with F5 it´s possible to debug it. The first time it is run, the database will be created.
5. The Web App must be open at <http://localhost:31331/noauth>

## Visitor

### Web

1. Open solution MyCompany.Visitors.sln
2. Select MyCompany.Visitors.Web
3. Open the Web project and set NoAuth as specific page in the start action.
4. By running the project with F5, it´s possible to debug it. The first time it is run, the database is created.
5. The Web App must be open at <http://localhost:31330/>noauth

### Windows Store

1. Open the MyCompany.Visitors.sln solution.
2. Select MyCompany. Visitors.Web and start it without debugging.
3. Select MyCompany.Visitors.Client.WindowsStore.
4. Open the file Settings | AppSettings.cs
   1. The ApiUri property must be <http://localhost:31330/>
5. Run the project with F5.

## Expenses

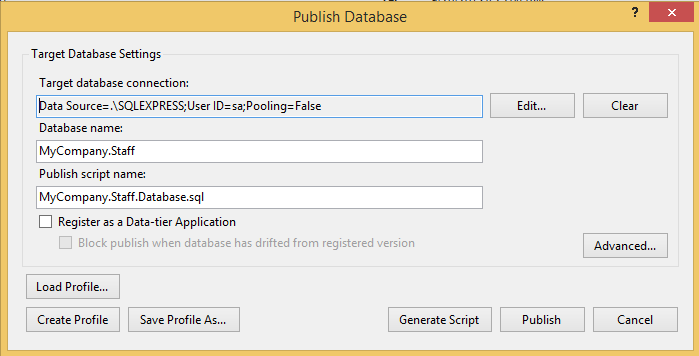
### Web

1. Open the MyCompany.Expenses.sln solution
2. Select MyCompany. Expenses.Web
3. Open the Web project and set NoAuth as specific page in the start action.
4. By running this project with F5, it´s possible to debug it. The first time it is run, the database is created.
5. The Web App must be open at <http://localhost:31329/>noauth.
   1. Note that it’s an API site, so the default page will be blank.

### Windows Store

1. Open the MyCompany.Expenses.sln solution
2. Select MyCompany. Expenses.Web
3. Start it without debugging.
4. Select MyCompany. Expenses.Client.WindowsStore.
5. Open the file js | config.js
   1. The ServiceUrlBase property must be <http://localhost:31329/>
6. Run the windows store project with F5.

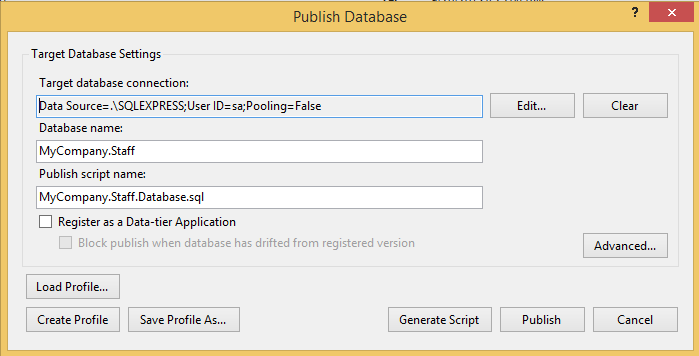
## Staff

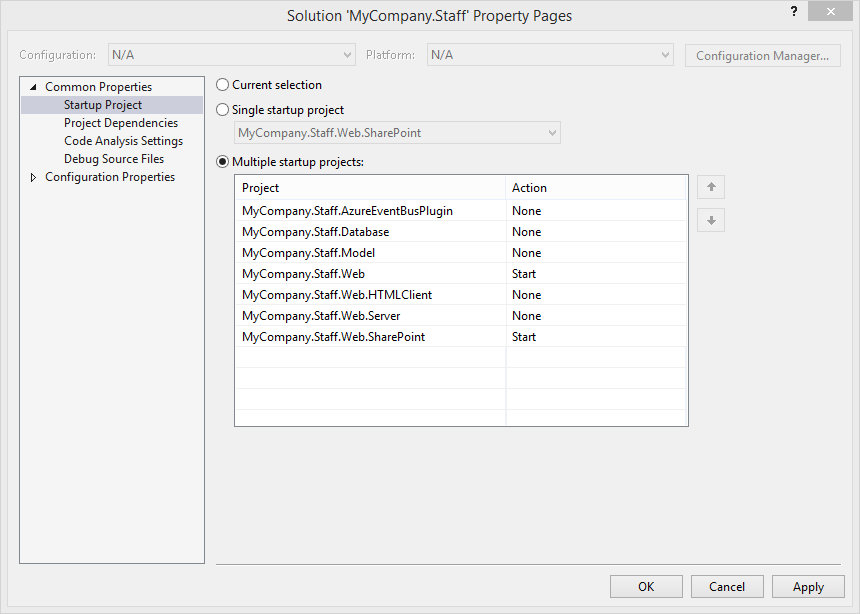
1. Open the MyCompany.StaffAlone.sln solution.
2. Right-click on MyCompany.Staff.Database project and click on Publish. (MyCompany.Staff.Database.publish.xml)
3. Click on the Edit button in Target Database Connection
4. Load the profile that is inside the Database project to deploy to LocalDB.  
   
5. Click on Publish. The database will be published in your localDB.
6. Right-click on MyCompany.Staff.Web project and run it with F5.

## Vacation with SharePoint

1. Open solution MyCompany.Vacation.SharePoint.sln
2. In the Solution Explorer, select the project MyCompany. Vacation.SharePoint.
3. Press F4.
4. Set the Site URL to your O365 dev portal.
5. Right-click on the solution in the Solution Explorer and select “Set StartUp projects…”
6. Change to Action->Start the projects MyCompany. Vacation.Web and MyCompany. Vacation.SharePoint.
7. By running this project with F5 it´s possible to debug it.

## Staff with SharePoint

1. Open the MyCompany.Staff.sln solution.
2. Right-click on MyCompany.Staff.Database project and click on Publish.
3. Click on the Edit button in Target Database Connection
4. Load the profile that is inside the Database project to deploy to LocalDB.  
   
5. Click on Publish. The database will be published in your localDB.
6. In the Solution Explorer, select the project MyCompany.Staff.Web.SharePoint.
7. Press F4.
8. Set the Site URL to your O365 dev portal.
9. Edit the web.config file of MyCompany.Staff.Web.Server project to change the connectionStrings to your O365 dev portal.
10. Right-click on the solution in the Solution Explorer and select “Set StartUp projects…”
11. Change to Action->Start the projects MyCompany.Staff.Web and MyCompany.Staff.Web.SharePoint.
12. F5.



# Windows Azure Active Directory

Windows Azure Active Directory is a service that provides identity and access management capabilities in the cloud. In much the same way that Active Directory is a service made available to customers through the Windows Server operating system for on-premises identity management, Windows Azure Active Directory (Windows Azure AD) is a service that is made available through Windows Azure for cloud-based identity management.

