National College of Ireland

BSc in Computing- Cyber Security

2017/2018

Technical Report

ThinScale Technology

Giuseppe Stirpe

Student Number: x13132181

Supervised by Vikas Sahni

National College of Ireland

Table of Contents

Executive Summary

The objective of this project, is to design an application for ThinScale Technology Employees and Administrators. This application will allow the employee to register first and then log in using a dedicated form where they will be able to create events, book holidays and have a general overview of which days are available through a built-in scheduler form. They will also have access to a reporting tool, which will show general information about days already booked, employees information, sales data and they will also have the possibility of using a browser form within the application, to send emails and change their personal password. The most important part of the application is for the Administrators. From their point of view, they will be able to modify employee's details, create a new entry, delete them, check and create payments, print pay slips, send emails, review reports and delete or review events or holiday coming from the scheduler form. All users will be using the same login form, but based on their role, the application will show different windows forms and user will have different permissions.

All the data shown in the forms is pulled from an external SQL Server stored in Microsoft Azure. The data is retrieved using stored procedure to increase security and lower the risk of SQL injection or penetration from non-authorized personnel and using firewall ports within the Azure portal to restrict access only to authorized IP address or IP addresses ranges. To increase also the account security, all passwords are hashed and encrypted, before being stored in the database, few criteria and requirements like minimum length, special characters, symbols and numbers, are also enforced. Admin or delegates that have the authority to do so will have the capabilities to check employees' payment records, print pay slips, and compute new payments. The application will not be available on the public domain and the setup.exe file will be only available for download from "Box Works App": a file sharing and collaboration tool. If the employees want the application, they will request it through the web page and a link will be sent to them for a secure download.

Web site URL:

(http://www.giuseppestirpe.com/home/giuseppestirpe89)

1 Introduction

The aim of this document is to provide all the technicality involved in developing ThinScale Management Application. The below section will describe in detail what was used and why all the components were used.

1.1 Background

After I was successfully hired from ThinScale Technology, back in March, I started being exposed to the internal company organization applications and projects. After few months into the job, I noticed that there wasn't a tool to organize employee's annual leaves, sick days or the employee in general. The idea for this application entered my mind straight away. Why not developing an application, which all of us can use to schedule holidays, appointments, events and give the Administrator the option to have a more organized internal employee structure? I did some market researches and found some interesting applications that already exist for managing employees and holidays. Applications like AnnualLeave.com, OrangeHRM, E-days or even Salesforce were already on the market and well documented, but they come with a high tag price. It is not unusual for a start-up like ThinScale to reduce expenses and focus on what is truly needed. At the end of the day, they would invest more money in something that can give them back a higher R.O.I. rather than an application to organize employees. After coming up with the idea, I asked one of the engineers if I could use the office infrastructure to develop the application and I also asked to a couple of my colleagues if the application would be useful for the company. After receiving valuable feedbacks, I scheduled a meeting with Brendan, the company's director. I explained to him my point of view, the planning, the approach I would use for developing it and a mockup prototype (see appendix) showing the application' layout. After a small discussion, he suggested few changes to be made to improve the application, but at the end, he was happy and surprised that I wanted to develop something to help the company's internal organization. A few weeks later, during the Dragon's Den presentation, the lecturers were satisfied as well, and I could finally start planning and developing my application: ThinScale Application Suite was ready to be built.

1.2 Aims

The aim of the project is to build an application that allows employee to book holidays, view reports, use a built-in web browser and send emails, and to give Admins a tool to organize their employee in an easy and functional way. The application will show different forms based on the user' role, after a successful registration. Employee will have a restricted view where they can only create, update, view booked events/appointments/vacation but only delete their own one. They will be able to review reports, send emails and change their passwords, while Administrators will have a full create, read, update, delete, capabilities throughout the entire application. They will be able to create employee's entries, compute payrolls, send emails and view or delete holidays. The application will be accessible only on desktop, being just a Windows form application, but it will be available all the time with the use of a VPN (Virtual Private Network), or a RDP (Remote Desktop Protocol) connection. Before the final release a series of stress testing mechanism will be applied prior, during and after development, to ensure that crashes, bugs, behaviour and functionalities are in line with the expectations.

1.3 Technologies

The application will be developed using the C# programming language and WinForms with the aid of Visual Studio IDE. C# is a class-based, object-oriented programming language developed by Microsoft, while Windows Forms is a graphical class library included as a part of Microsoft .NET Framework, providing a platform to write rich client applications for desktop, laptop, and tablet PCs. The motivation towards this programming language and environment was driven by the fact that I am exposed daily with this technology, and mostly because since my third year in college, I was comfortable with C#, being ASP.Net, but the choice at the end was easy. Just to prophase, that before diving in with the full implementation of my main project, I started with smaller ones. I developed a calculator to get comfortable with the general toolbox, palette and drag and drop functionalities, to move on with a semi functional outlook style application using the out of the box Visual Studio look and feel. While developing using the default Visual Studio tool box (See Appendix), I found myself being a bit restrict in terms of customization and style, so from day one I knew I had to find something else to get the look and the "wow" factor I was looking to have in my application. After researching online for external library for WinForms I had the choice between DevExpress or Telerik. Both were really user friendly and easy to use in conjunction with Visual Studio; they both had a massive documentation and videos to follow, but at the end it come down to the price of the license. Telerik was 1299$ per perpetual license per developer while DevExpress was free due to the fact I use that at work. I didn't want the company to add an extra license for me on top of what they already have but after speaking with the main engineer he decided to add one for me so at the end, I went for DevExpress. From a back-end point of view, I've never had any previous NoSQL experience, so for this reason, a relational database will be used instead. I will be starting locally using a SQL Server Management Studio 2016 and the Object Explorer to move at later stage, to a complete hosted cloud SQL on Microsoft Azure. The Object Explorer provides a hierarchical user interface to view and manage the objects in each instance of SQL Server within Visual Studio IDE. To retrieve, insert, update and delete information coming from the database to the application, I will use stored procedures. The benefits of using stored procedures in SQL Server rather than application code stored locally on client computers include:

- They allow modular programming.
- They allow faster execution.
- They can reduce network traffic.
- They are used as a security mechanism.
- They prevent SQL Injection (OWASP 2017 n1 Risk)

When the application is not closed but just minimized, the user will be notified that the application is still running with the aid of a NotifyIcon. A NotifyIcon is a .NET component that is used to notify users by displaying an icon and an optional popup Balloon tooltip in the notification area of the system taskbar. When the employees minimize the application, a toast notification will appear, and when the icon on the systray is double clicked the main form will be shown back.

Once the application will be ready for deployment, ClickOnce and Signtool will be utilized. ClickOnce enables the user to install and run a Windows-based smart client application by clicking a link on a web page or by running the setup.exe file generated from Visual Studio on their local machines. I will be going for the second choice as ThinScale Application Suite is not a web-based application. Signtool is a Microsoft technology that uses industry-standard cryptography to sign application codes with digital certificates that verify the authenticity of the application's publisher and to prove its integrity. By using Signtool and ClickOnce I will reduce the risk of a Trojan horse or malware injection by signing the app with a digital signature (See Appendix). Once the file setup.exe has been deployed, it will be stored on the file and sharing collaboration tool “Fun Works”.

The web page, where employees will go to retrieve the setup.exe installation file link will be developed using WordPress Framework. The web page will have a very simple look, with few pages, forms and links. Once the form is submitted, an email with the link will be sent to the employees where they can then download the software. Once downloaded, they will run the setup.exe on their personal machine ready to be used. A manual will be also given to the employee. (See Appendix)

1.4 Structure

Section 1 – describes the general introduction, why I decide to develop the application and what was the cause of the development. Technology section briefly describes the technology used for the implementation and structure will show the general layout of the document created.

Section 2 –describes all the functional and non-functional requirements used. Use case diagrams describes the core functionalities for each type of user. The architecture of the application is represented with the aim of a class diagram. The graphical user interface screenshots are used to represent the applications actual look. The testing section will explain all the application applied to stress test the application. Based on the results changes and or modification of the code might be implemented.

Section 3 – will conclude the evaluation and the development of this software project.

Section 4 –describes the future functionalities, future road maps and potential updates for the application. The idea is to be competitive not only within ThinScale but also with other companies, gain market visibility and eventually more R.O.I.

Section 5 – describes all the potential forums, web site and link used to gather more information.

Section 6 – this section contains a copy of the project proposal, the monthly journals and supervisor meetings.

2 System

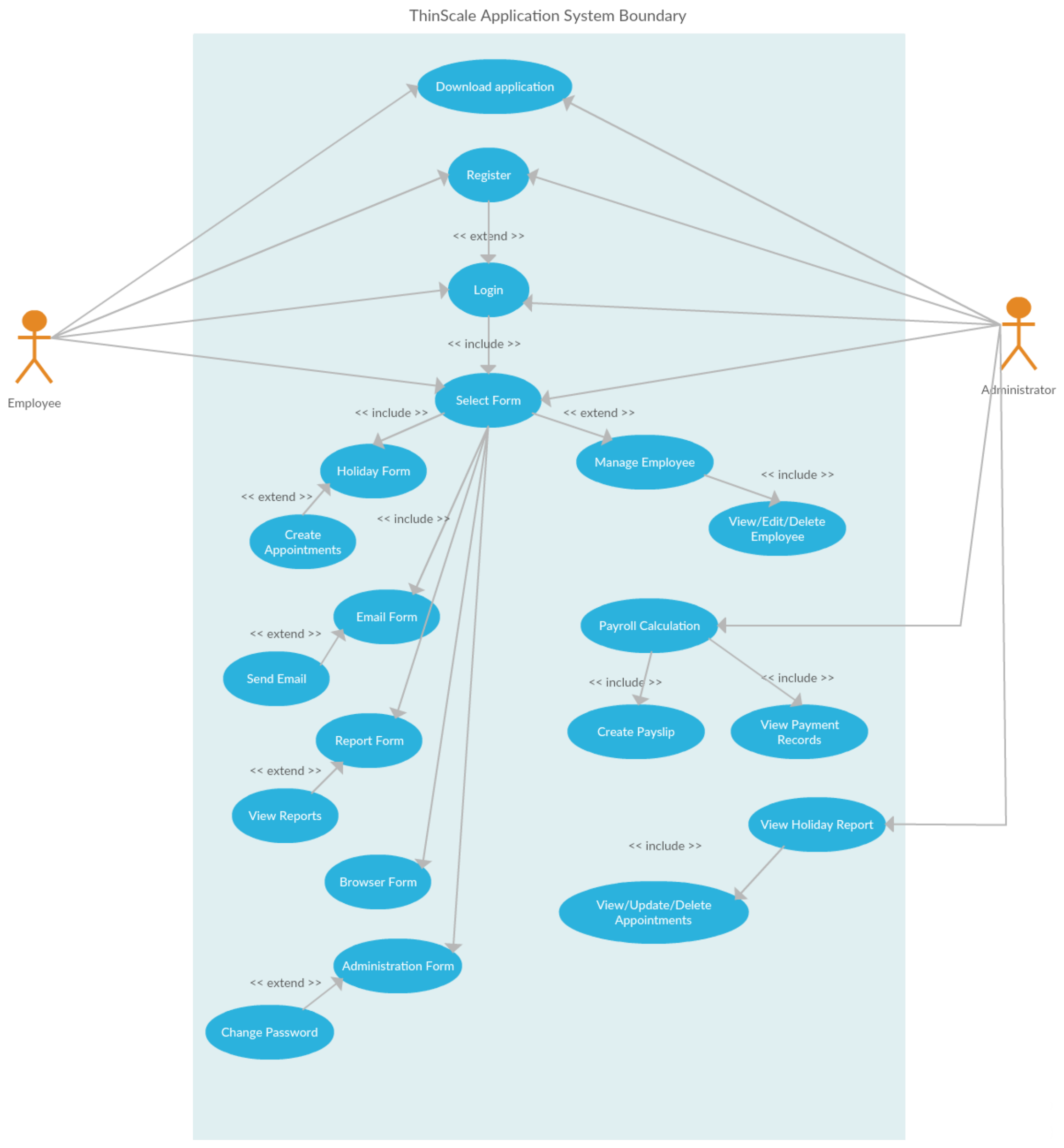
2.1 Requirements

All requirements are set out with a nice, simple graphical user interface and ease of use in mind. A small installation guide (See Appendix), will be provided for new employees and a small training will be required. On the first visit the employees must register to gain access to the application. From the Administrators point of view, registration is required too, but once logged in they will have more privileges over the application' forms and mostly over the employees list. The application is designed to have a nice flow overall and to operate clearly and easily.

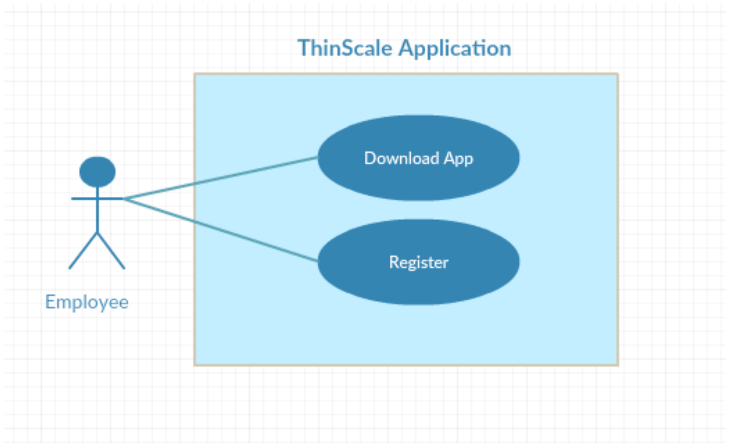
2.1.1 Functional requirements

The ThinScale Application Suite is comprised of different core components spread across each type of users, Employees and Administrator. The Application centres around a database that stores and returns required data based on the forms requested. Using an Azure Microsoft SQL Server's stored procedures, the application will return the required results. In this section, with use of a UML, the core functionalities are broken down per user with each action assigned to them. It describes the effects of each function and how the system performs each task based on the type of credentials they use to interact with the application.

2.1.1 Main Use Case Diagram



2.1.1 Requirement 1 – Download the application and Register



Description & Priority

This requirement is the primary function that provides the level of access to all users depending on their credentials and roles. If the user is an employee, the level of access to the application is restricted to few forms, however if the role is an Admin the level of access is higher.

Use Case – 1. Register

Scope

The scope of this use case is to provide the right credentials to access the application via a login.

Description

This use case describes how each employee gains access to the ThinScale Application by registering them into the application. This is priority 1 as provides employee credentials to access the application.

Flow Description

Precondition

System is idle on Registration Form waiting for user input.

Activation

This use case starts when a user clicks on the application icon and land on the Registration form.

Main flow

1. The system asks the employee to enter their username, password & role.
2. The Employee/Administrator enters unique username, password & role into the form.
3. The system checks user input against database to validate credentials
4. On validation of credentials the employee or Admin will be presented with the login form

Alternate flow

A1: Optional application view – Administrator

1. The system validates user credentials against access rights in stored database
2. The Administrator is directed to the Login form with read/write access
3. After successful register the use case continues onto point 2.

A2: Optional application view – Employee

1. The system validates user credentials against access rights in stored database
2. The employee is directed to application login forms.
3. After successful log in the use case continues onto point 2.

Exceptional flow

E1: Incorrect log in credentials entered

4. The system checks user log in credentials against database and rejects incorrect entry with a popup message on screen
5. The Admin or employee must re-enter the correct credentials
6. The use case continues at position 2 of the main flow after successfully login

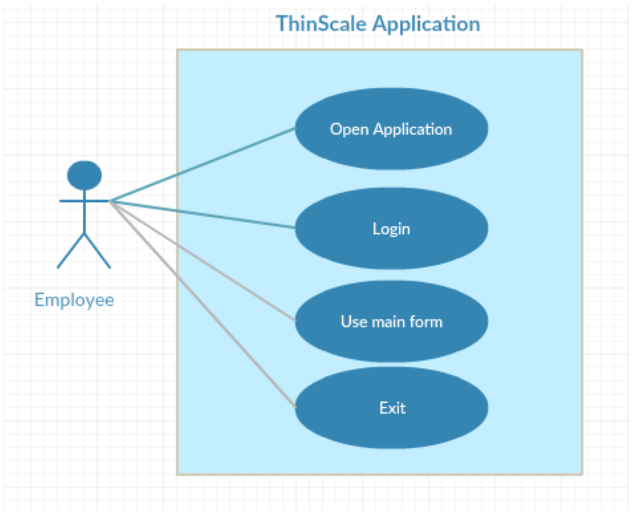
Termination

The system presents the Registration form.

Post condition

The system goes into a wait state

2.1.1 Requirement 2 - Log In



Description & Priority

This requirement is used to detects what view the employee will get after successfully validate their credentials. It is the "landing form" that all users must go through to use the application after successful registration.

Use Case – 2. Log In

Scope

The scope of this use case is to provide access to the employees of the application via a login form.

Description

This use case describes how each employee's gains access to the ThinScale Application using unique personal login credentials. This is priority 1 as provides employee access to the application.

Flow Description

Precondition

System is idle on Login Form awaiting user input.

Activation

This use case starts when a user clicks on the application icon and land on the Log In form if already registered.