Because it is your organization’s cloud directory, you decide who your users are, what information to keep in the cloud, who can use the information or manage it, and what applications or services are allowed to access that information.

When you use Windows Azure AD, it is Microsoft’s responsibility to keep Active Directory running in the cloud with high scale, high availability, and integrated disaster recovery, while fully respecting your requirements for the privacy and security of your organization’s information.

This point describes the steps that you need to complete to configure MyCompany product to use Windows Azure Active Directory. These steps are not necessary is you only use the test mode described in the previous points.

## Create new tenant

|  |  |  |
| --- | --- | --- |
| Demo Screen | Steps | Script |
|  | 1. Open Internet Explorer, and navigate to the Windows Azure portal developer site https://manage.windowsazure.com/ 2. Click on Active Directory 3. Click on Directory 4. Click on Add directory or the Add button on the tool bar at the bottom. | Windows Azure AD is a multi-tenant service. This means that it can simultaneously support many different organizations, storing directory information about users in each of them.  The tenants are applications rather than customer organizations. |
|  | 1. Click On “Add an application my organization is developing” | Windows Azure Active Directory is the service that provides the identity backbone of Microsoft offerings such as Office 365 and Intune. If you subscribe to any of those services, you already have a Windows Azure AD tenant available for your users to sign in. |
|  | 1. Click on “Create new directory” | If you don´t have an existing tenant, the Windows Azure Management Portal provides a mechanism to create a new tenant directly from the Management Portal’s pages. |
|  | 1. Fill all the information required | The Organization Name field is required, and its value will be used as a moniker whenever there’s the need to display the company name.  For Country or Region, the value selected in this dropdown will determine where your tenant will be created. Given that the directory will store sensitive information, please do take into account the privacy laws of the country in which your company operates.  The Domain Name field represents a critical piece of information. It is the part of the directory tenant domain name that is specific to your tenant—what distinguishes it from every other directory tenant.  At creation, every directory tenant is identified by a domain in the form [tenantname].onmicrosoft.com. That domain is used in the UPN of all the directory users and in general wherever it is necessary to identify your directory tenant. After creation, you can register additional domains that you own.  The Domain Name must be unique, and the UI validation logic will help you to pick a unique value.  It is recommended that you choose a handle that refers to your company, as it will help users and partners as they interact with the directory tenant. |
|  | 1. The directory will be created 2. Click on the directory name to go to his details | Filling out the dialog is all it takes to create a directory tenant. As soon as I click the check button on the lower right corner, Windows Azure AD creates a new tenant for you according to the parameters specified. |

## Create demo users

You need to create three demo users following these steps:

* [Andrew.Davis@[tenantname].onmicrosoft.com](mailto:Andrew.Davis@[tenantname].onmicrosoft.com)
* [Carole.Poland@[tenantname].onmicrosoft.com](mailto:Carole.Poland@[tenantname].onmicrosoft.com)
* [Thomas.Andersen@[tenantname].onmicrosoft.com](mailto:Thomas.Andersen@[tenantname].onmicrosoft.com)

|  |  |  |
| --- | --- | --- |
|  | 1. Click on Users 2. Click on Add User 3. Select “New user in your organization” 4. Write the user name 5. Click next | The directory tenant is initially empty, except for the Microsoft Account administering the Windows Azure subscription in which the new tenant was created.  The Microsoft Account is listed here to signal that it has Global Administrator privileges for the tenant. However, that is true only for operations performed through the Windows Azure Management Portal UI. The Microsoft Account cannot actually authenticate against the directory tenant, such as for web SSO, hence it cannot be used as a test user for our web SSO walkthrough.  Let’s add a new user to the directory, so that I will be able to exercise the web SSO scenario later in the demo. |
|  | 1. Write the personal info (first and last name and display name) 2. Select “Global Administrator” as Role 3. Write an alternate email address 4. Click next. | The Add User dialog begins by asking if you want to create a directory user or if you want to add an existing Microsoft Account (which will have the same limitations as the Microsoft Account currently in use for administering the subscription). We need a directory user for our workflow, so let’s go ahead and pick the New user in your organization entry.  The management Portal generates a temporary password, which will have to be used for the first login. At that time, the user will be required to change the password. Please save the temporary password somewhere, as we’ll need it once we have all the components in place to test the scenario. |
|  | 1. Click on create to display the password and copy it, you’ll need it later. |  |
|  | 1. Show that the user has been created |  |

## Update the tenantname in all MyCompany projects

### Travel

|  |  |  |
| --- | --- | --- |
|  | 1. Open solution MyCompany.Travel.sln 2. Open Find and Replace 3. Add **[tenantname]** in the first field 4. In the second field, add the tenant that you have created in the previous step 5. Open the Web project and set NoAuth as specific page in the start action. 6. F5. | By default, all project has the key **[tenantname]** in the tenant name so you must change it to use you new tenant.  After changing the tenant name you must run the application. This operation will create the database using your tenant.  If the database already exists, you must delete it before. |

You must complete these steps with all the solutions where you want to use Windows Azure Active Directory:

* MyCompany.Travel.sln
* MyCompany.Expenses.sln
* MyCompany.Visitors.sln
* MyCompany.Vacation.sln

Even though you configure WAAD, the test mode always is available.

## Register WAAD applications

### Travel Web

This steps describes how to register the travel web application.

|  |  |  |
| --- | --- | --- |
|  | 1. Go to the **Active Directory** tab in the Windows Azure Management Portal 2. Open your tenant configuration 3. Click on **Integrated Apps** 4. Click on Add | This area is dedicated to listing all the applications that are registered in your directory tenant.  No application can take advantage of Windows Azure AD without having been registered: this is both for security reasons (only apps that the administrator approves of should be allowed) and practical considerations (interaction with Windows Azure AD entails use of specific open protocols, which in turn require the knowledge of key parameters describing the app).  For this reason you must register all the MyCompany applications in your new tenant. |
|  | 1. Click on **Add an application myorganization is developing** |  |
|  | 1. Click on **Web Application** 2. Set the name MyCompany.Travel.Local (or the name that you prefer) |  |
|  | 1. Add the <http://localhost:31332/> as App URL and ID. | [**http://localhost:31332/**](http://localhost:31332/) is the URI where MyCompany.Travel.Web run locally. |
|  | 1. Now, Follow the MyCompany Tour section to test the application |  |

### Register MyCompany Web Apps

Now, you must repeat the previous steps to register all the web applications that use Windows Azure Directory.

|  |  |
| --- | --- |
| **App Name** | **URI** |
| MyCompany.Visitors.Local | <http://localhost:31330/> |
| MyCompany.Travel.Local | <http://localhost:31332/> |
| MyCompany.Expenses.Local | <http://localhost:31329/> |
| MyCompany.Vacation.Local | <http://localhost:31331/> |

### Register MyCompany Native Apps

The registration process for native apps is a bit difference.

|  |  |
| --- | --- |
| **App Name** | **Redirect URIs** |
| MyCompany.Visitors.Native | <http://localhost:31330/> |
| MyCompany.Travel.Native | http://localhost:31332/ |
| MyCompany.Expenses.Native | <http://localhost:31329/> |

This steps show how to create the native application for Travel Desktop:

|  |  |  |
| --- | --- | --- |
|  | 1. Click on **Add an application myorganization is developing** |  |
|  | 1. Click on **Native Application** 2. Set the name MyCompany.Travel.Native (or the name that you prefer) |  |
|  | 1. Add the <http://localhost:31332/> as Redirect URL. |  |
|  | 1. Go to the new tenant details 2. **In the web apis section**, add MyCompany.Travel.Local | This action is needed to allow the connection to the Web API from the native application. |
|  | 1. Now, get the ClientId of your new aplication 2. Open MyCompany.Travel.sln 3. Select MyCompany.Travel.Client.Desktop 4. Open the app.config 5. Update the setting **ServiceClientId** 6. Update the testmode to false | You can search for the key [ClientId] to replace it using this value |
|  | 1. Follow the MyCompany Tour section to test the applicatoin |  |

# Visitors Tour

Visitors is composed of several client applications that allows you to request and manage visits and visitors for your company. Different features are available depending on your role in the organization, and the client application being used (Web app, Windows Store app or Phone app).

The web application is an application for employees to schedule visits and manage visitor information. The Windows Phone application presents a visitor-card to be filled with the visitor information and simplifies the process of handling a visitor’s arrival through NFC communication. The Windows 8 application is an application that targets security staff, so they can manage visits and visitor information from a security staff member’s point of view.

Email notifications are sent in some circumstances, and there are also web notifications using SignalR.

This tour only show how the main features.

### Goals

Demo support to highlight Scaffolding, ASP.NET Identity, Bootstrap, NFC communication and XAML/C# Store apps.

### Technologies used

* Server logic exposed using ASP.NET Web API 2.0
* ASP.NET MVC 5 SPA developed with Durandal, jQuery and Knockout.js
* ASP.NET Scaffolding
* Entity Framework 6 Code-first
* Route prefixes
* Crop images
* LocalDb for local development environment
* Server components deployed on Windows Azure Web Sites and SQL Azure
* NFC communication
* MVVM
* Live tiles
* Portable libraries

## Web application

|  |  |  |
| --- | --- | --- |
| Demo Screen | Steps | Script |
|  | 1. Open the **MyCompany.Visitors.sln** solution from the **Visitors** folder. | We are going to present the Visitor web application.  It’s an application to schedule and manage employee visits to our company. |
|  | 1. In the solution explorer window right-click in the **MyCompany.Visitors.Web** project and select **Set as Startup** **Project**. 2. Run the application by pressing **F5**. 3. Login. User: **Andrew.Davis@[tenantname].onmicrosoft.com** Password: **[password]** | Let’s start the application  If WAAD is not configure, use the test mode adding the “NoAuth” word in the “Start Action” (Specific Page) of the web project. |
|  | 1. When the page is loaded, click the Visitors tab | Here you can see the visitors view. It lists all the visits to our company, including the date and time, and the person visiting.  Switching to the Visitors tab, we can see a list of the visitors we’ve had to our company. |
|  | 1. Click on the Open Detail view button | Let’s take a look at the details view for one of the visitors. |
|  | 1. Close the dialog. 2. Click on the edit visitor button. | Here, you can see a profile of visitor, including when they last visited.  We can edit the visitor’s details to update their information. |
|  | 1. Close the dialog. 2. Click on New Visitor button. | Let’s see how to create a new visitor.  Clicking on the New Visitor button will open a form, including all the relevant fields. |
|  | 1. Fill the form. 2. Press Add image button. | Once I’ve filled in their details, I can add a photograph. |
|  | 1. Select an image. 2. Press Next button. |  |
|  | 1. Resize and move the selection. 2. Click Finish button. 3. Close the dialog. | It can be useful to crop and reframe the image, so let’s do that.  I’m cropping the image to focus on our visitor’s face.  There, that looks good. Once we click finish, the image will be uploaded. |
|  | 1. Click on Visits tab. 2. Click on the view visit info button. | Going back the Visits tab, we can see all the scheduled visits.  Clicking on the clock icon will show the visit information. |
|  | 1. Close the dialog. | Here we see details of the visit, including who is meeting who.  Note that we can call employee who is the subject of the visit using Lync by clicking on the Lync icon. |
|  | 1. Click on edit visit button. 2. Close the dialog. 3. Close the application. | We may want to reschedule a visit or change who will be visiting. This can be done by clicking on the pen icon. |

## Windows store application

|  |  |  |
| --- | --- | --- |
| Demo Screen | Steps | Script |
|  | 1. Open the **MyCompany.Visitors.sln** solution from the **Visitors** folder. | As well as a web app, there is also a Windows Store version of the Visitors app. |
|  | 1. In the solution explorer window right-click in the **MyCompany.Visitors.Web** project and select **Set as Startup** **Project**. 2. Run the application by pressing **Ctrl +** **F5**. | Let’s start up the project.  If WAAD is not configure, use the test mode adding the “NoAuth” word in the “Start Action” (Specific Page) of the web project. |
|  | 1. In the solution explorer window right-click in the **MyCompany.Visitors.Client.WindowsStore** 2. Click on **Debug -> Start new instance** 3. **Login**. User: [**Andrew.Davis@[tenantname].onmicrosoft.com**](mailto:Andrew.Davis@mycompanyapps.onmicrosoft.com)Password: **[password]** | If WAAD is not configure, use the test mode that we have seen in the previous point. |
|  | 1. Take a **look** at the new **Hub** control. | Here you can see the new Hub control available to Windows Store apps.  It presents the same information as the web app in a manor typical of a Windows Store app. |
|  | 1. **Click** on any visit on **other visits** section. | Here we can see the details of a visit. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Visitors\StoreApp\27-09-2013 12-16-13.png | 1. **Click** on the **Lync** icon in the visit. 2. **Close** the **application** and go back. | Let’s call the employee via Lync to confirm some details about the visit.  As you can see, Lync opens and handles the call. |
|  | 1. **Click** on the today visits title. 2. **Go** back. | Now, let’s check who is visiting our company today. |
|  | 1. **Right-click** anywhere on the screen. 2. **Click** on Add button. | Looks like no one has scheduled the visit I’d planned for today.  We can add a new visit by right clicking and pressing the Add button. |
|  | 1. **Fill** in the information. | I’m filling out the visit details… |
|  | 1. **Click** on Add a visitor button. | And then I can add the attendees. |
|  | 1. **Right-click** anywhere on the screen. | Looks like one of the attendees is not in our system. |
|  | 1. **Click** Add button. 2. **Fill** in the information. 3. **Click** Add an image. | Let’s add a new visitor, by right clicking and pressing the Add button. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Visitors\StoreApp\27-09-2013 13-41-05.png | 1. **Click** on image. | We need an image, so I’ll take one now using the web cam integration. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Visitors\StoreApp\27-09-2013 14-25-18.png | 1. **Resize** and **move** the crop selector. 2. **Click** on **Ok** button. | As before, we can crop the image.  Perfect. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Visitors\StoreApp\27-09-2013 14-25-33.png | 1. **Resize** and **move** the crop selector. 2. **Click** on **Confirm**. | We can also crop the image for the thumbnail image. |
|  | 1. **Fill** in the information. |  |
|  | 1. **Right-click** anywhere on the screen. 2. **Click** on **save** button. | Now, let’s save our newly created user by bringing up the menu bar and pressing the Save button. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Visitors\StoreApp\27-09-2013 12-15-40.png | 1. **Open** the charms bar. 2. **Click** on settings. 3. **Click** on configuration. 4. **Close** the application. | The app can be configured by pressing Setting in the Charms bar, and clicking Configuration. |