Main flow

1. The system asks the employee to enter their username, password & role.
2. The Employee/Administrator enters unique username, password & role into the form.
3. The system checks user input against database to validate credentials
4. On validation of credentials the employee or Admin gains access to the main application forms.

Alternate flow

A1: Optional application view – Administrator

1. The system validates user credentials against access rights in stored database
2. The Administrator is directed to the application with read/write access
3. After successful log in the use case continues onto point 3.

A2: Optional application view – Employee

1. The system validates user credentials against access rights in stored database
2. The employee is directed to application with access to specific forms.
3. After successful log in the use case continues onto point 3.

Exceptional flow

E1: Incorrect log in credentials entered

1. The system checks user log in credentials against database and rejects incorrect entry with a popup message on screen
2. The Admin or employee must re-enter the correct credentials if registered
3. The Admin or employee will be redirect to Registration form if credentials are not in the database
4. The use case continues at position 3 of the main flow after successfully login if registered

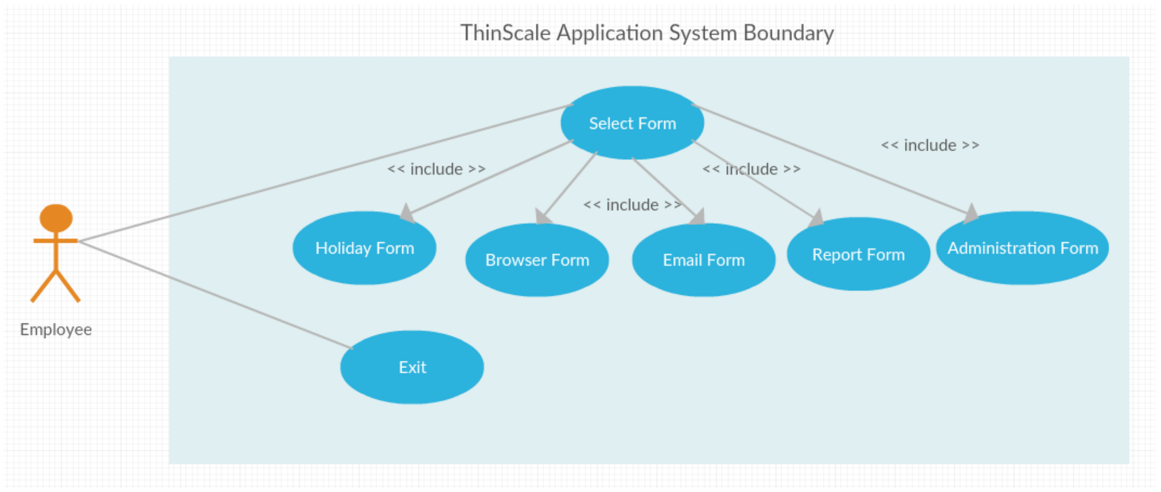
Termination

The system presents the login form.

Post condition

The system goes into a wait state

2.1.1 Requirement 3 – View Selected Forms



Description & Priority

Employees can view based on their credential specific forms. This is a priority 3 function enabling employees to browse and select the forms.

Use Case – 3. View Selected Forms

Scope

The scope of this use case is to show a list of all available forms to registered employees.

Description

This use case describes the first point of access for the employees displaying just the forms they are required to use and view.

Flow Description

Precondition

The system loads the main GUI where employees can interact using buttons from a Ribbon Bar or a Navbar.

Activation

This use case starts when an employee is logged in.

Main flow

1. The system identifies all registered employees stored in database
2. The employee browses and selects the forms
3. The system loads the forms they are requesting
4. The employee interacts with the application

Alternate flow

A1: Select different form

1. The system loads selected form
2. The employee clicks the main ribbon bar to navigate through the forms
3. The use case returns to position 3 if all forms are closed.

Exceptional flow

E1: Employee logs out

1. The system will close, and the login form will reappear back
2. The employee either close the application or logs back in
3. If the employee doesn't close the form the use case returns to position 2, else it exits.

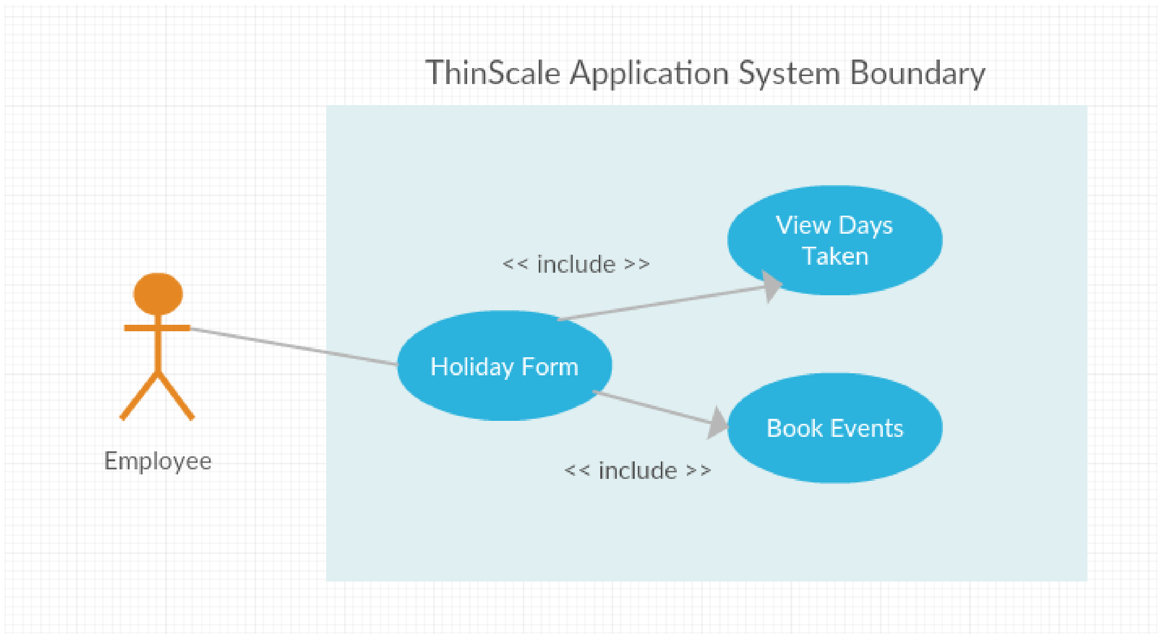
Termination

The system presents the login form.

Post condition

The system goes into a wait state

2.1.1 Requirement 4 – View Holiday Form



Description & Priority

Employees can view the days, events and appointments already booked and book new one. This is a priority 4 function, displaying the overall holiday situation.

Use Case – 4. View Holiday Form

Scope

The scope of this use case is to display to the employees, which day is already booked, who booked it, and which is one is free.

Description

This use case describes how a user can view booked events listing from a calendar scheduler.

Flow Description

Precondition

The system loads all details of selected month displaying weekly, monthly or daily availability

Activation

This use case starts when a customer selects the holiday scheduler form

Main flow

1. The system loads a calendar scheduler form
2. The employee can select different dates to be booked

Alternate flow

A1: Select different month

1. The system loads selected month
2. The employee closes the form and return to main window
3. The use case returns to position 3.

Exceptional flow

E1: Employee logs out

1. The system will close, and the login form will reappear back
2. The employee either close the application or open another form
3. If the employee doesn't close the form the use case returns to position 2, else it exits.

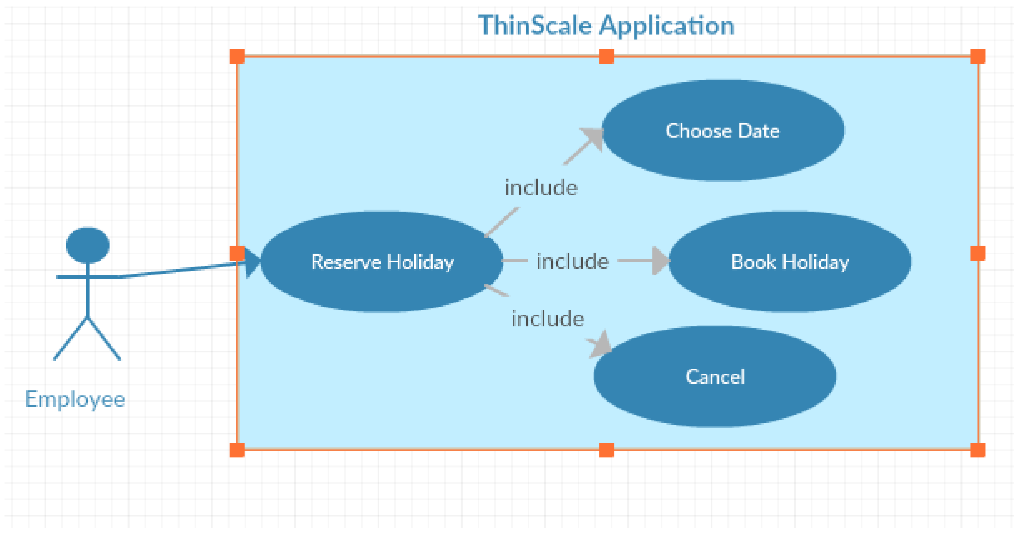
Termination

The system exits.

Post condition

The system goes into a wait state.

2.1.1 Requirement 5 – Reserve Holiday



Description & Priority

Employee can view the selected form and book holiday. This is a priority 5 function enabling employees to book their desired vacation period.

Use Case – 5. Reserve Holiday

Scope

The scope of this use case is to provide the form where employees can go and book their holiday with a scheduler.

Description

This use case describes how an employee can use a scheduler to book holidays

Flow Description

Precondition

The system loads a scheduler form sorted by month, day, or full year.

Activation

This use case starts when an employee clicks on the holiday forms.

Main flow

1. The system displays the scheduler form
2. The employee can navigate the months
3. The system loads specific selected month or day
4. The employee will view availability of the desired date

Alternate flow

A1: Select different month

1. The system loads selected month
2. The employee closes the form to return to main form window
3. The use case returns to position 3.

Exceptional flow

E1: Employee logs out

1. The system will close, and the login form will reappear back
2. The employee either close the application or logs back in
3. If the employee doesn't close the form the use case returns to position 2, else it exits.

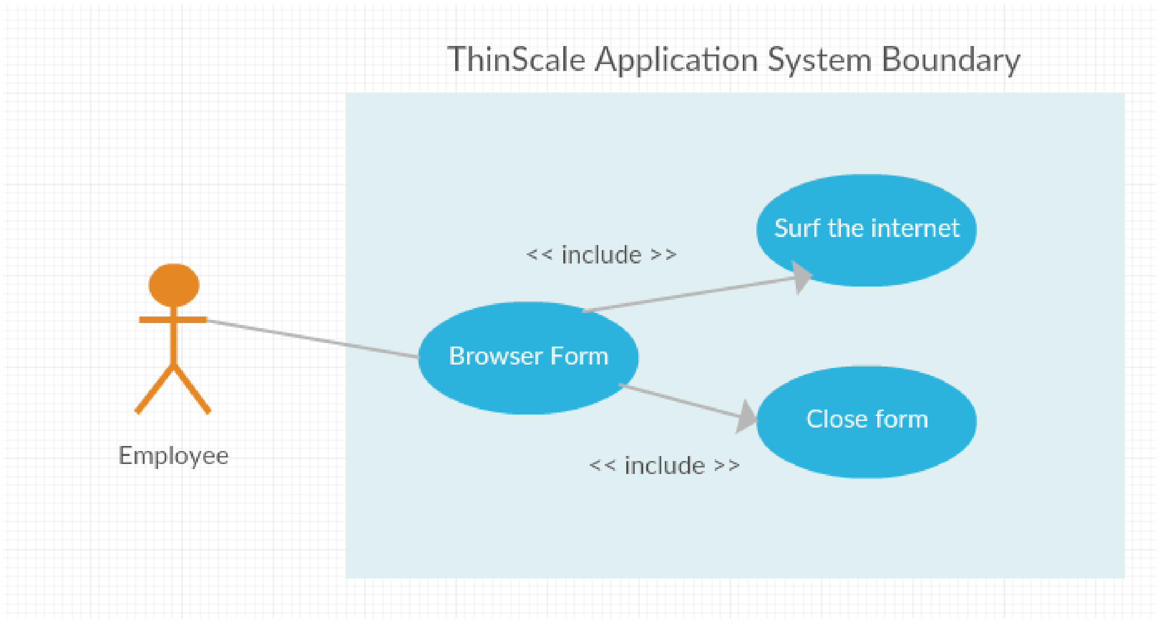
Termination

The system presents the login form.

Post condition

The system goes into a wait state

2.1.1 Requirement 6 – Browser Form



Description & Priority

Employee can view the selected form and surf the internet. This is a priority 5 function enabling employees to browse the web within the application.

Use Case – 5. Browser form

Scope

The scope of this use case is to provide the form where employees can go and surf the internet.

Description

This use case describes how an employee can use a build in web form

Flow Description

Precondition

The system loads a browser form.

Activation

This use case starts when an employee clicks on the browser form.

Main flow

1. The system displays the browser form
2. The employee can navigate through google or other
3. The system loads specific selected URLs

Alternate flow

A1: Select different URLs

1. The system loads selected link
2. The employee closes the form to return to main form window
3. The use case returns to position 3.

Exceptional flow

E1: Employee logs out

1. The system will close, and the login form will reappear back
2. The employee either close the application or logs back in
3. If the employee doesn’t close the form the use case returns to position 2, else it exits.

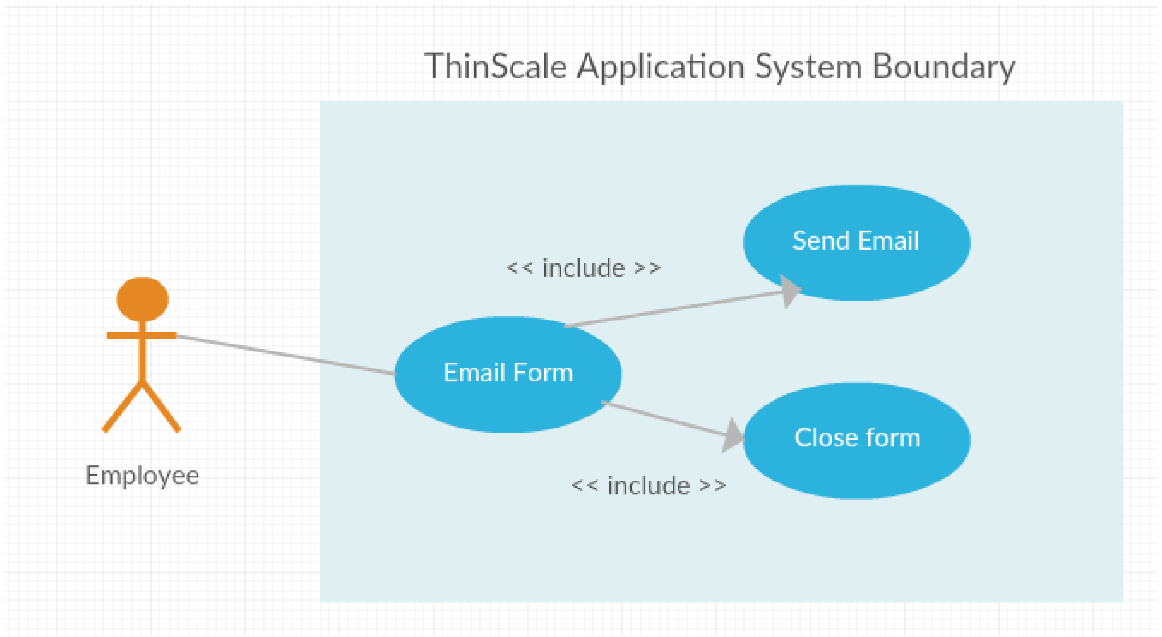
Termination

The system presents the login form.

Post condition

The system goes into a wait state

2.1.1 Requirement 7 – Email Form



Description & Priority

Employee can view the selected form and send emails. This is a priority 5 function enabling employees to send emails within the application

Use Case – 5. Email form

Scope

The scope of this use case is to provide a form where employees can go and send emails

Description

This use case describes how an employee can use a build in SMTP form for emails

Flow Description

Precondition

The system loads an email form.

Activation

This use case starts when an employee clicks on the email form.

Main flow

1. The system displays the email form
2. The employee can send emails

Alternate flow

E1: Employee logs out

1. The system will close, and the login form will reappear back
2. The employee either close the application or logs back in
3. If the employee doesn’t close the form the use case returns to position 2, else it exits.