## Windows Phone Application

|  |  |  |
| --- | --- | --- |
| Demo Screen | Steps | Script |
|  | 1. Open the **MyCompany.Visitors.sln** solution from the **Visitors** folder. | Now, we can take a quick look at the Windows Phone version of our Visitors app. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Visitors\WindowsPhone\27-09-2013 13-36-41.png | 1. In the **Solution Explorer** right click on **MyCompany.Visitors.Client.WP** and click on Set as StartUp Project. 2. Press **F5** to run the application. | I’ll just start it up… |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Visitors\WindowsPhone\27-09-2013 12-57-52.png | 1. **Fill** in the form. 2. **Click** on Add image. | Here, we can add a visitor. |
|  | 1. **Select** an image from the gallery or click on the photo button to take a new picture. | We can either add an existing image or take a selfie with the phone’s camera. |

# Expenses Tour

*Expenses* is composed of several client applications to facilitate expense management for the employees.

The *web mobile* application is an application for employees, managers and HR. Each role can perform different actions. Employees can create and edit their travel requests. Managers can accept or deny a team member’s request. HR can add attachments to requests and accept them.

The *Windows 8 HTML5* *native* application targets managers, so that they can review the team’s expenses, approve or deny them, and view a summary.

The *Windows Phone 8 native application* targets both employees and managers. The employees can report their expenses and managers can review/approve team expenses.

This tour only show how the main features.

### Goals

Demo support to highlight *Web API* features like *attribute routes*, *CORS*, and *OWIN self-hosting*. Also, *HTML/WinJS native store apps*, and *Web Mobile apps* for any mobile client.

### Technologies being used

* Server logic exposed using ASP.NET Web API 2.0
* Entity Framework 6 Code-First
* LocalDb for local development environment
* Server components deployed on Windows Azure Web Sites and SQL Azure
* CORS. Web and Web API hosted on different domains.
* OWIN. Web API hosted on a Windows Azure Worker Role.
* Attribute Route prefixes.
* Web Mobile.
* jQuery Mobile.
* Promises.
* Snapped Views.
* Live tiles.
* Push notifications.
* Portable libraries.
* MVVM

## Windows Store Application

|  |  |  |
| --- | --- | --- |
| Demo Screen | Steps | Script |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Expenses\Open solution.png | 1. Open the **MyCompany.Expenses.sln** solution from the **Expenses** folder. | Now I’m going to show the Expenses Windows Store application from our app suite.  This is an application to manage our company expenses. |
|  | 1. In the solution explorer window right-click in the **MyCompany.Expenses.Web** project and select **Set as Startup** **Project**. 2. Run the application by pressing **Ctrl +** **F5**. | If WAAD is not configure, use the test mode that we have seen in the previous point. |
|  | 1. In the solution explorer window right-click in the **MyCompany.Expenses.Client.WindowsStore** 2. Click on **Debug -> Start new instance** 3. **Login**. User: [Andrew.Davis@[tenantname].onmicrosoft.com](mailto:Andrew.Davis@mycompanyapps.onmicrosoft.com) Password: [password] | If WAAD is not configure, use the test mode that we have seen in the previous point. |
|  | 1. Navigate through the **window**. | Here we can see the typical Hub view used by Window Store applications. |
|  | 1. Click on an **expense** from the **Team expenses** section. | Here is an expense request made by one of the employees. |
|  | 1. Right-click **anywhere** on the screen. | By bringing up the app bar, we can approve or deny the request. |
|  | 1. Go **back**. 2. Click on an item from **Total expenses** per employee section. 3. Click on the different expense **type buttons** to view the different values. | Now, let’s look at one of the expense items.  We can look at the different expense types to see their individual values. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Expenses\windows store\27-09-2013 13-11-17.png | 1. Open the **charms** bar. 2. Click on **settings**. 3. Click on **configuration**. 4. Close the **application**. | As before, we can change the configuration from the Charms bar. |

# Travel Tour

Travel is made up of several applications that allow employees to request and manage travel.

The web application targets any type of employee, including managers and HR. Each role can perform different actions. Regular employees can create and edit their own travel requests. Managers can accept or deny team member requests. HR can add attachments to the requests and accept them.

The desktop application (WPF) targets HR, so they can edit and process travel requests.

In both applications email notifications are sent in some circumstances, and there are also web notifications using SignalR.

This tour only show how the main features.

### Goals

To highlight the importance of desktop clients (WPF) for certain scenarios. It shows new technologies in WPF, like SignalR client lib for .NET and Toast notifications

### Technologies being used

* Server logic exposed using ASP.NET Web API 2.0
* Entity Framework 6 Code-First
* LocalDb for local development environment
* Server components deployed on Windows Azure Web Sites + SQL Azure
* Possibility of being integrated as a provider-hosted application in SharePoint
* SPA developed with Durandal, jQuery and Knockout.js
* Different views for mobile devices
* Bing Maps integration
* WPF