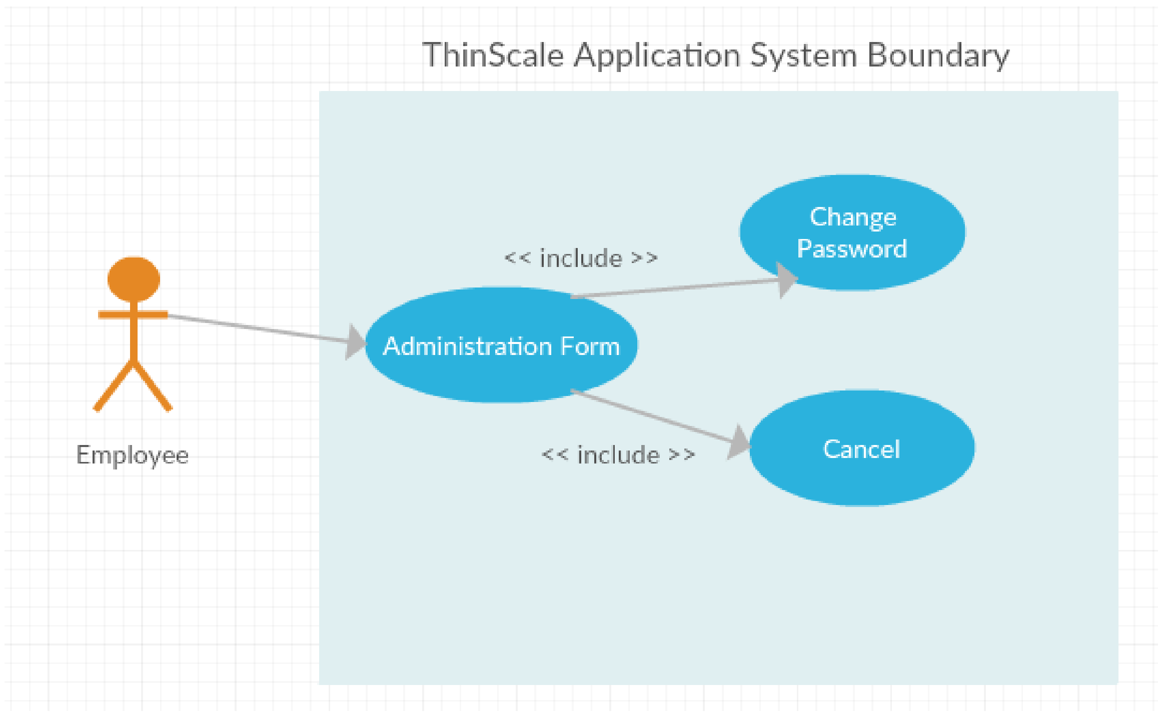
Termination

The system presents the login form.

Post condition

The system goes into a wait state

2.1.1 Requirement 8 – Open Administration Form



Description & Priority

Employee can open the selected form and change their password. This is a priority 5 function enabling employees to change their password within the application.

Use Case – 5. Open Administration Form

Scope

The scope of this use case is to provide a form where employees can go and change their password

Description

This use case describes how an employee can use a form to change their password

Flow Description

Precondition

The system loads a password change form.

Activation

This use case starts when an employee clicks on the change password form.

Main flow

1. The system displays the change password form
2. The employee can reset the password
3. The form close

Alternate flow

E1: Employee logs out

1. The system will close, and the login form will reappear back
2. The employee either close the application or logs back in
3. If the employee doesn’t close the form the use case returns to position 2, else it exits.

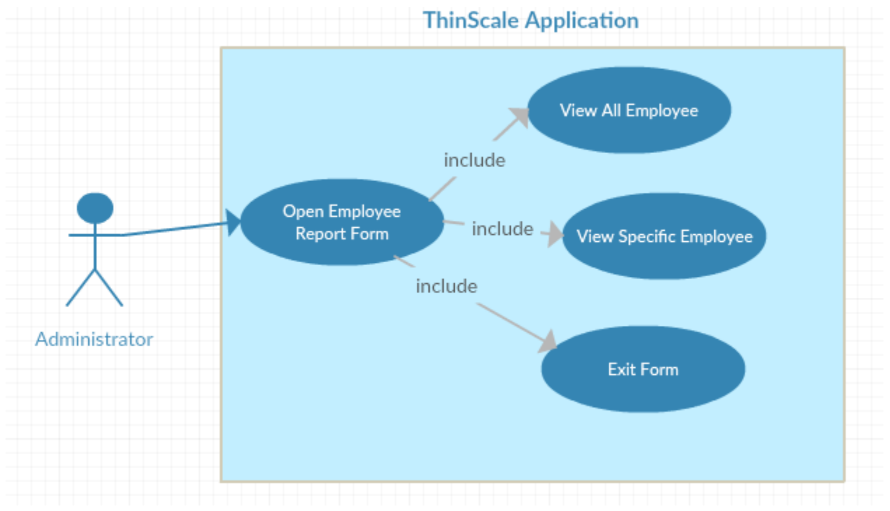
Termination

The system presents the login form.

Post condition

The system goes into a wait state

2.1.1 Requirement 9 – View Employees Report



Description & Priority

ThinScale Administrators can view the employee's report, which is displayed using different charts. This is a priority 3 for the administrators to have an overview about employees' information.

Use Case – 7. View Employees Report

Scope

The scope of this use case is to allow the users to view some information about the employees using the built-in charts.

Description

This use case describes how users can view, with the aid of chart information about the employees held in the database.

Flow Description

Precondition

The system loads the employee's data that is already saved into the application

Activation

This use case starts when an administrator logs in and clicks to the reporting form and selects "All Employees"

Main flow

1. The system displays the employee’s charts data
2. The user clicks the “Employee Report” button within the ribbon bar
3. The user closes the form

Exceptional flow

E1: Users logs out

1. The system will close, and the login form will reappear back
2. The employee either close the application or logs back in
3. If the employee doesn’t close the form, the use case returns to position 2, else it exits.

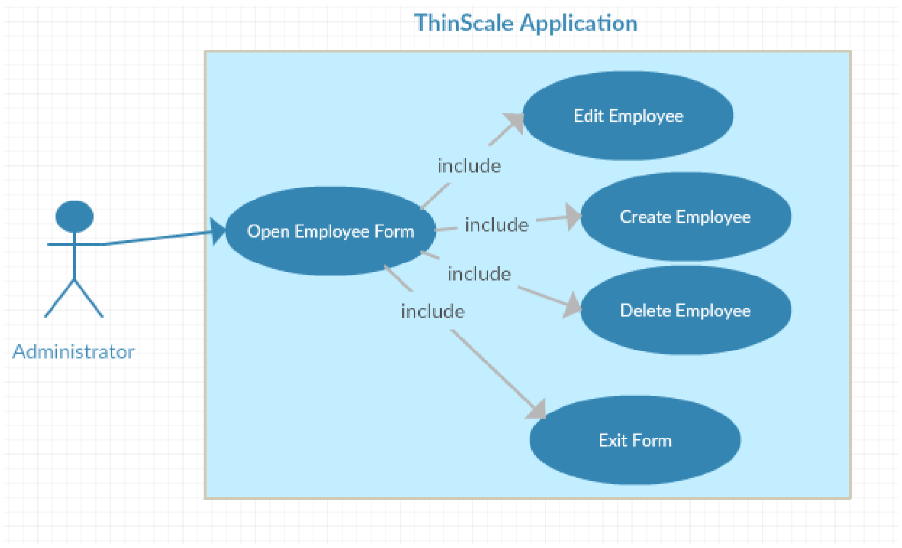
Termination

The system presents the login form.

Post condition

The system goes into a wait state

2.1.1 Requirement 10 – View/Edit Employees



Description & Priority

Administrators can view and edit the employee's details including id, names, address and so on, which is displayed in the allocated grid view. This is a priority 2 for Administrators to maintain employee details.

Use Case – 7. View/Edit Employees Details

Scope

The scope of this use case is to allow the administrators to maintain all the employee's details.

Description

This use case describes how an administrator can view, edit, delete and update the employee's details held in the database. Update names, status, address and more

Flow Description

Precondition

The system loads the employee's profiles.

Activation

This use case starts when an administrator logs in and clicks the "Employee form" button within the ribbon bar.

Main flow

1. The system displays the employee grid view and employee's details
2. The administrator clicks on the grid view to update, delete or view entry
3. The system enables full access to edit the section
4. The administrator makes required changes to employee's profile and saves changes to update database where a confirmation dialog will appear.

Alternate flow

A1: Undo Change
1. The system displays employees profile
2. The administrator makes change to profile but does not wish to commit and closes the form
3. The system exits and does not save changes to the database.

Exceptional flow

E1: Administrator logs out
1. The system will close, and the login form will reappear back
2. The employee either close the application or logs back in
3. If the employee doesn't close the form the use case returns to position 2, else it exits.

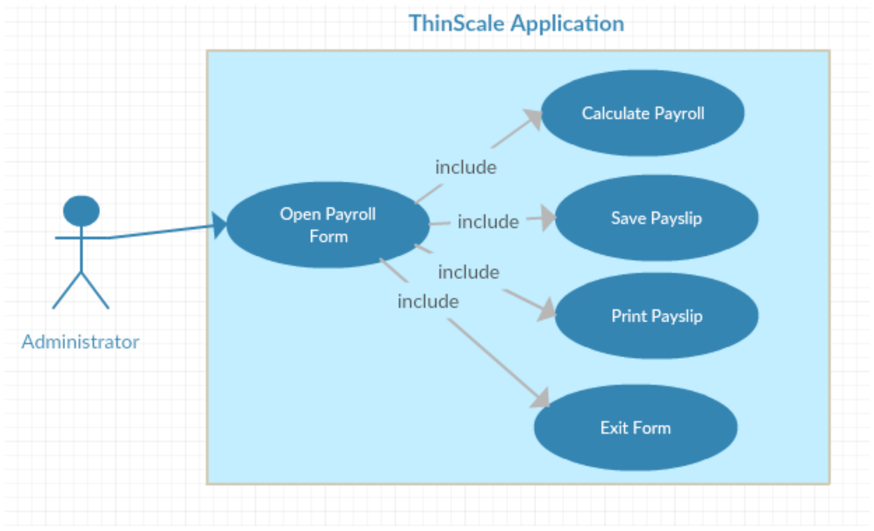
Termination

The system presents the login form.

Post condition

The system goes into a wait state

2.1.1 Requirement 11 – Calculate Payment and Payslip



Description & Priority

ThinScale administrators can calculate an employee payroll based on the number of hours he/she has worked and eventually create a payslip for them. This is a priority 4 for administrators to maintain the overall expense for the employees.

Use Case – 8. Calculate Payment and Payslip

Scope

The scope of this use case is to allow the administrators to update, create and view the number of hours an employee has worked and to generate and print prototype payslips.

Description

This use case describes how an administrator can view and update the employees' payroll held in the database based on the numbers of hours worked and to print it or display the payslip within a form.

Flow Description

Precondition

The system loads the payroll form and then the administrator will either get the employee Id or name to generate and calculate the overall number of hours to convert for salary purpose.

Activation

This use case starts when an administrator logs in and clicks the “Payroll calculator” button and gets the proper form

Main flow

1. The system displays the form where administrator can get the details they desire
2. The administrator clicks the “get” button to retrieve employee’s id or the “first name” or “last name”
3. The system enables full access to retrieve employee’s information
4. The administrator can update the payroll information, save it to the database and alternatively print it out as a document.

Alternate flow

A1: Undo Change

1. The system displays the payroll form
2. The administrator makes change to the employee’s details but does not wish to commit change and close the form.
3. The system close and does not commit change to database.

Exceptional flow

E1: Administrator logs out

1. The system will close, and the login form will reappear back
2. The employee either close the application or logs back in
3. If the employee doesn’t close the form the use case returns to position 2, else it exits.

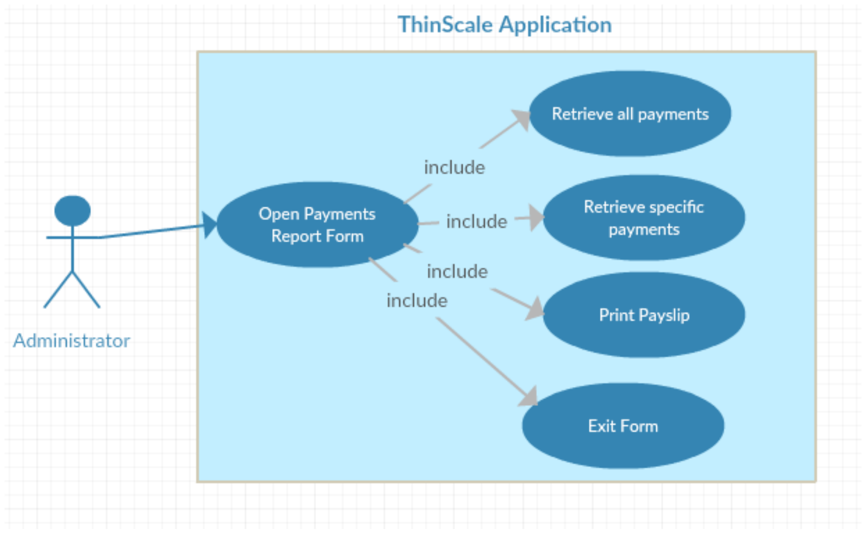
Termination

The system presents the login form.

Post condition

The system goes into a wait state

2.1.1 Requirement 12 – View Payments Reports



Description & Priority

ThinScale administrators can view all the payments that have been processed with the payroll calculator form and then saved in the database. This is a priority 5 for administrators to maintain the overall expense related to the employees.

Use Case – 9. View Payments Reports

Scope

The scope of this use case is to allow the administrators to view all the payments saved with the option to view the entire year or a specific month

Description

This use case describes how an administrator can view the employees' payments held in the database based on the information saved with the payroll calculator.

Flow Description

Precondition

The system loads the payments form and then the administrator will either get all the payments or can be granular enough to select specific month

Activation

This use case starts when an administrator logs in and clicks the reporting form “All payments” and gets all the results

Main flow

1. The system displays the form where administrator can get the details they desire
2. The administrator clicks the “All Payments” button to retrieve all the employee’s payments over one year or “Current Month” form to select a specific month
3. The system enables full access to retrieve employee’s payment information
4. The administrator can export or send via mail the payments information and alternatively print it out as a document.

Exceptional flow

E1: Administrator logs out

1. The system will close, and the login form will reappear back
2. The Administrators either close the application or logs back in
3. If the employee doesn’t close the form the use case returns to position 2, else it exits.

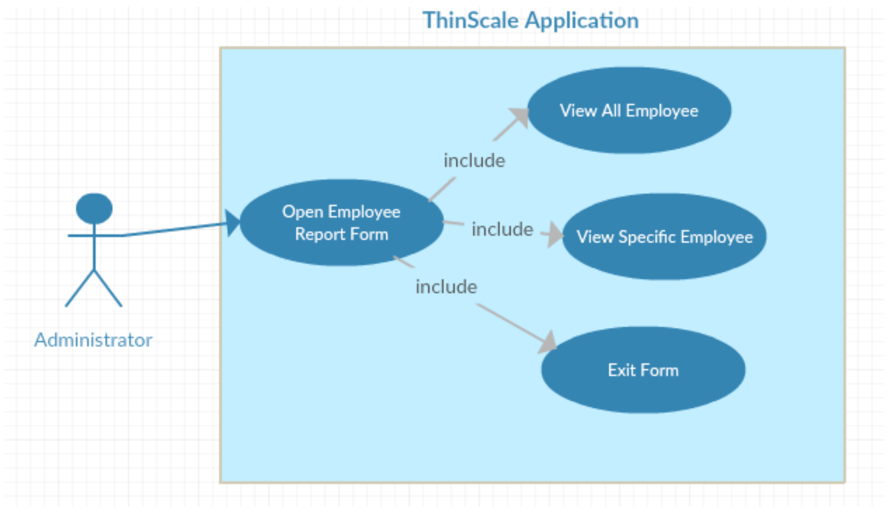
Termination

The system presents the login form.

Post condition

The system goes into a wait state

2.1.1 Requirement 13 – View All Employees Report



Description & Priority

ThinScale employee can view the employee’s report, which is displayed in a specific form. This is a priority 3 for the administrators to have an overview about employees’ information.

Use Case – 7. View All Employees Report

Scope

The scope of this use case is to allow the administrators to view all the information about the employees using the application.

Description

This use case describes how an administrator can view, through a simple form all the employees held in the database with their respective details.

Flow Description

Precondition

The system loads the employee’s profile that is already saved into the application using a specific form

Activation

This use case starts when an administrator logs in and clicks to the reporting form and selects “All Employees”

Main flow

4. The system displays the employees form
5. The administrator clicks the “All employees” button within the ribbon bar
6. The system enables full access to view the form
7. The administrator can print, save or attach the file vie email.

Exceptional flow

E1: Administrator logs out

4. The system will close, and the login form will reappear back
5. The employee either close the application or logs back in
6. If the employee doesn’t close the form, the use case returns to position 2, else it exits.

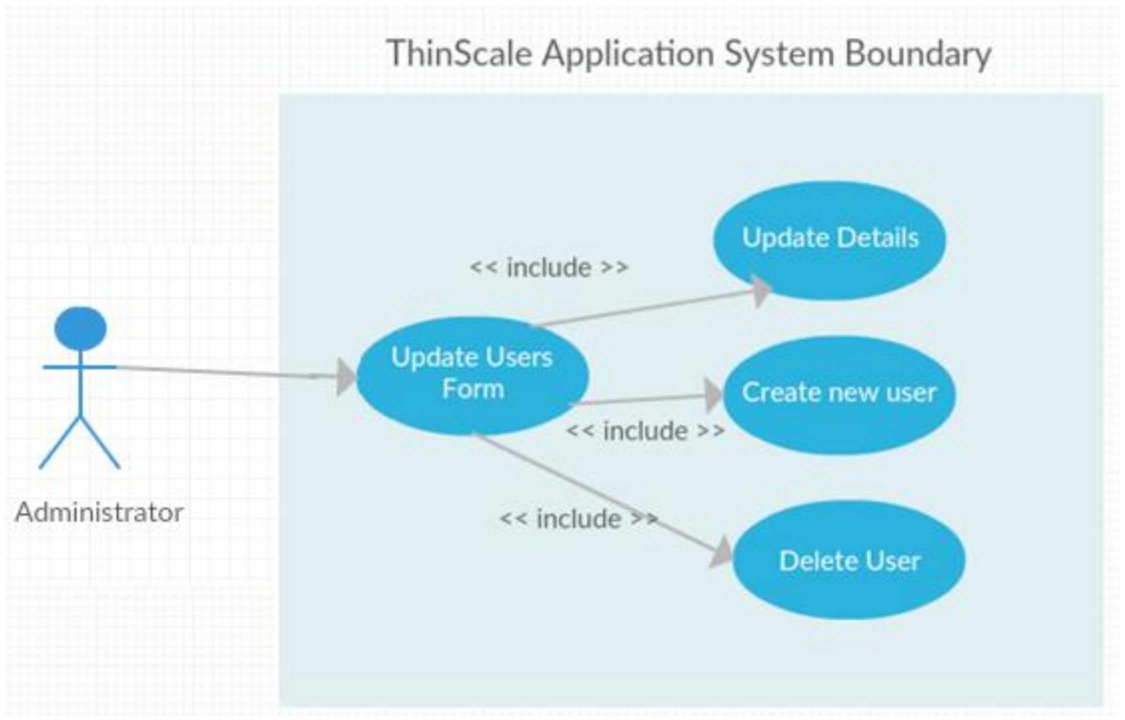
Termination

The system presents the login form.

Post condition

The system goes into a wait state

2.1.1 Requirement 14 – Update Users Form



Description & Priority

ThinScale Administrators can view the employee's details, modified and delete the user. This is a priority 3 for the administrators to have an overview about employees' information.

Use Case – 7. Update Users Form

Scope

The scope of this use case is to allow the administrators to view all the information about the employee already register, change their password, and or delete them or create a new one.

Description

This use case describes how an administrator can create, modify or delete, through a simple form, the employee's login details held in the database.

Flow Description

Precondition

The system loads the employee’s login details that is already saved into the application using a specific form.

Activation

This use case starts when an administrator logs in and clicks the Administration section and clicks “Update Users”

Main flow

8. The system displays the form
9. The administrator clicks the Update Users”” button within the ribbon bar
10. The system enables full access to view the form
11. The administrator can modify, create or delete a user

Exceptional flow

E1: Administrator logs out

7. The system will close, and the login form will reappear back
8. The employee either close the application or logs back in
9. If the employee doesn’t close the form, the use case returns to position 2, else it exits.

Termination

The system presents the login form.

Post condition

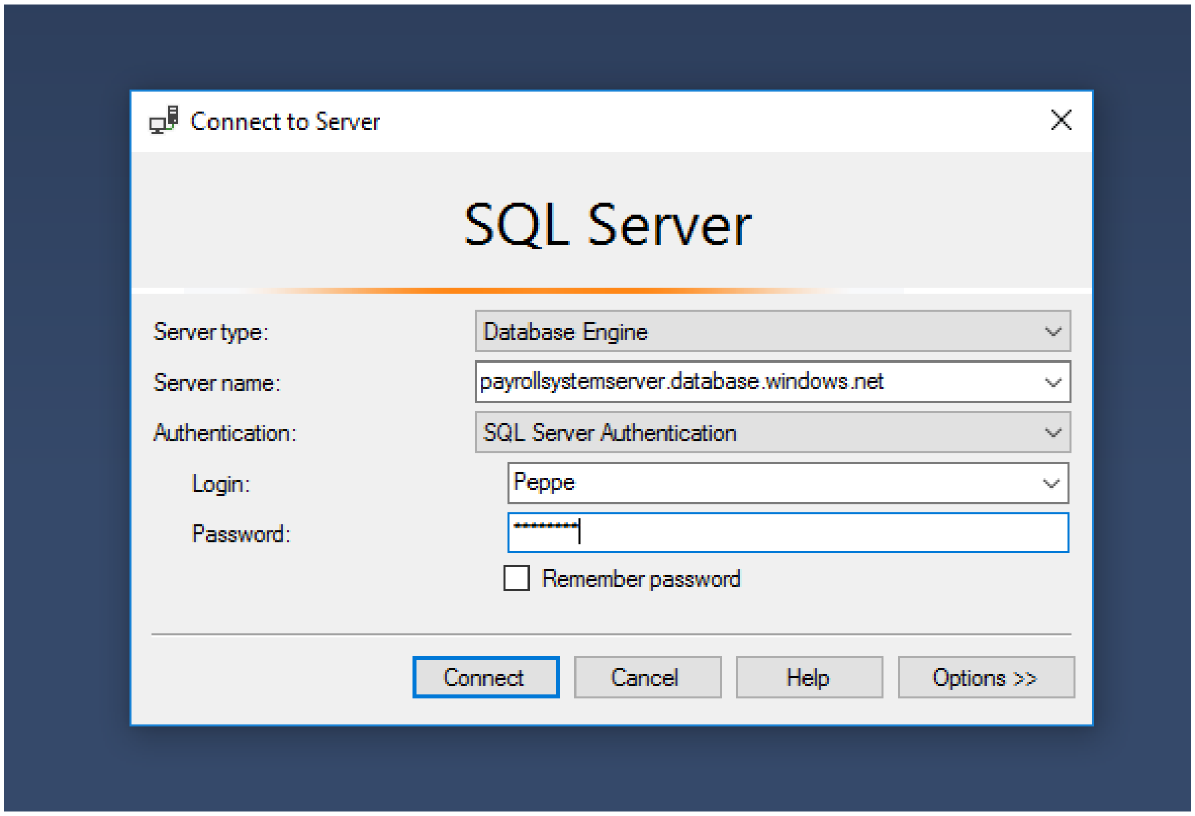
The system goes into a wait state

2.1.1 Requirement 15 – Notes

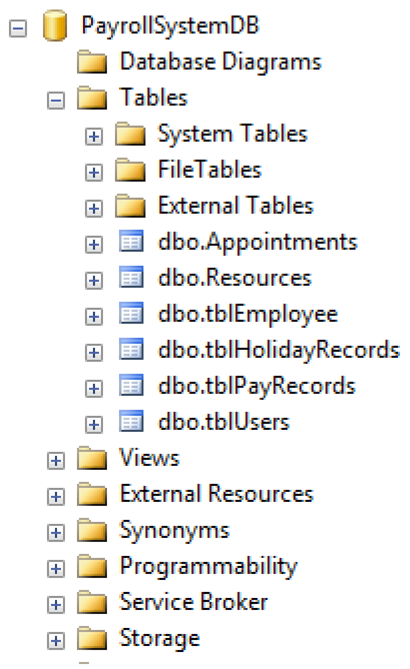
Requirements 1 to 9 are also applicable to Administrators.

2.1.2 Data requirements

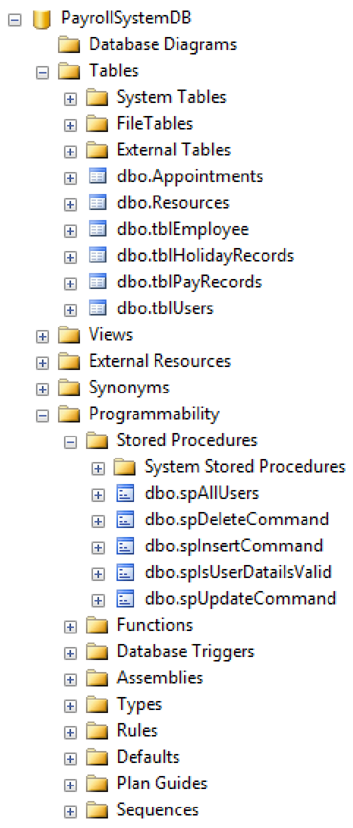
This section describes all the data requirements needed to interact with the application. Without a database attached, Admins won't be able to see the list of the employee or to compute the payments based on the number of hours an employee have worked. As a matter of fact, an application without a database won't be an application. As mentioned in the technology section the database is an Azure Microsoft MYSQL Database. It is a Relational Database which uses store procedures to retrieve the data in a more secure and easy fashion. There are 6 main tables stored in one PayrollSystemDB database and 5 stored procedures as highlighted from the screen below.



Server type: Database Engine
Server name: payrollsystemserver.database.windows.net
Authentication: SQL Server Authentication
Login: Peppe
Password: ********
☐ Remember password

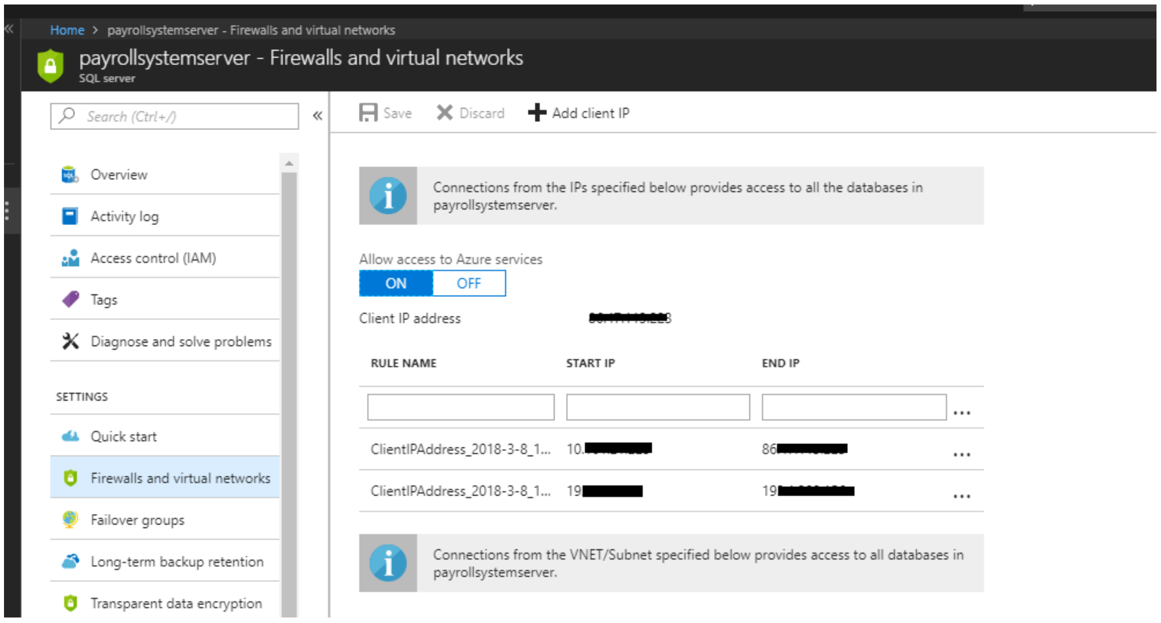


- PayrollSystemDB
  - Database Diagrams
  - Tables
    - System Tables
    - FileTables
    - External Tables
    - dbo.Appointments
    - dbo.Resources
    - dbo.tblEmployee
    - dbo.tblHolidayRecords
    - dbo.tblPayRecords
    - dbo.tblUsers
  - Views
  - External Resources
  - Synonyms
  - Programmability
  - Service Broker
  - Storage



- PayrollSystemDB
  - Database Diagrams
  - Tables
    - System Tables
    - FileTables
    - External Tables
    - dbo.Appointments
    - dbo.Resources
    - dbo.tblEmployee
    - dbo.tblHolidayRecords
    - dbo.tblPayRecords
    - dbo.tblUsers
  - Views
  - External Resources
  - Synonyms
  - Programmability
    - Stored Procedures
      - System Stored Procedures
      - dbo.spAllUsers
      - dbo.spDeleteCommand
      - dbo.spInsertCommand
      - dbo.spIsUserDatailsValid
      - dbo.spUpdateCommand
    - Functions
    - Database Triggers
    - Assemblies
    - Types
    - Rules
    - Defaults
    - Plan Guides
    - Sequences

To add security on top of Azure, the database is only accessible within the IP or IP Range I specified.



Home > payrollsystemserver - Firewalls and virtual networks
payrollsystemserver - Firewalls and virtual networks
SQL server
Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
SETTINGS
Quick start
Firewalls and virtual networks
Failover groups
Long-term backup retention
Transparent data encryption
Save
Discard
Add client IP

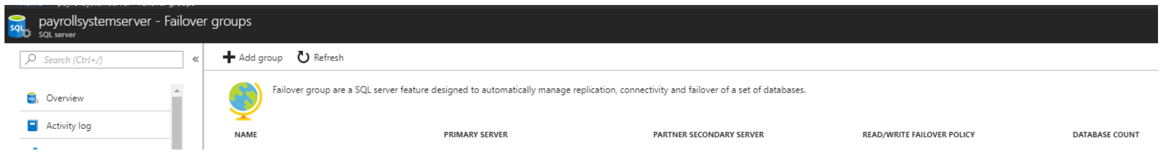
Connections from the IPs specified below provides access to all the databases in payrollsystemserver.

Allow access to Azure services [✓ ON] OFF

Client IP address [redacted]

RULE NAME	START IP	END IP
ClientIPAddress_2018-3-8_1...	10.[redacted]	86[redacted]
ClientIPAddress_2018-3-8_1...	19[redacted]	19[redacted]

Connections from the VNET/Subnet specified below provides access to all databases in payrollsystemserver.



Failover group are a SQL server feature designed to automatically manage replication, connectivity and failover of a set of databases.

NAME	PRIMARY SERVER	PARTNER SECONDARY SERVER	READ/WRITE FAILOVER POLICY	DATABASE COUNT

2.1.3 Non-Functional requirements

Describes any non-functional requirements required by the system. Examples are provided below.

Performance/Response time requirement

The time required installing the application, register, login in and use it will be less than 5 minutes. The application will use what is called a “Next-next-finish” installation with little to no interaction. Once registered and logged in, this operation might take up to 2/3 seconds due to the encryption and hashing of the password; navigation between forms will be almost instantaneously. The admin might see some latency depending on the size of the employees’ list that has to be retrieved from the database. These response times may have to be re-evaluated retrospectively if the employees base grows exponentially and refinement of the database may be needed, the current set up of the database is for a small to a medium business. Additionally, when the database will grow failover rules will be applied as well. (See Data Requirements). Please also see Customer Testing for more precise Metrix.

Availability requirement

ThinScale application can be accessed any time by employees and administrators working internally, and externally through VPN or RDP. There will be little to no maintenance required, only if important updates will be released, and being a windows application, any strange behaviours or crashes will be logged in the Event Viewer. All maintenance will be appropriately scheduled, and logs from the Event Viewer will be evaluated weekly. In case of strange warning message or error coming from the application, a tool called LOGGLY will be utilizing to get a better overview of what happened during those crashes. I will be using the free version for 30 days; hence information might be limited after this period. However, during trial, it will be possible to view the app performance, system behaviour and unusual activity across the stack. It is also possible to monitor metrics and KPI through its data analytics and reporting tool. A maintenance window will be agreed beforehand. When changes are being implemented a backup of the main database will be made and if the maintenance window is breeched the database will revert to the backup, with minimum to no data loss.

Operating System Requirements

The Application will be developed for Windows OS only, hence the employees or Admins must use at least a Windows 7 or higher machine, a .net Framework 4.6.1 or higher, Internet Explorer 8.0 or higher. Ideally, 1 to 2 Gb of Ram and at least 500 MB of free Hard Disk space. Due to the data stored separate from the application, SSD disk or SATA are equally accepted. A network connection is also required in the situation where email and browser form want to be utilize. There is no defined requirement for internet speed, ideally more than 10Mb/s.

Security requirement

A main concern on how to prevent SQL Injection, brute forcing a login entry and protect the database is a must. Employees and Administrators passwords will be encrypted, before being stored on the SQL database and hashed. This is to stop anyone who has access to the database or might gain unauthorised access to the database from having easy, plain text user's passwords. Passwords will be encrypted with a PBKDF (**Password-Based Key Derivation Function**) along with an added salt value and IV (**initialization vector**). Since Salt and IV must be the same between the encryption and decryption of a given string, the salt and IV will be prepended to the cipher text upon encryption and extracted from it again to perform the decryption. The result of this is that encrypting the exact same plaintext with the exact same password gives an entirely different cipher text' result each time. The salt should be randomly generated for every employee and then assigned to that employee, meaning that every employee will have a different salt even when using the same password. For querying data back and forth between the application and database a selection of stored procedure will be used to eliminate the need for a direct call to a specific table or row in the back-end source code. Testing in a UAT environment will be done locally on three virtual machines, with three different Operating system being Windows 7 Pro, Windows 8 Pro and Widows 10 Pro. (See VMs Testing).