## Web application

|  |  |  |
| --- | --- | --- |
| Demo Screen | Steps | Script |
|  | 1. Open the **MyCompany.Travel.sln** solution from the **Travel** folder. | I’m now going to use the Travel web app from our app suite.  This app allows us to schedule and manage employee travel in our company. |
|  | 1. In the solution explorer window right-click in the **MyCompany.Travel.Web** project and select **Set as StartUp** **Project**. 2. Run the application by pressing **F5**. | If WAAD is not configure, use the test mode that we have seen in the previous point. |
|  | 1. Log in. User: **Carole.Poland@[tenantname].onmicrosoft.com** Password: **[password]** | I’m logging in as Carole Poland. Sge is not a manager nor in HR, so she can only manage her own travel.  Here we can see a list of all her upcoming trips. |
|  | 1. Click on **New Travel** button to create a new travel. 2. **Fill** in the form. 3. Click on **Accept Changes** button | I’m going to make a new travel request for Carole…Let’s send her to San Francisco.  Now that I’ve filled out the details, you will see the new travel request in the list. |
|  | 1. Open a new Internet Explorer windows in InPrivate mode. 2. Go to http://localhost:31332 3. Log in into the application. User: **Andrew.Davis@[tenantname].onmicrosoft.com** Password: **[password]** 4. Click on **Team Travels** tab**.** 5. Type Carole in the **search box**. You will see the travels requests from Carole Poland, including the recently added one. 6. Click on **Reject** request button next to the new travel request. | Now I’ll log in as Andrew Davis. As a manager he can manage his own travel and accept or reject his team’s travel requests.  As Andrew is Carole‘s manager, the new travel request will appear in his Team Travels list.  I can use the search field to search for it. Here it is.  Carole doesn’t really need to go to San Francisco. Let’s reject the request. |
|  | 1. Write a **reason**. 2. Click on **Accept**. | We should really give her a reason why we are rejecting her request.  Not enough budget… Send. |
|  | 1. Return to Carole Poland’s window. 2. Click on **New Travel** button to create a new travel. 3. Fill the **form**. 4. Click on **Accept Changes** button | Now, let’s return to Carole.  If first she doesn’t succeed, try try again.  Let’s hope this request gets accepted. |
|  | 1. Switch to Andrew Davis **window** | Now I’m Andrew again!  You will notice that a toast notification has arrived via SignalR. |
|  | 1. Click on Team Travels tab**.** 2. Click on Approve request button**.** | Andrew is in a bit better mood, so we will approve the request. |
|  | 1. Click on **Heat Map**. | To see the most common destinations, we can switch to the Heat Map.  This shows the integration with Bing maps. |
|  | 1. Click on a **pushpin**. | If we select a city, you can see information about people traveling to that city and some statistics about the trips. |
|  | 1. Open a new Internet Explorer windows in **InPrivate** mode. 2. Go to **http://localhost:31332** 3. Login. User: **Thomas.Andersen@[tenantname].onmicrosoft.com** Password: **[password]** 4. Click on All travels. 5. Change page by clicking on the page numbers. 6. Click on the Edit request button on the first travel. | Now I’m switching to Thomas Andersen; a member of the HR group. Members of this group can see the travel for all of the company, add attachments, and mark them as processed.  I’ll edit a request… |
|  | 1. Click on the download attachment icon  to download the attachment. 2. Click on the delete attachment icon  to delete the attachment. 3. Insert a name for the attachment in the attachments textbox and click on the upload file button to select an attachment and upload it. 4. Click on the Accept changes button. 5. Close Internet Explorer | There is a attachment, so let’s download it.  I don’t really need that, I’ll delete it.  Let’s upload the correct attachment instead. |
|  | 1. Change the browser that **Visual Studio** will launch when debugging your application to Google Chrome 2. Press **F5** to run the application. 3. **Login**. User: **Andrew.Davis@[tenantname].onmicrosoft.com** Password: [password] 4. Ensure that User-Agent switcher for Chrome is installed in Chrome | Now I’ll change the browser that will open for debugging to Google Chrome. |
|  | 1. Click on **User-Agent switcher** for chrome. 2. Select **Windows Phone** 3. Select **Windows Phone 7** | Using the User-Agent switcher, I’ll change to mask as Windows Phone 7. |
|  | 1. Resize the window up to a **width** of 480px. | I’ve changed the width of the browser to 480 px to simulate the width of a mobile device. |
|  | 1. Click on **Team Travels** tab. | Here we can see the list of team travels as would be seen on mobile |
|  | 1. Click on one **travel** to see the details. | …and here is the travel details. |

## Desktop application

|  |  |  |
| --- | --- | --- |
| Demo Screen | Steps | Script |
|  | 1. Open the **MyCompany.Travel.sln** solution from the **Travel** folder. | There is also a Windows Store version of the Travel app. |
|  | 1. Right-click on **MyCompany.Travel.Client.Desktop**. 2. Click on **Set as StartUp Project** 3. Press **F5** to run the application |  |
|  | 1. Log in. User: **Thomas.Andersen@[tenantname].onmicrosoft.com** Password: **[password]** 2. Navigate through the pages using the pagination control. 3. Click on the edit button of the first travel. | Let’s log in as our HR guy, Thomas Andersen, again. |
|  | 1. Click on the **download** attachment icon  to download the attachment. 2. Click on the **delete** attachment icon  to delete the attachment. 3. Insert a **file name** in the attachments textbox and click on the upload file button to select an attachment and upload it. | As before, we can download, delete and upload attachments. |
|  | 1. Click on the **accept changes** button. |  |
|  | 1. Click on the **New travel** button. 2. Fill in the **form**. | Here is the form for adding a new trip in the Windows Store client. |

# Staff Tour

Staff is a LightSwitch HTML client targeting HR employees, who may create new employee data or update it.

There are two versions. The first one is a standalone web application. The second one is an O365 CBA (Office 365 Cloud Business Application – aka. LightSwitch App for SharePoint).

Like other MY COMPANY apps, it is integrated with the rest of the applications through an event-bus, which is internally implemented using Windows Azure Service Bus, so that any change to an employees’ data can be propagated to the other applications.

### Goals

To show how you can use LightSwitch as a great RAD (Rapid Application Development) tool for data-driven applications or sub-systems, either standalone or integrated with SharePoint and Office 365.