Reliability requirement

Reliability testing will be measured against the software-testing pre-phase. To determine whether the application meets its performance a metric will be allocated. Metrics such as response time, throughput, CPU or memory utilization will be used to benchmark the results got in test and compare them with the one got in production where multiple users will be accessing the database at the same time. A failure will be categorized as a system failure with higher priority. For the first month of launch metrics won't be utilized as issues, latency and slowness are to be expected. After this period the result will be compared and if everything is not as expected the required updates or modification of the source code will be performed.

Maintainability requirement

For security purposes the code won't be leaving the ThinScale Development department and it will be only accessible internally through a secure connection and if a VPN is used when working remotely the use of a certificate on the developer machine will be enforced. In the UAT environment the code will be fully commented, fully committed using SVN tool, and a detailed documentation will be created containing notarized information as how the application works and how it was developed. At the beginning the scope of the application is limited of maintaining existing functionality and all further functionalities are yet to be discussed on.

Portability requirement

Portability won’t be huge part of this application because, as discussed in the operating system requirements, it will be only running on a windows machine. The website, written in PHP, HTML, CSS, which are web standards are inheritable portable, will be accessible from anywhere and from any web browsers, but you will be able to download the setup.exe file only if accessed through a desktop and after authentication and validation has been performed. The application will be installed locally on the machine, but it will be accessible from anywhere if a VPN or RDP, are in place on the local machine.

Extendibility requirement

There are a few key areas that are currently out of scope of the project but would be areas of interest and may be implemented in the future releases. Ideally, if the application will end up successful, a code conversion for different Operating Systems could be applied. Designing an app, which is fully portable from Linux, Ubuntu and MacOS will be the end goal. I already researched some tool and Xamarin might be utilize for this purpose. Xamarin can be integrated with Visual Studio, and delivers native Android, iOS, and Windows apps with a single shared .NET code base.

MFA (Multi factor authentication) is also a feature on the list. Adding multiple layers of security is always good in terms of software development. I will avoid SMS 2FA, based also to a meeting I had about OWASP security risk, where a security expert exposed the vulnerability and the potential intercept of that system. If implementable I will be going for Google Authenticator, Authy or Yubico Authenticator.

Another tool that might be good to implement, also for security, is the option to login using a domain account via a smart card and a smart card reader. Rather than make the users using a normal type 1 factor password, the idea to give MFA with a card reader will increase security and will lower down potential breaches coming from external sources.

Reusability requirement

Reusability of code will be necessary, as part of the code, to create the forms, will be reutilized. Different behaviours and look are applied but the structure behind the functions will be almost the same for every single form. For the administrator's point of view, the SQL database rows and columns will be always the same and it might change only in case they want to add or remove columns or rows that they don't need anymore. If a new schema is required, the support team or the proper engineer will perform the upgrade for them, working externally on the Azure Database backup or failover, so no interruption of the end client will be required.

2.1.4 User requirements

ThinScale Application Suite is designed to offer ThinScale Company employees a comprehensive, clear and easy way of checking for their holidays, to book a new one using a scheduler form, surf the Internet freely within the same application, send emails using an internal SMTP server, and it also enables the employees to review reports utilizing different charts'. For the Administrators, point of view, they will be able to organize create, delete or update employees using specific form and to use different controls to calculate, generate and print payslip. Like the employees, Admins will be able to surf the internet and send emails as well, they will be able to reset employee's passwords and create new entries. The 2 user types are:

1. **<u>Employees</u>** – The employees have their own unique log in which enables them to:
    - **Register to the application** – Register to the application with unique credentials using a form,
    - **Access the application** – Login validation with unique credentials validated,
    - **View the holiday scheduler control** – GUI displaying the days and or appointments already booked,
    - **Select a date and book a holiday/meeting/ tasks** – Select date/time and reserve the holiday at chosen time and date.
    - **View charts** – They can navigate through different charts to check holiday availability.
    - **Surf the internet**– They can use an internal web browser to surf the internet
    - **Send Emails** – They can use an internal SMTP client to send emails (GMAIL only now).
    - **Desktop**: the employee needs a desktop to install and use the application
    - **Windows 7:** at least windows 7 must run on the machine.

- **Internet access:** to retrieve the installation file from Fun Box, emails and internet surfing.
- **1 GB free space:** to ensure higher performance

2. **<u>Administrator</u>** – Unique log in granting administrator rights to;
    - **Register to the application** – Register to the application with unique credentials using a form,
    - **Add Employees** – Add employees into the system
    - **Update Employees** - Update employees
    - **Delete Employees** - Delete employees
    - **Check the holiday scheduler control**– Same as employees
    - **View / Delete holiday**- Review the dates from the employees or delete them if not allowed to be booked
    - **Create/ View Payments form –** Generate payslip and view reports
    - **View charts** – They can navigate through different charts to
    - **Surf the internet**– They can use an internal web browser to surf the internet
    - **Send Emails** – They can use an internal SMTP client to send emails (GMAIL only now).
    - **Desktop**: the employee needs a desktop to install and use the application
    - **Windows 7:** at least windows 7 must run on the machine.
    - **Internet access:** to retrieve the installation file, emails and internet.
    - **1 GB free space:** to ensure higher performance
    -

One of the benefits of the ThinScale Application Suite is the centralized control. With just one software, Admin can be monitoring employee's holidays, their tasks, events and payments. Different forms allow employees to see the daily, monthly, or yearly calendar through a scheduler, giving them more flexibility and general control. It is the ease of use and simplicity that this application offers to both employees and Administrators that make this product unique.

2.1.5 Environmental requirements

This section explains the overall environment used to develop ThinScale application.

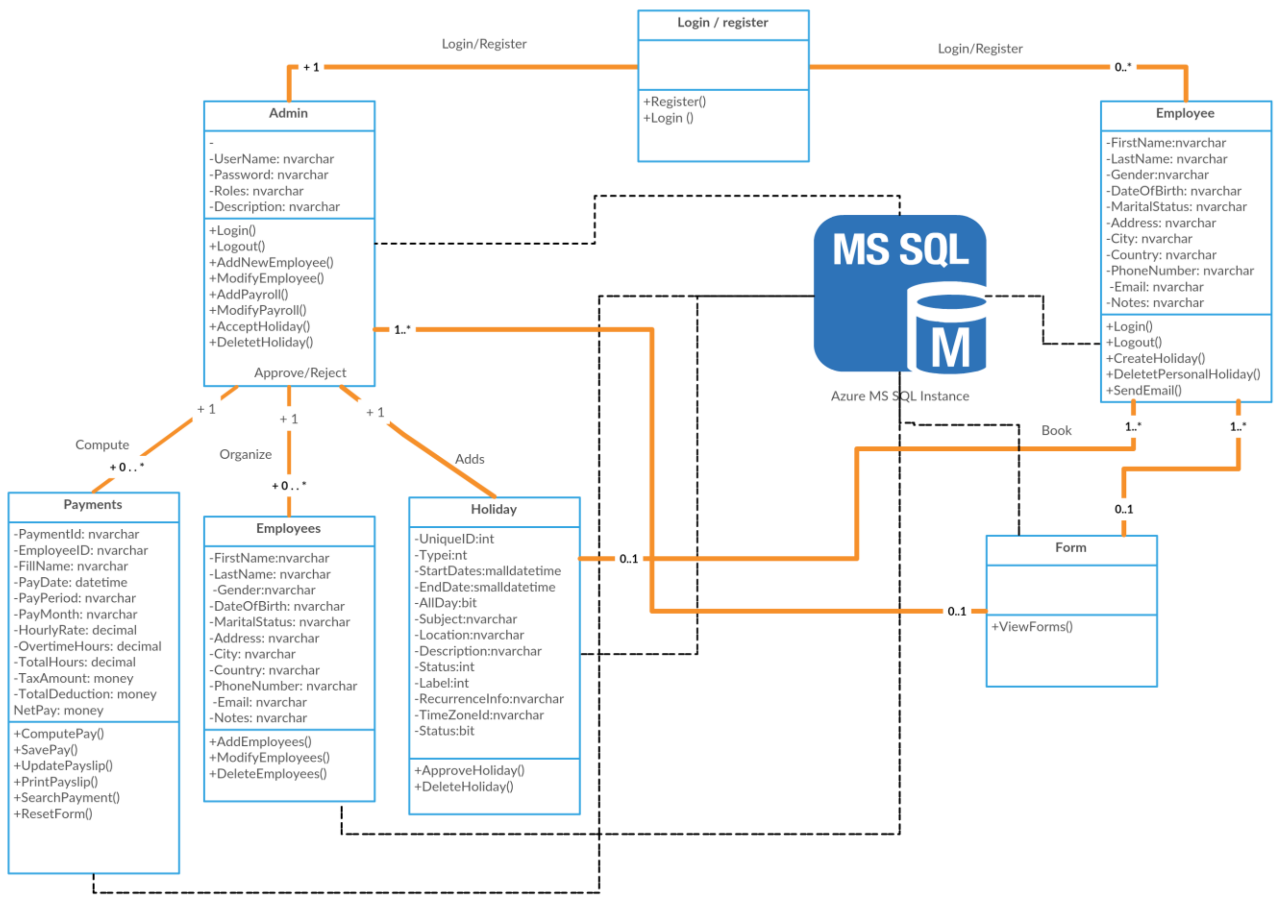
- NUC intel i7 inside 16GB DDR4 RAM, 500GB SSD.
- Windows 10 SDK 10.0.10586.212.
- VMware vSphere Web Client to create the virtual machines.
- 3 VMs with Windows 7/8/10 for testing purpose.
- VS_RemoteTools.exe debug tools to attach the PID over the VMs
- Visual Studio 2017 to code the application in C#.
- DevExpress license to give the modern look.
- Windows Certificate Manager Signtool to sign the application.
- TortoiseSVN for committing the code.
- Fun Box to store the setup.exe file.
- WordPress for the website

2.1.6 Usability requirements

The only way an employee can interact with the application is by first requesting the download link from the web page. Then by installing the application on his/her personal machine, register with valid credentials and then log in. No prior training is required but a small guide will be provided for general information in relation on how install the software. (See appendix)

2.2 Design and Architecture

The application, as mentioned in the section above, is built using Visual Studio and C#. The structure of the ThinScale application is centred on a Microsoft Azure Microsoft SQL Server database to store and return fundamental information to admins and employees. Using Creately.com it was possible to design a Class diagram (below) to outline the structure of the application and how data is shared and returned to all employees and admins.


Login / register
+Register()
+Login ()
Login/Register
Login/Register
+ 1
0..*
Admin
-
-UserName: nvarchar
-Password: nvarchar
-Roles: nvarchar
-Description: nvarchar
+Login()
+Logout()
+AddNewEmployee()
+ModifyEmployee()
+AddPayroll()
+ModifyPayroll()
+AcceptHoliday()
+DeletetHoliday()
Approve/Reject
Employee
-FirstName:nvarchar
-LastName: nvarchar
-Gender:nvarchar
-DateOfBirth: nvarchar
-MaritalStatus: nvarchar
-Address: nvarchar
-City: nvarchar
-Country: nvarchar
-PhoneNumber: nvarchar
-Email: nvarchar
-Notes: nvarchar
+Login()
+Logout()
+CreateHoliday()
+DeletetPersonalHoliday()
+SendEmail()
MS SQL
M
Azure MS SQL Instance
1..*
Book
1..*
1..*
0..1
+ 1
+ 1
+ 1
Compute
Organize
Adds
+ 0 . . *
+ 0 . . *
Payments
-PaymentId: nvarchar
-EmployeeID: nvarchar
-FillName: nvarchar
-PayDate: datetime
-PayPeriod: nvarchar
-PayMonth: nvarchar
-HourlyRate: decimal
-OvertimeHours: decimal
-TotalHours: decimal
-TaxAmount: money
-TotalDeduction: money
NetPay: money
+ComputePay()
+SavePay()
+UpdatePayslip()
+PrintPayslip()
+SearchPayment()
+ResetForm()
Employees
-FirstName:nvarchar
-LastName: nvarchar
-Gender:nvarchar
-DateOfBirth: nvarchar
-MaritalStatus: nvarchar
-Address: nvarchar
-City: nvarchar
-Country: nvarchar
-PhoneNumber: nvarchar
-Email: nvarchar
-Notes: nvarchar
+AddEmployees()
+ModifyEmployees()
+DeleteEmployees()
Holiday
-UniqueID:int
-Typei:nt
-StartDates:malldatetime
-EndDate:smalldatetime
-AllDay:bit
-Subject:nvarchar
-Location:nvarchar
-Description:nvarchar
-Status:int
-Label:int
-RecurrenceInfo:nvarchar
-TimeZoneId:nvarchar
-Status:bit
+ApproveHoliday()
+DeleteHoliday()
0..1
0..1
Form
+ViewForms()

2.3 Implementation

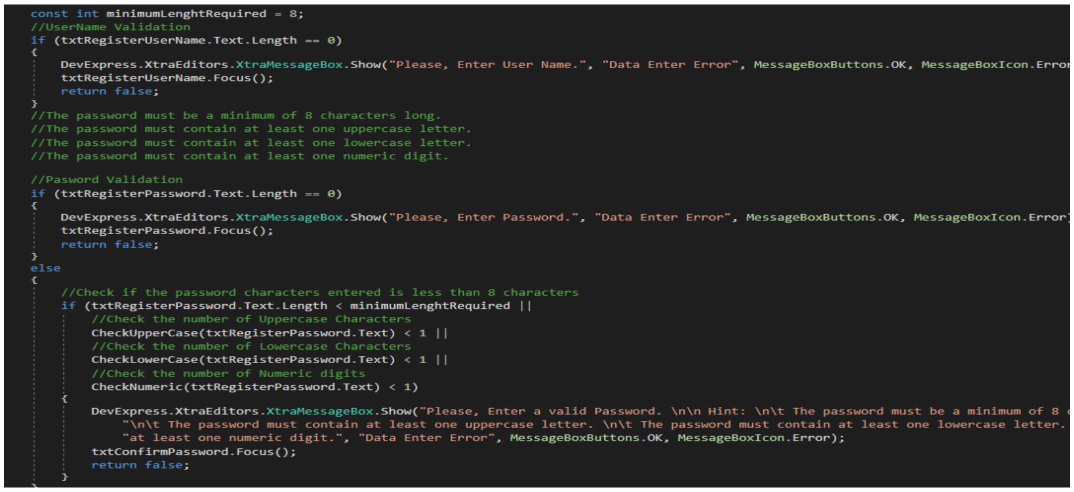
This section describes in detail the technology used to develop ThinScale application.

2.3.1 Technology Overview

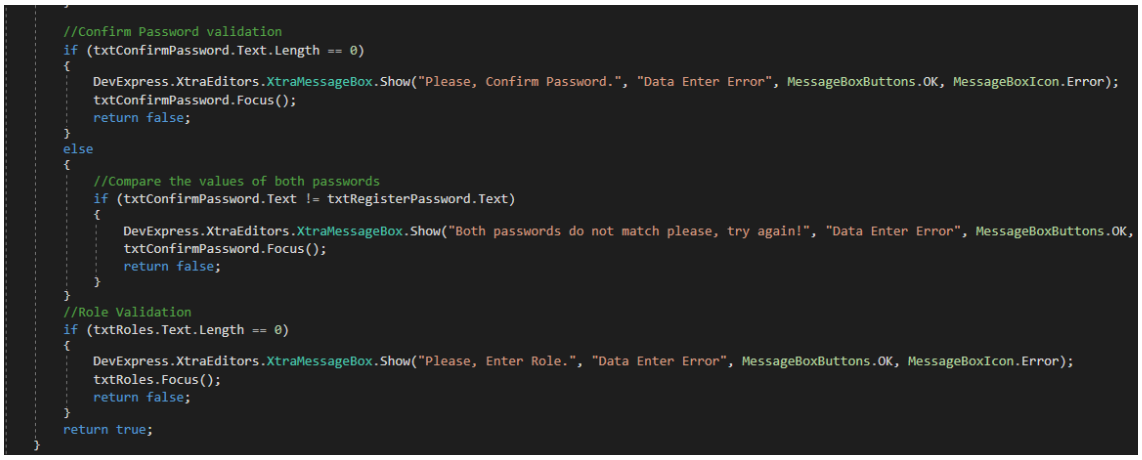
Being exposed every day to Visual Studio and C# for work purposes, the choice of developing an application was easy. The application was built with few functionalities in mind but as soon as I saw the potential I started adding more based on the time and the resources I had in my hand. The following code snippets and screenshots are coming from the actual application I have developed.

2.3.2 Security, Password, Validation

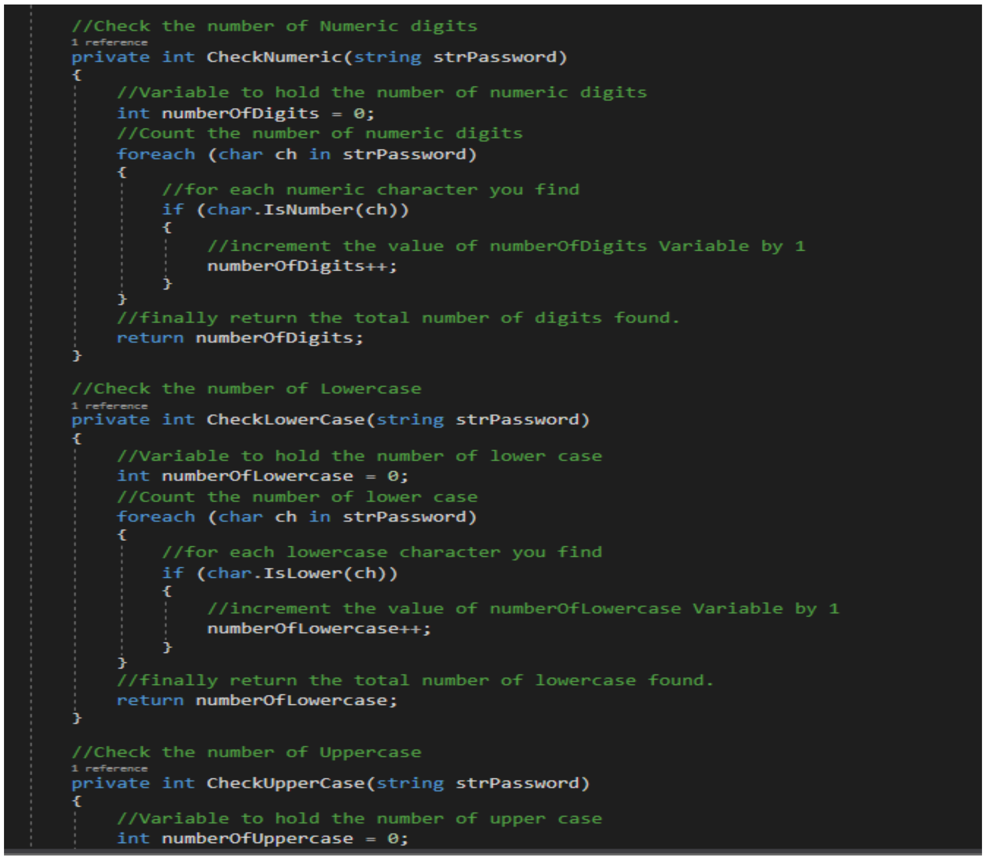
To ensure password strength to protect against hacker or brute forcing, few requirements were enforced, like minimum length, characters, symbols and numbers. Encryption and decryption have been also implemented. Hashing and salting are implemented too, to ensure that, if a non-authorized person will ever get access to the database, stored on Azure, will never see the password stored in plain text. Following, in the image below, you will see a snippet implemented that checks for validation.



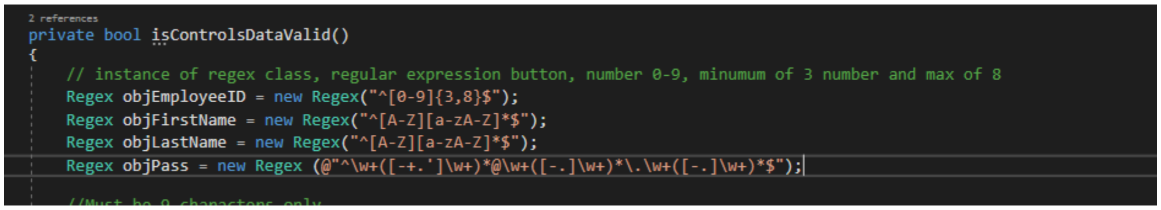
```
const int minimumLenghtRequired = 8;
//UserName Validation
if (txtRegisterUserName.Text.Length == 0)
{
    DevExpress.XtraEditors.XtraMessageBox.Show("Please, Enter User Name.", "Data Enter Error", MessageBoxButtons.OK, MessageBoxIcon.Error
    txtRegisterUserName.Focus();
    return false;
}
//The password must be a minimum of 8 characters long.
//The password must contain at least one uppercase letter.
//The password must contain at least one lowercase letter.
//The password must contain at least one numeric digit.

//Pasword Validation
if (txtRegisterPassword.Text.Length == 0)
{
    DevExpress.XtraEditors.XtraMessageBox.Show("Please, Enter Password.", "Data Enter Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
    txtRegisterPassword.Focus();
    return false;
}
else
{
    //Check if the password characters entered is less than 8 characters
    if (txtRegisterPassword.Text.Length < minimumLenghtRequired ||
        //Check the number of Uppercase Characters
        CheckUpperCase(txtRegisterPassword.Text) < 1 ||
        //Check the number of Lowercase Characters
        CheckLowerCase(txtRegisterPassword.Text) < 1 ||
        //Check the number of Numeric digits
        CheckNumeric(txtRegisterPassword.Text) < 1)
    {
        DevExpress.XtraEditors.XtraMessageBox.Show("Please, Enter a valid Password. \n\n Hint: \n\t The password must be a minimum of 8
            "\n\t The password must contain at least one uppercase letter. \n\t The password must contain at least one lowercase letter.
            "at least one numeric digit.", "Data Enter Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        txtConfirmPassword.Focus();
        return false;
    }
```



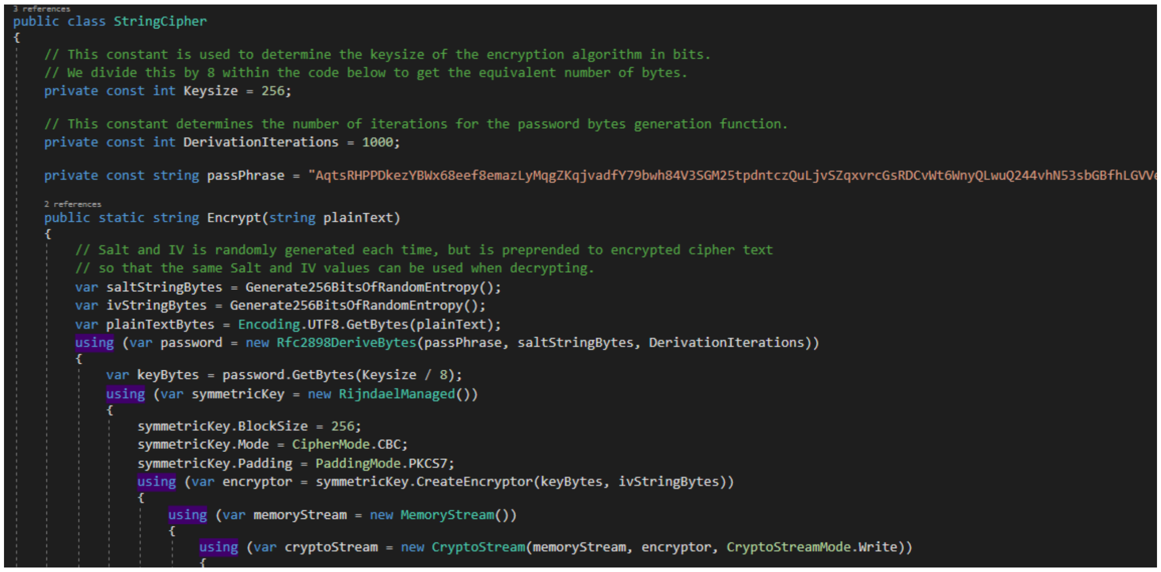
```
    //Confirm Password validation
    if (txtConfirmPassword.Text.Length == 0)
    {
        DevExpress.XtraEditors.XtraMessageBox.Show("Please, Confirm Password.", "Data Enter Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        txtConfirmPassword.Focus();
        return false;
    }
    else
    {
        //Compare the values of both passwords
        if (txtConfirmPassword.Text != txtRegisterPassword.Text)
        {
            DevExpress.XtraEditors.XtraMessageBox.Show("Both passwords do not match please, try again!", "Data Enter Error", MessageBoxButtons.OK,
            txtConfirmPassword.Focus();
            return false;
        }
    }
    //Role Validation
    if (txtRoles.Text.Length == 0)
    {
        DevExpress.XtraEditors.XtraMessageBox.Show("Please, Enter Role.", "Data Enter Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        txtRoles.Focus();
        return false;
    }
    return true;
}
```



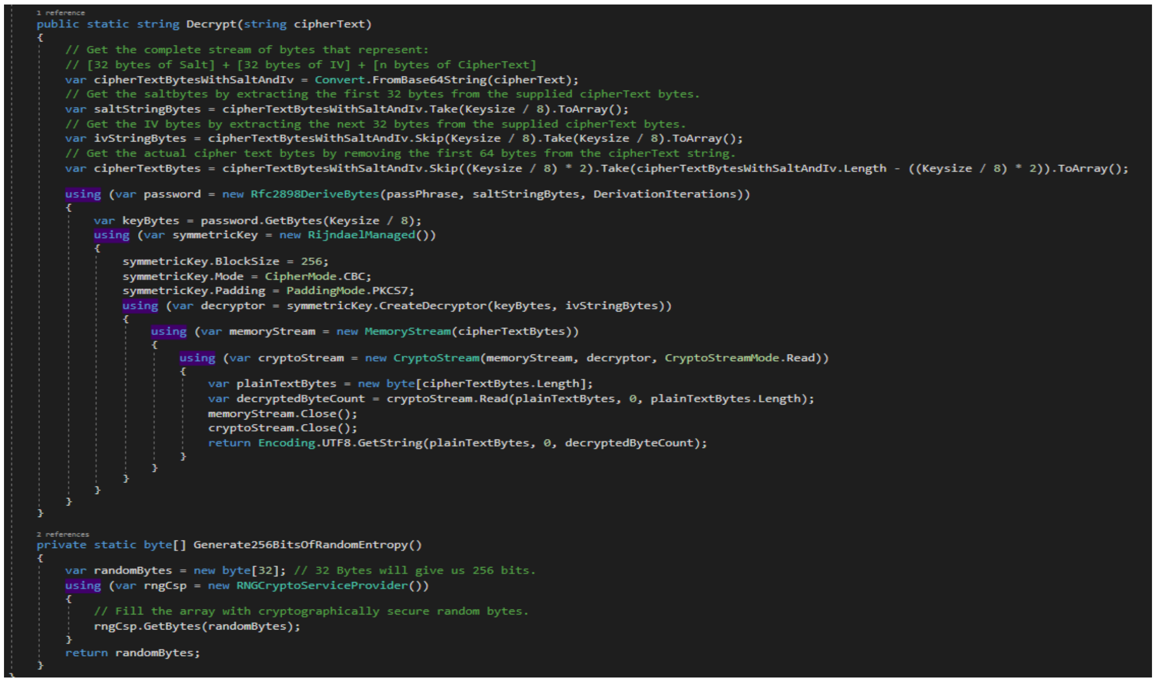
```
//Check the number of Numeric digits
1 reference
private int CheckNumeric(string strPassword)
{
    //Variable to hold the number of numeric digits
    int numberOfDigits = 0;
    //Count the number of numeric digits
    foreach (char ch in strPassword)
    {
        //for each numeric character you find
        if (char.IsNumber(ch))
        {
            //increment the value of numberOfDigits Variable by 1
            numberOfDigits++;
        }
    }
    //finally return the total number of digits found.
    return numberOfDigits;
}

//Check the number of Lowercase
1 reference
private int CheckLowerCase(string strPassword)
{
    //Variable to hold the number of lower case
    int numberOfLowercase = 0;
    //Count the number of lower case
    foreach (char ch in strPassword)
    {
        //for each lowercase character you find
        if (char.IsLower(ch))
        {
            //increment the value of numberOfLowercase Variable by 1
            numberOfLowercase++;
        }
    }
    //finally return the total number of lowercase found.
    return numberOfLowercase;
}

//Check the number of Uppercase
1 reference
private int CheckUpperCase(string strPassword)
{
    //Variable to hold the number of upper case
    int numberOfUppercase = 0;
```



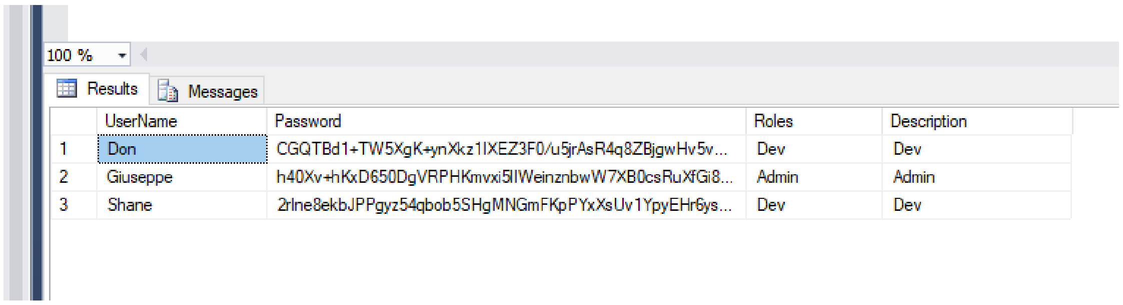
```
2 references
private bool isControlsDataValid()
{
    // instance of regex class, regular expression button, number 0-9, minumum of 3 number and max of 8
    Regex objEmployeeID = new Regex("^[0-9]{3,8}$");
    Regex objFirstName = new Regex("^[A-Z][a-zA-Z]*$");
    Regex objLastName = new Regex("^[A-Z][a-zA-Z]*$");
    Regex objPass = new Regex (@"^\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*$");

    //Must be 9 characters only
```



```
3 references
public class StringCipher
{
    // This constant is used to determine the keysize of the encryption algorithm in bits.
    // We divide this by 8 within the code below to get the equivalent number of bytes.
    private const int Keysize = 256;

    // This constant determines the number of iterations for the password bytes generation function.
    private const int DerivationIterations = 1000;

    private const string passPhrase = "AqtsRHPPDkezYBWx68eef8emazLyMqgZKqjvadfY79bwh84V3SGM25tpdntczQuLjvSZqxvrcGsRDCvWt6WnyQLwuQ244vhN53sbGBfhLGVVe

    2 references
    public static string Encrypt(string plainText)
    {
        // Salt and IV is randomly generated each time, but is preprended to encrypted cipher text
        // so that the same Salt and IV values can be used when decrypting.
        var saltStringBytes = Generate256BitsOfRandomEntropy();
        var ivStringBytes = Generate256BitsOfRandomEntropy();
        var plainTextBytes = Encoding.UTF8.GetBytes(plainText);
        using (var password = new Rfc2898DeriveBytes(passPhrase, saltStringBytes, DerivationIterations))
        {
            var keyBytes = password.GetBytes(Keysize / 8);
            using (var symmetricKey = new RijndaelManaged())
            {
                symmetricKey.BlockSize = 256;
                symmetricKey.Mode = CipherMode.CBC;
                symmetricKey.Padding = PaddingMode.PKCS7;
                using (var encryptor = symmetricKey.CreateEncryptor(keyBytes, ivStringBytes))
                {
                    using (var memoryStream = new MemoryStream())
                    {
                        using (var cryptoStream = new CryptoStream(memoryStream, encryptor, CryptoStreamMode.Write))
                        {
```



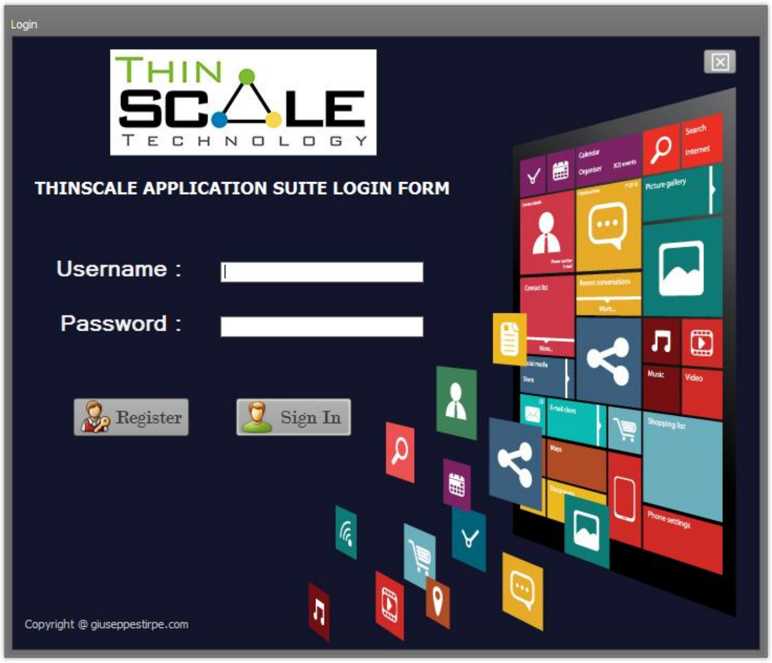
```
1 reference
public static string Decrypt(string cipherText)
{
    // Get the complete stream of bytes that represent:
    // [32 bytes of Salt] + [32 bytes of IV] + [n bytes of CipherText]
    var cipherTextBytesWithSaltAndIv = Convert.FromBase64String(cipherText);
    // Get the saltbytes by extracting the first 32 bytes from the supplied cipherText bytes.
    var saltStringBytes = cipherTextBytesWithSaltAndIv.Take(Keysize / 8).ToArray();
    // Get the IV bytes by extracting the next 32 bytes from the supplied cipherText bytes.
    var ivStringBytes = cipherTextBytesWithSaltAndIv.Skip(Keysize / 8).Take(Keysize / 8).ToArray();
    // Get the actual cipher text bytes by removing the first 64 bytes from the cipherText string.
    var cipherTextBytes = cipherTextBytesWithSaltAndIv.Skip((Keysize / 8) * 2).Take(cipherTextBytesWithSaltAndIv.Length - ((Keysize / 8) * 2)).ToArray();

    using (var password = new Rfc2898DeriveBytes(passPhrase, saltStringBytes, DerivationIterations))
    {
        var keyBytes = password.GetBytes(Keysize / 8);
        using (var symmetricKey = new RijndaelManaged())
        {
            symmetricKey.BlockSize = 256;
            symmetricKey.Mode = CipherMode.CBC;
            symmetricKey.Padding = PaddingMode.PKCS7;
            using (var decryptor = symmetricKey.CreateDecryptor(keyBytes, ivStringBytes))
            {
                using (var memoryStream = new MemoryStream(cipherTextBytes))
                {
                    using (var cryptoStream = new CryptoStream(memoryStream, decryptor, CryptoStreamMode.Read))
                    {
                        var plainTextBytes = new byte[cipherTextBytes.Length];
                        var decryptedByteCount = cryptoStream.Read(plainTextBytes, 0, plainTextBytes.Length);
                        memoryStream.Close();
                        cryptoStream.Close();
                        return Encoding.UTF8.GetString(plainTextBytes, 0, decryptedByteCount);
                    }
                }
            }
        }
    }
}

2 references
private static byte[] Generate256BitsOfRandomEntropy()
{
    var randomBytes = new byte[32]; // 32 Bytes will give us 256 bits.
    using (var rngCsp = new RNGCryptoServiceProvider())
    {
        // Fill the array with cryptographically secure random bytes.
        rngCsp.GetBytes(randomBytes);
    }
    return randomBytes;
}
```