### Technologies being used

* LightSwitch HTML5 Client
* Fluid Layout
* Can be used as a standalone web app or integrated with SharePoint (called O365 Cloud Business Application), as provider-hosted
* Custom jQuery Mobile theme
* Custom controls (upload images, Bing maps, etc.)
* Custom service methods (Web API, handlers, etc.)
* Deployed in Windows Azure Web Sites
* Custom SharePoint Chrome
* Integrated with BCS (Business Catalog Services)

|  |  |  |
| --- | --- | --- |
| Demo Screen | Steps | Script |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\001-LS_Open solution.png | 1. Open the **MyCompany.StaffAlone.sln** solution from the **Staff** folder. | Next, we’ll look at the Staff application.  This allows us to manage the staff of our company. |
|  | 1. In the solution explorer window right-click in the **MyCompany.Staff.Web** project and select **Set as StartUp** **Project**. 2. Run the application by pressing **F5**. |  |
|  | 1. In the solution explorer window right-click in the **MyCompany.Staff.Database** project and select Publish. |  |
|  | 1. Click the **Load Profile…** button. |  |
|  | 1. In the root folder of the database project, select the publish profile of the database. |  |
|  | 1. Click the **Publish** button to create the database. |  |
|  | 1. Resize your **window**. | Here I can resize the window and the layout will responsively adapt to the new size. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\02-LS_Home_Responsive.png | 1. Click on **Calendars**. | I would like to add a new calendar.  First, I need to click the calendar tile and add a calendar. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\03-LS_CRUD_AddCalendar.png | 1. Click on **Add Calendar**. |  |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\03-LS_CRUD_Calendars (2).png | 1. Close the **dialog**. |  |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\04-LS_OfficeDetails.png | 1. Click on **Offices**. 2. Click on **Seattle Office**. | The Offices tile shows all the available offices.  Here is the Seattle office. Bing Maps is integrated to show the office location. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\05-LS_Offices.png | 1. Click on **Offices Map**. 2. Go to **main** menu. 3. Click on **Teams**. | The Offices Map shows all the offices around the world. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\06-LS_Teams.png | 1. Click on **Core business** team. | You can see a list of all the teams of your company via the Teams tile. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\07-LS_Teams.png | 1. Click on **team members**. 2. Go back to **main** menu. | Here are all the individual team members. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\08-LS_Employees.png | 1. Click on **employees**. | The Employees tile shows all the employees in the company. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\08-LS_Employees_Fluid.png | 1. Resize the window to see the **fluid** interface. | As before, the interface is fully responsive. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\09-LS_Employees_Details.png | 1. Click on **Adam Barr**. | Here’s the details of our friendly web developer, Adam Barr. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\10-LS_Employees_Details_Delete.png | 1. Resize the **window** to full size. 2. Go **back**. 3. Select **Andrew Davis**. 4. Click on **Delete** button. 5. Click on **No** button. | Do you remember our manager, Andrew? Let’s delete him.  Just joking, we might still need him for approving our trip home. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\11-LS_Employees_Details_Edit.png | 1. Click on the **image**. | Instead, we can crop him image. |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\12-LS_Employees_Details_Change_Image_BootStrap.png | 1. Select an **image**. 2. Move the **crop** selector. 3. Click **Next**. |  |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\13-LS_Employees_Details_Change_Image_Crop.png | 1. Resize and move the **crop** selector. 2. Click on **Save** button. |  |
| D:\Plain\Projectes\mycompany\demoscripts\screenshots\Tour\Staff\14-LS_Employees_Details_SignalR.png | 1. Open a **new instance** of the application. 2. Go to **employees -> Andrew Davis** in both windows. 3. Change the **name** in one of the windows. 4. See that the name has changed in the **other window**. | Now I want to show how the data is synchronized in real time.  I’m opening another instance of the app, and opening Andrew’s profile again.  I’ll change his name to Andy.  As you can see, his name was updated right away in our other instance. |

# Vacation Tour

Vacation is a web application that allows employees to request and manage vacation periods. The web application targets both regular employees and managers. Each role can perform different actions. Employees can create and edit their vacation requests. Managers can manage the team vacation requests and can view overlapping requests.

Email notifications are sent in some circumstances and there are also web notifications using SignalR.

This tour only show how the main features.

### Technologies being used

* Server logic exposed using OData endpoints
* Entity Framework 6 Code-first
* SPA application developed with AngularJS and jQuery
* LocalDb for local development environment
* Server components deployed on Windows Azure Web Sites + SQL Azure

|  |  |  |
| --- | --- | --- |
| Demo Screen | Steps | Script |
|  | 1. Open the **MyCompany.Vacation.sln** solution from the **Vacation** folder. | Finally, let’s take a look at the Vacation application. |
|  | 1. In the solution explorer window right-click in the **MyCompany.Vacation.Web** project and select **Set as StartUp** **Project**. 2. Run the application by pressing **F5**. | If WAAD is not configure, use the test mode that we have seen in the previous point. |
|  | 1. Log in. User: **Carole.Poland@[tenantname].onmicrosoft.com** Password: **[password]** 2. Navigate through the different **months**. 3. Take a look at the **bottom** of the page, where you can find the **vacations requested** by you. | After all that travelling Carole needs a vacation.  At the bottom you can see the vacation requests. |
|  | 1. Select the month **November**. 2. Click on a **vacation request**. You will see the request detail and you can cancel it. 3. Close the **dialog**. | Turns out November is a busy month, so I’ll cancel the request. |
|  | 1. Click and drag on the **calendar** to request new vacation days. | I’ll make a new request instead.  You cannot click on a weekend or holidays. If the selection contains weekends or holidays, these days won’t be included. |
|  | 1. Enter a **comment**. | I’ll add a comment and submit the request.  The calendar will show the new request and you will also see the request at the bottom of the page. |
|  | 1. Open a new Internet Explorer windows in **InPrivate** mode. 2. Go to **http://localhost:31331** 3. Log in. User: **Andrew.Davis@[tenantname].onmicrosoft.com** Password: **[password]** 4. Click on **Team Vacations** tab. | Now back to Andy. As a Manager, he can view all the requests by team members, both as a list and by overlapping requests. He is able to accept or deny these requests. |
|  | 1. Click on the **deny request** button of the last vacation request. A dialog will appear. 2. Click on **Yes** to deny the vacation request. | As he is really mean, he is going to deny the request. |
|  | 1. Switch to Carole Poland’s **window**. 2. Click and drag on the **calendar** to request new vacation days. | Carole is determined to take some time off, so let’s create a new request. |
|  | 1. Enter a **comment**. | As before, the request will show in the calendar and at the bottom of the page. |
|  | 1. Take a look at Andrew Davis’ **window**. | In Andy’s window you will see a toast notification has arrived thanks to SignalR |
|  | 1. Click on **Overlaps** tab. | In the overlaps tab, you will see a grid showing the overlapping team vacation requests. You can also accept or deny the requests from here by clicking on the request. |

# Deploy MyCompany to local IIS

This point describes how to deploy MyCompany to your local IIS. This steps could be helpful is you want to do a live demo without using Visual Studio.

## Premilinary steps

### Internet Information Server

Go to Control Panel -> Uninstall or change a program -> Turn Windows Features on or off.

In Internet Information Services -> World Wide Web Services -> Application Development Features select ASP.NET 4.5 and WebSocket Protocol.

In .NET Framework 4.5 -> WCF Services select HTTP Activation.

### SQL Server 2012

Install SQL Server Express 2012 (.\SQLEXPRESS) with SQL Server Management Studio. Remember to enable the authentication mixed mode.

Configure the sa user with the password that you want.

If you have another SQL Server instance installed or with different credentials, it´s possible to use it by doing additional steps.

### Edit host file

Open Notepad.exe as the administrator and change the file c:\Windows\System32\Drivers\etc\hosts adding the following line at the end of the file:

127.0.0.1 mycompanyserver

We need to be able to access the website by name for the authentication system to function correctly.

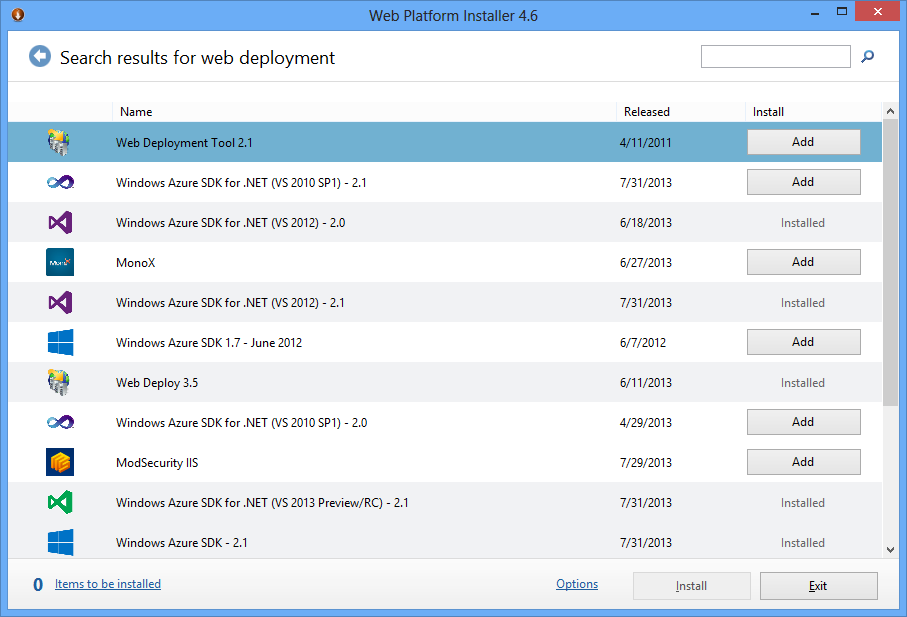
### Emails directory

Create a folder called C:\temp\mycompany and grant write permissions for the group **IIS\_IUSRS** (to allow receiving of emails).

Grant write permission to “c:\inetpub\wwwroot” for the group **IIS\_IUSRS**

### Web Deployment Tool

Install the Web Deployment Tool:



## Travel Web

1. Open Visual Studio as administrator.
2. Open solution MyCompany.Travel.sln
3. Select MyCompany.Travel.Web
4. Click on Publish
5. Deploy using the profile “FileSystem” which compiles using the Release\_LocalDeploy configuration.
6. Deploy to c:\inetput\wwwroot\travel
7. In Release\_LocalDeploy set the connection string to .\SQLEXPRESS using the sa/[password] credentials.
   1. In the Release\_LocalDeploy.config file you can change this value before deploying.

## Vacation Web

1. Open Visual Studio as administrator.
2. Open solution MyCompany.Vacation.sln
3. Select MyCompany.Vacation.Web
4. Click on Publish
5. Deploy using the profile “FileSystem” which compiles using the Release\_LocalDeploy configuration.
6. Deploy to c:\inetput\wwwroot\vacation
7. In Release\_LocalDeploy, set the connection string to .\SQLEXPRESS using the sa/[password] credentials.
   1. In the Release\_LocalDeploy.config file you can change this value before deploying.

## Visitor Web

1. Open Visual Studio as administrator.
2. Open solution MyCompany.Visitors.sln
3. Select MyCompany.Visitors.Web
4. Click on Publish
5. Deploy using the profile “FileSystem”, which compiles using Release\_LocalDeploy configuration.
6. Deploy to c:\inetput\wwwroot\visitors
7. In Release\_LocalDeploy set the connection string to .\SQLEXPRESS using the sa/[password] credentials.
   1. In the Release\_LocalDeploy.config file you can change this value before deploying.

## Expenses Web

1. Open Visual Studio as administrator.
2. Open the MyCompany.Expenses.sln solution
3. Select MyCompany. Expenses.Web
4. Click on Publish
5. Deploy using the “FileSystem” profile, which compiles using the Release\_LocalDeploy configuration.
6. Deploy to c:\inetput\wwwroot\expenses
7. In Release\_LocalDeploy set the connection string to .\SQLEXPRESS using the sa/[password] credentials.
8. In the Release\_LocalDeploy.config file you can change this value before deploying.

## Create IIS applications for Web Apps

Open the *IIS Manager* application and expand the *Sites* folder. We will use the **Default Web Site**.

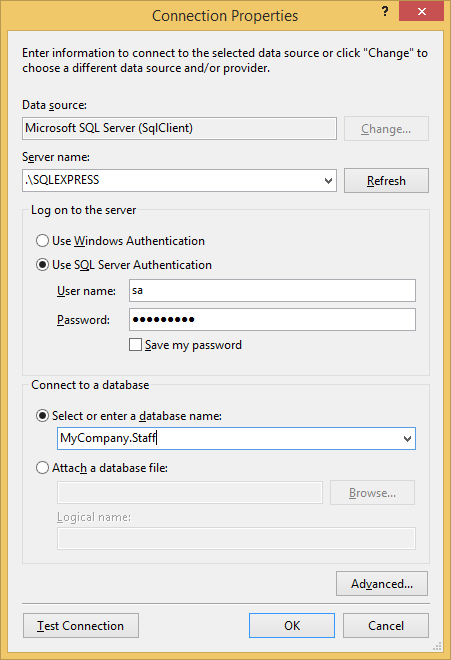
You will see four directories; visitors, vacation, travel and expenses. Convert them to an application.