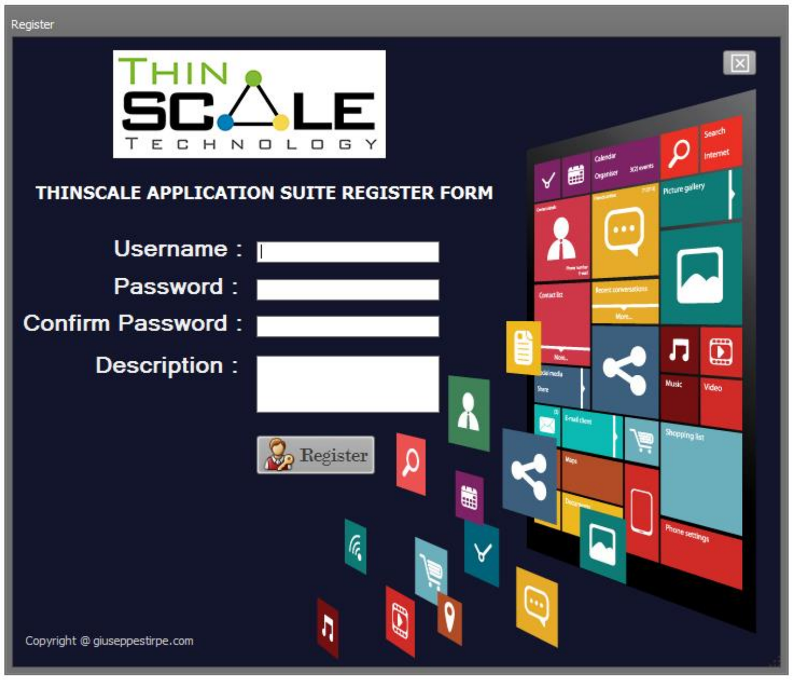


	UserName	Password	Roles	Description
1	Don	CGQTBd1+TW5XgK+ynXkz1IXEZ3F0/u5jrAsR4q8ZBjgwHv5v...	Dev	Dev
2	Giuseppe	h40Xv+hKxD650DgVRPHKmvxi5lIWeinznbwW7XB0csRuXfGi8...	Admin	Admin
3	Shane	2rlne8ekbJPPgyz54qbob5SHgMNGmFKpPYxXsUv1YpyEHr6ys...	Dev	Dev

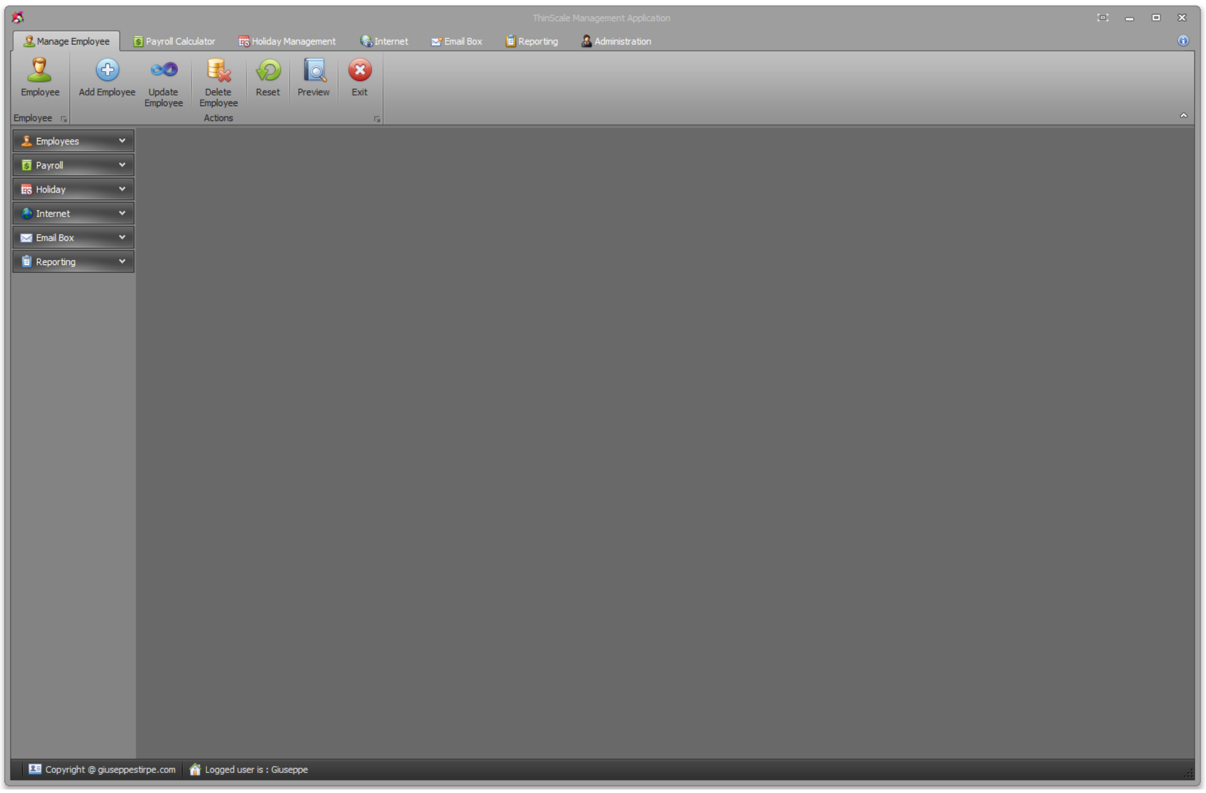
2.4 Graphical User Interface (GUI) Layout

This is the login and the registration form where employee or admin will land after launching the application previously installed.


Login
THIN SCALE TECHNOLOGY
THINSCALE APPLICATION SUITE LOGIN FORM
Username :
Password :
Register
Sign In
Copyright @ giuseppestirpe.com



Once signed in there will prompt with the main form divided into different sections. In the header there is a Ribbon Bar with multiple heading that employees (limited view) and Admins (full view) can use to navigate through different forms. Within the Ribbon bar there are buttons options which will show, depending on the user logged, different forms. The "Manage Employee" is where Admin will create, update or delete employees, the "Payroll Calculator" is where all the payments and payslips are created, "Holiday Management" form is used by the employees to book or cancel vacation, events, or appointments, "Internet" Form is where both employees and admins will go to browse the web, "Email Box" is for sending email and "Reporting" is used to have a detailed overview of holidays, payments and employees. The "Administration" form, only available to Admins, is where password resets, creation of new user, deleting or updating is performed. There is also a left Navigation Bar to help the interaction with the main form without reaching out to the main ribbon bar every time. An about icon on the far left will show general information about the application including the version number, copyrights and company name.



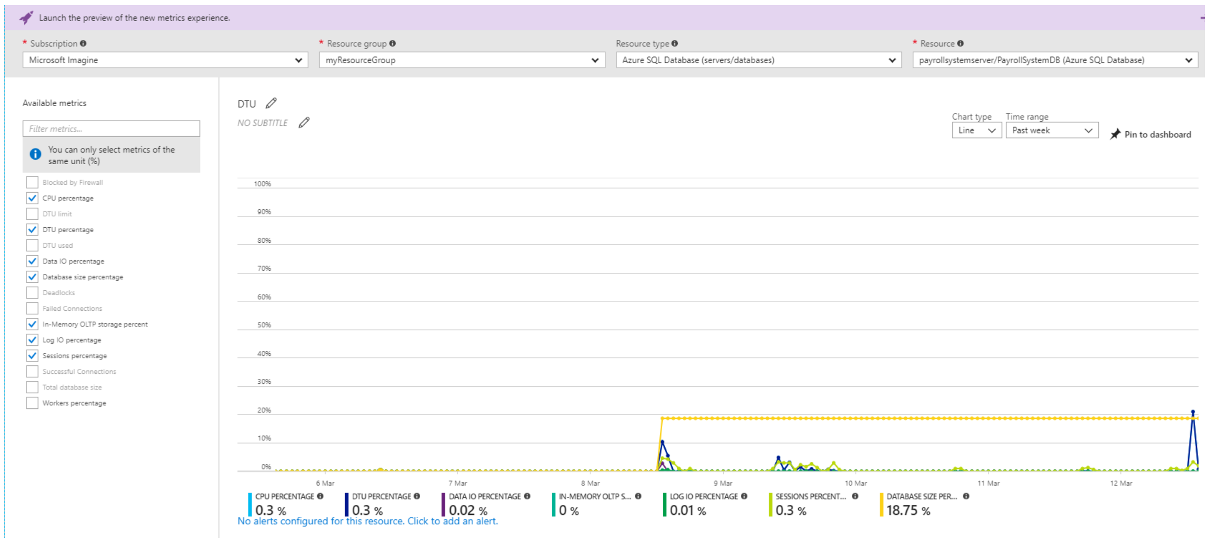
2.5 Testing

"Shift-left testing" is a process of conducting more constant testing though the development phase rather than just the end. I will be dedicating a large effort trying to detect and correct errors before they will get bigger and preventing the need to debug later in the development process. After creating and positioning few elements on the form, they will be tested for responsiveness and usability straight away. I will begin by making sure that all the controls, buttons, and text fields on the screen are working. I will be testing password encryption, registration, login form, and database validation in response to a forced error or invalid characters or symbols, using fuzz testing. The goal is to prevent the application from crashing and get the relative system errors or exceptions instead. I will be manually testing for SQL injection in the login form to check if by passing values that will be potentially dangerous ($password = 1' or '1' = '1; SELECT * FROM Users WHERE Username='1' OR '1' = '1' AND Password='1' OR '1' = '1'), I get any results or not. The code will be reviewed with the help of a tool called Clone Detective. Clone Detective allows analysing code duplication in the source code, which will lead to inconsistency and poorly factored implementation. For code managing, refactoring and commits, I will be using TortoiseSVN. TortoiseSVN is a free open source tool, released under the GNU General Public License and it is implemented as a Microsoft Shell extension that can get attached to Visual Studio by using third party plugin like VSTortoise. By using it, there will be a better code management, so in case of any extensive or unexpected bugs will be possible to revert to a previous version. Reverting means discarding all modifications previously made to one or more files, which were not committed. Committing means saving the last version of the code into the previously created TortoiseSVN directory, which will be then forged within Visual Studio. Thanks to TortoiseSVN you will never have the chance to lose important data or have conflicts when multiple people are working on the same code and at the same time. You will have the option to review the logs, comments or commits, if present, to revision graph, to branch folders, to merge source code and so on. After reviewing the code and every single element on the controls, I will continue testing the application with tools that perform analysis of the source code or byte code instead. These static tools will look for weakness or misuse that the compiler can't analyse. Because in our company we use only Microsoft licensed software, the tool that I am going to use for my application is Code Analysis Tool .NET (CAT.NET). CAT.NET will help identify security flaws within the code by scanning the binary or the assembly of my application. It will trace data flow among the statements, class, methods and assemblies. Once the scanning is finished, it will display the issues it finds in a list within Visual Studio output window, which I can use to jump directly to the source code that is found to have that problem. Once the code will be refactored and the first part of testing is complete, I will be giving the setup.exe to few colleagues to exploit the software and get important feedback. (See Customer Testing) They will be able to see bugs or behaviour I couldn't spot while working on the code. After a few weeks, I will collect the information back and based on their feedbacks, changes will be made. After the respective modification of the code, and documentation, the second part of testing will be performed. This time, automation testing will be used. As mentioned before fuzz test or fuzzing will be performed. Fuzzing is an automated software testing technique that involves providing invalid, unexpected, or random data as inputs to a computer program. The program is then monitored for exceptions such as crashes or failing built-in code assertions or for finding potential memory leaks. Rather than manually repeating all the previous assessments, I will use another software, called Ranorex, to test the app. Decided to go with Renorex, because based on the Microsoft .NET environment, the Ranorex testing framework provides an API for C# and VB.NET that integrates easily with Visual Studio. Automated testing increases the accuracy and saves resources and, most importantly, time. Once the code will be fixed, the application will be released again for the last part of testing. I will install the application on three different virtual machines with three different operating systems. I will monitor and study the behaviour of those and will, for the last time, stress test the application. If the results are as expected, then the application will finally be released and the final setup.exe file will be created. The last missing piece it will be hosting the file in the cloud, Fun Box Application, then make the employees aware of the web page where to retrieve the file, download it, install it and use it.

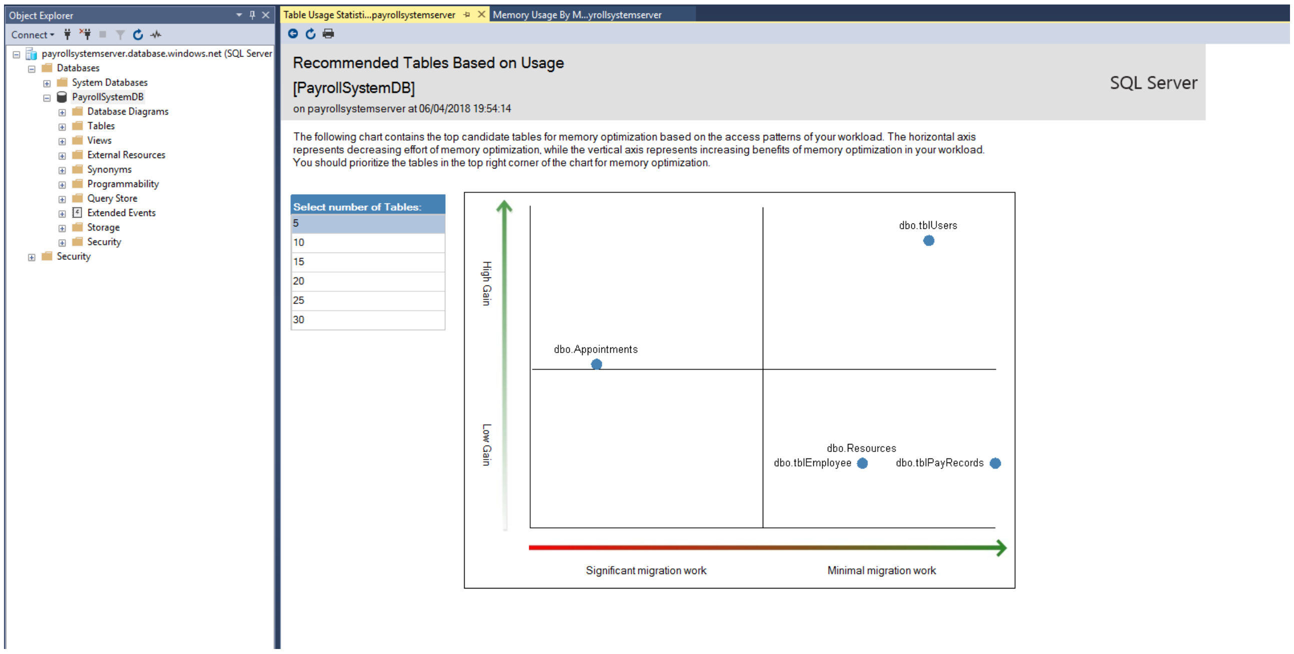
6.2 Customer testing

Started on the 15/03/2018, ThinScale Application Suite was ready to be deployed. I gave the setup.exe to 5 of my colleague. They tested it for a week or so while I was performing the same test on the 3 different virtual machines mentioned above. At the end of the first phase of testing I was benchmarking the results with theirs, and adjustment were made to resolve all issues that were found. This was one of the first testing phase before the final deployment. Following we can see metrics from the Azure SQL while multiple user where logging and using the application. Even under stress, I could see that the CPU utilization never spiked more than 5%. I will keep a closer look while more people will start using full time the application.