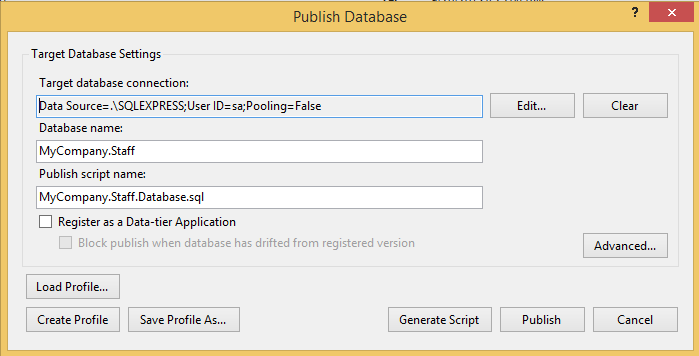
After that, you can open the web pages at the following addresses:

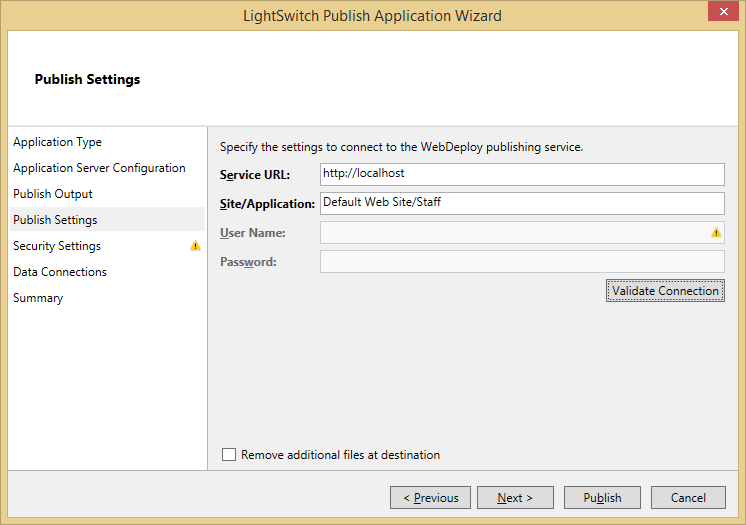
* <http://mycompanyserver/visitors/>noauth
* <http://mycompanyserver/travel/>noauth
* <http://mycompanyserver/vacation/>noauth
* <http://mycompanyserver/expenses/>noauth

## Staff

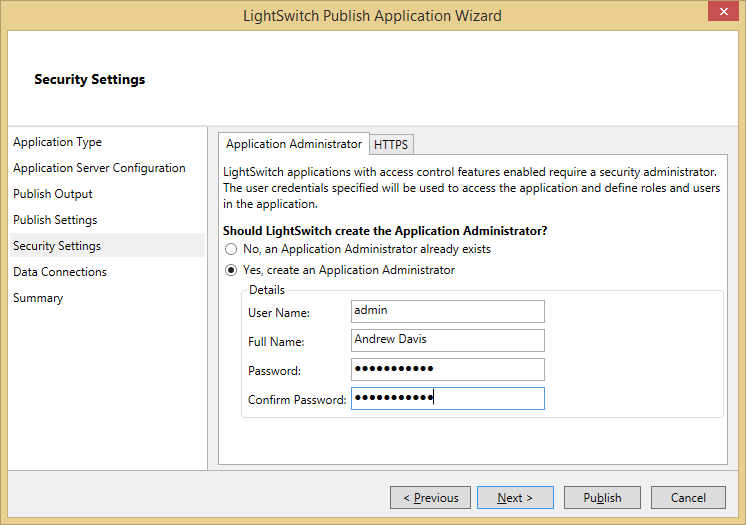
1. Open Visual Studio as an administrator.
2. Open the MyCompany.StaffAlone.sln solution.
3. Right-click on MyCompany.Staff.Database project and click on Publish.
4. Click on the Edit button in Target Database Connection
5. Fill in the following information; .\SQEXPRESS, sa/[password] and database name: MyCompany.Staff



1. Click OK.
2. Fill in the information from the previous form.  
   
3. Click on Publish. The database will be published in your .\SQLEXPRESS instance.
4. Right-click on MyCompany.Staff.Web project and click on Publish.
5. Select the **Application Type** tab.
6. Click on Complete application. Click **Next**.
7. Click on IIS Server. Click **Next**.
8. Click on publish directly to a server now. Click **Next**.
9. Set the service URL: <http://localhost>
10. Set the site/Application: **Default Web Site/Staff**

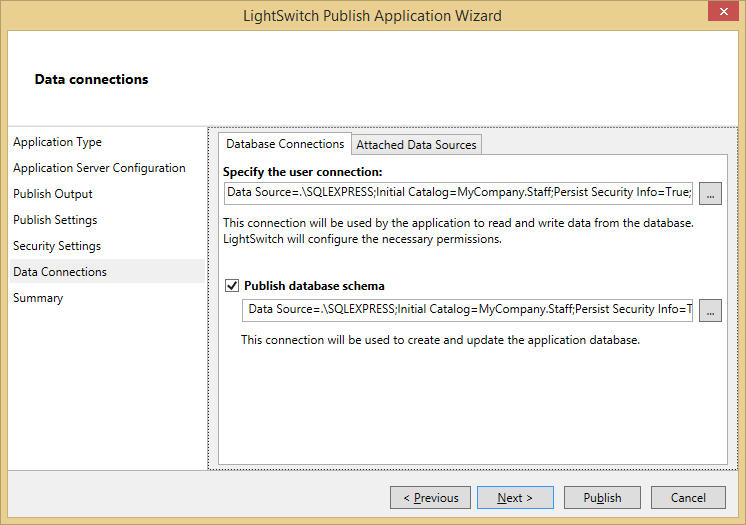


1. Click **Validate Connection**. Assuming it succeeds, click **Next**.
2. Select **Yes, create an Application Administrator**. Use the username **admin** with the password **[password]**
3. Select the **HTTPS** tab, and select **No, HTTPS is not required**. Click **Next**.



1. Across the two tabs there are three places to configure data connection strings. Use the following for all three:

Data Source=.\SQLEXPRESS;Initial Catalog=MyCompany.Staff;Persist Security Info=True;User ID=sa;Password=[password]



1. Click **Publish**.
2. Open the browser and go to <http://mycompanyserver/staff/HTMLClient>
3. User admin – [password]

## Visitors Windows Store

1. Open the MyCompany.Visitors.sln solution.
2. Select MyCompany.Visitors.Client.WindowsStore.
3. Open the file Settings | AppSettings.cs
4. Search for the property ApiUri and set the value <http://mycompanyserver/visitors/> in the default value.
   1. The URI can be change in the setting page.
5. Select MyCompany.Visitors.Client.WindowsStore.
6. Click on Store | Create App package.
7. Click on “No”.
8. Click on Create.

This action created a new package that you can install in your Windows 8.1.

## Expenses Windows Store

1. Open solution MyCompany.Expenses.sln.
2. Select MyCompany. Expenses.Client.WindowsStore.
3. Open the file js | config.js
4. Search for the property ApiUri and set the value <http://mycompanyserver/expenses/> in the default value of defaultApiUrl.
   1. The URI can be change in the setting page.
5. Select MyCompany. Expenses.Client.WindowsStore.
6. Click on Store | Create App package.
7. Click on “No”.
8. Click on Create.

This action created a new package that you can install in your copy of Windows 8.1.

## Travel Desktop (WPF)

1. Open the MyCompany.Travel.sln solution.
2. Select MyCompany.Travel.Client.Desktop.
3. Open the app.config file
4. Change the ServiceUrlBase property to <http://mycompanyserver/travel/>
5. Build and get the files from the bin\Debug|Release directory.