CPU information



Store Procedures usage



Recommended Tables Based on Usage

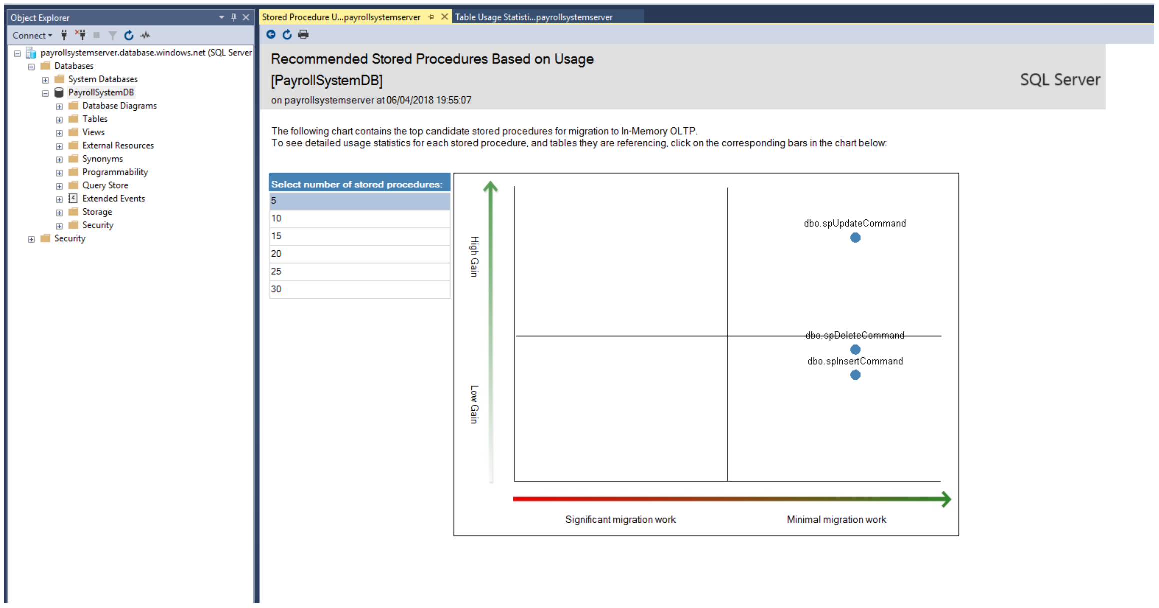
[PayrollSystemDB]

on payrollsystemserver at 06/04/2018 19:54:14

SQL Server

The following chart contains the top candidate tables for memory optimization based on the access patterns of your workload. The horizontal axis represents decreasing effort of memory optimization, while the vertical axis represents increasing benefits of memory optimization in your workload. You should prioritize the tables in the top right corner of the chart for memory optimization.

Select number of Tables:
5
10
15
20
25
30

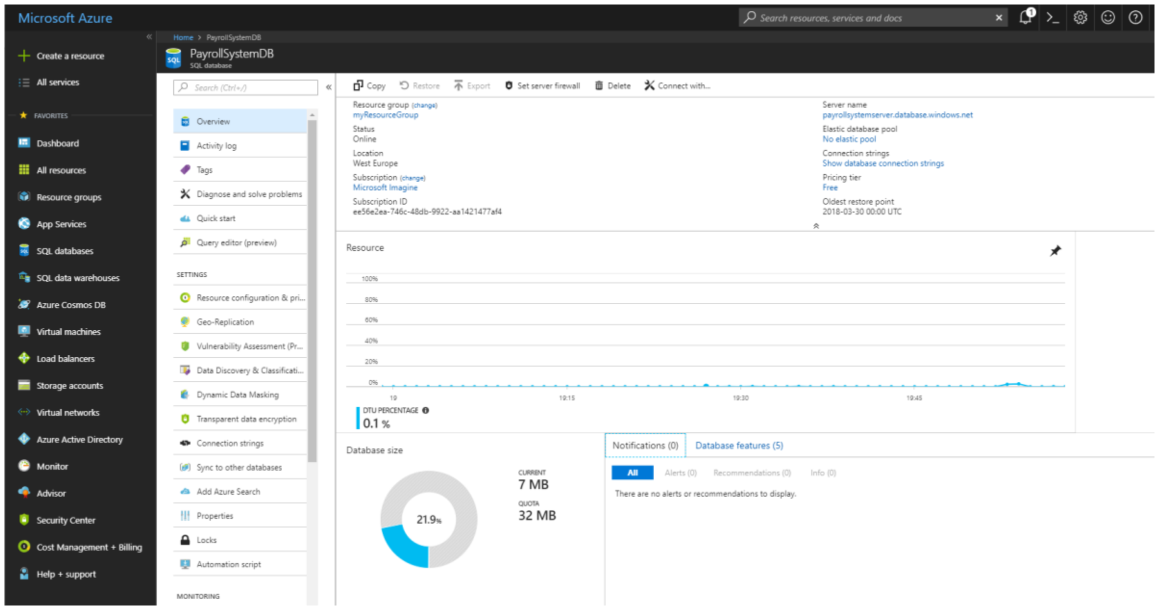


Recommended Stored Procedures Based on Usage
[PayrollSystemDB]
on payrollsystemserver at 06/04/2018 19:55:07

The following chart contains the top candidate stored procedures for migration to In-Memory OLTP.
To see detailed usage statistics for each stored procedure, and tables they are referencing, click on the corresponding bars in the chart below:

Select number of stored procedures:
5
10
15
20
25
30

Database size



Resource group (change)
myResourceGroup
Status
Online
Location
West Europe
Subscription (change)
Microsoft Imagine
Subscription ID
ee56e2ea-746c-48db-9922-aa1421477af4

Server name
payrollsystemserver.database.windows.net
Elastic database pool
No elastic pool
Connection strings
Show database connection strings
Pricing tier
Free
Oldest restore point
2018-03-30 00:00 UTC

DTU PERCENTAGE
0.1 %

Database size
21.9%
CURRENT
7 MB
QUOTA
32 MB

There are no alerts or recommendations to display.

3 Conclusions

ThinScale Application Suite was mainly developed as a hobby, but I can see a bright future for it. The advantages of being a windows form, means documentation will be widely available online, and updates and future features will be easily implemented thanks to C# programming language. The scope of this application is quite restricted being only one operating system compatible, but statistic don't lie, and Windows have the highest number of desktop users, and therefore the largest selection of commercial software. The idea, to be fully portable across multiple platform, like Linux or MacOS, and the integration of UWP (Unified Windows Platform) are anyway in scope. Integrating Xamarin and Visual Studio will be easy, so I don't exclude, being fully portable, to be next step. Thanks to DevExpress library, I could create a modern, stylish and functional application. Using SQL as my main back-end storage, the interaction between the three, Visual Studio, Database and libraries was effortless. I know at the end of the day you can look back and say: I could have done this rather than that, but pressure was high, and time was flying. Overall, I am happy with it.

Throughout the four years of this course, I've meet new people, now good friends, I've encountered challenges, I've developed new skills and further the one I knew, I've created lots of little application but most importantly I've learned a lot and I've grown as a person and as a team player. I think, ThinScale Application Suite is the representation of the passion, the thrive always to learn, to challenge myself and succeed as a student and as an I.T professional.

4 Further development or research

The scope for constant improvement on this concept is huge with the following functions already planned for future release;

- Multi Factor authentication
- Internal Chat Box (send news or interchange information)
- UWP implementation
- Multi operating system compatible
- Internal Log Management
- Biometric Ready

The above examples show that the application not only has relevance today within ThinScale Company but is already looking forward towards evolving into a multi company EMS (Employees Management System) application.

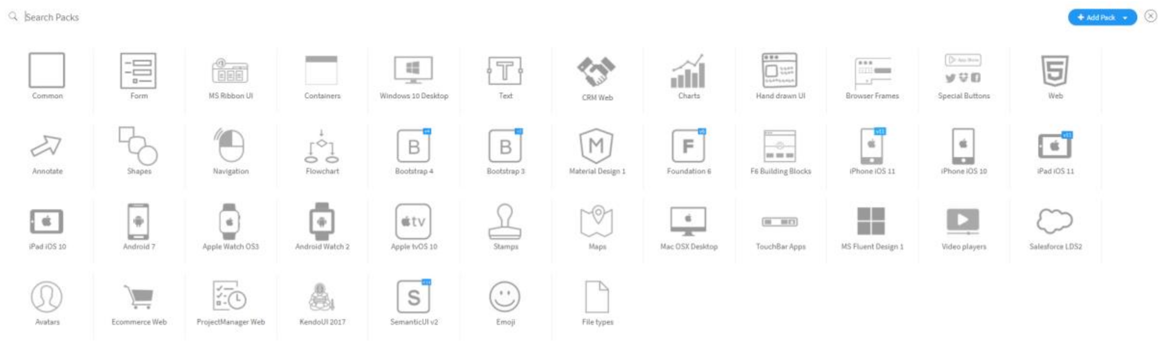
5 References

- Docs.microsoft.com. (2017). *ClickOnce Security and Deployment*. [online] Available at: https://docs.microsoft.com/en-gb/visualstudio/deployment/clickonce-security-and-deployment [Accessed 22 Oct. 2017].

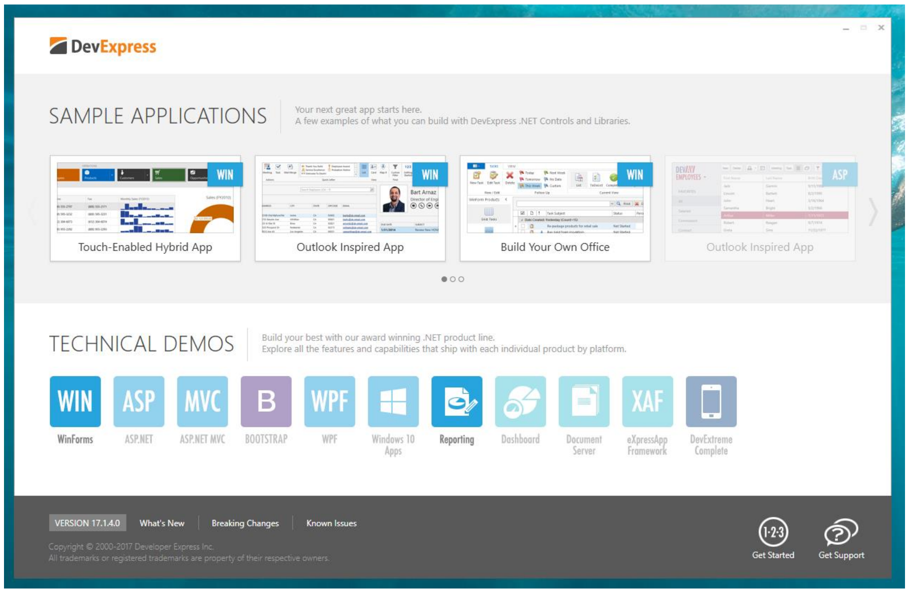
- Docs.microsoft.com. (2017). *Create a Stored Procedure*. [online] Available at: https://docs.microsoft.com/en-us/sql/relational-databases/stored-procedures/create-a-stored-procedure [Accessed 22 Oct. 2017].

- En.wikipedia.org. (2017). *C Sharp (programming language)*. [online] Available at: https://en.wikipedia.org/wiki/C_Sharp_(programming_language) [Accessed 22 Oct. 2017].

- Wireframepro.mockflow.com, (2018). *MockFlow - Wireframe Tools, Prototyping Tools, UI Mockups, UX Suite*. [online] Available at: https://wireframepro.mockflow.com [Accessed 2 May 2018].



- Inc., D. (2017). *.NET UI Controls for Developers of Mobile, Desktop, Web & Reporting Applications | www.DevExpress.com*. [online] Devexpress.com. Available at: https://www.devexpress.com/ [Accessed 22 Oct. 2017].

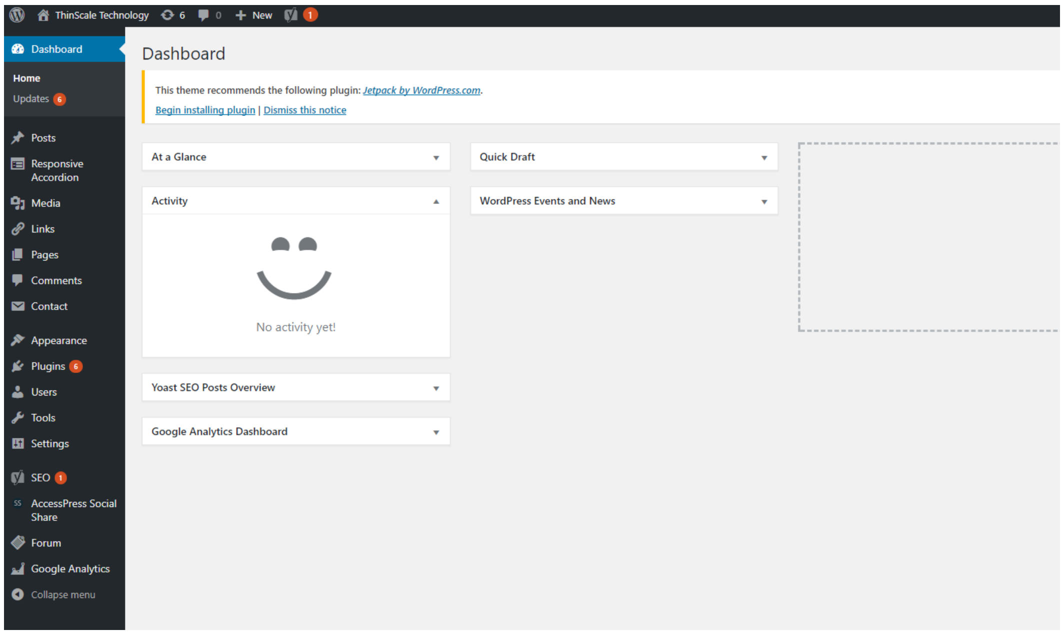


Your next great app starts here.
A few examples of what you can build with DevExpress .NET Controls and Libraries.

Build your best with our award winning .NET product line.
Explore all the features and capabilities that ship with each individual product by platform.

Copyright © 2000-2017 Developer Express Inc.
All trademarks or registered trademarks are property of their respective owners.

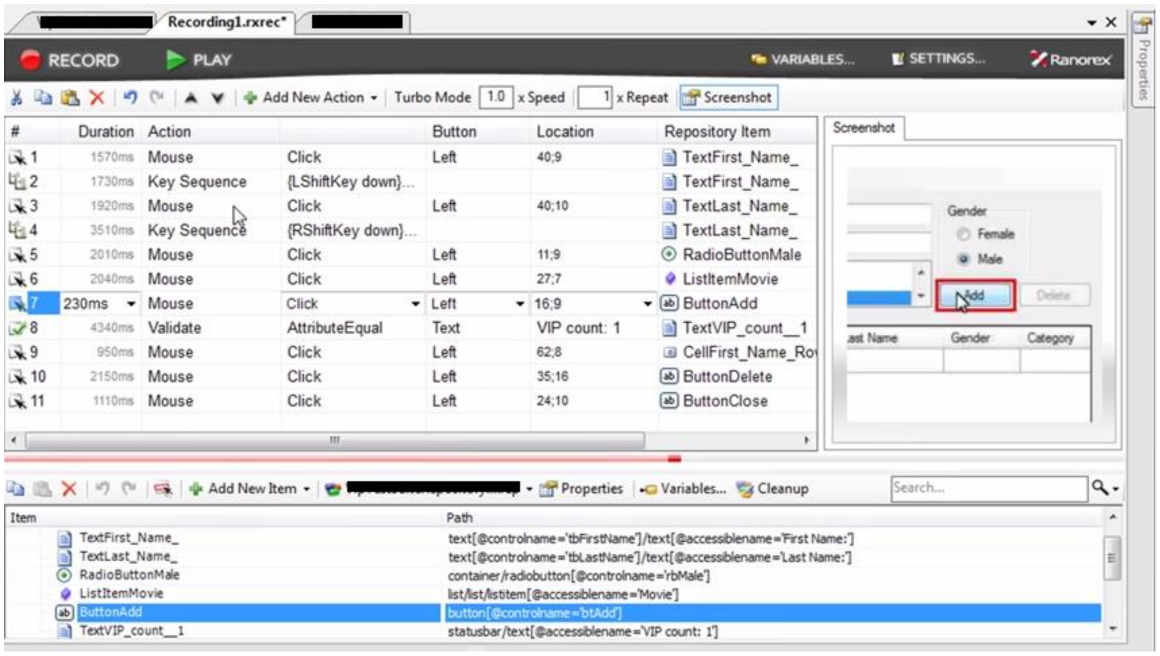
- WordPress.com. (2017). *Create a site with WordPress*. [online] Available at: https://wordpress.com/com-vs-org/?sgmt=gb&utm_source=adwords&utm_campaign=G_Search_Brand_Desktop_IE_en_x_x&utm_medium=cpc&keyword=wordpress&creative=210654422609&campaignid=648381530&adgroupid=28074825850&matchtype=e&device=c&network=g&targetid=kwd-313411415&locationid=1007850&gclid=Cj0KCQjwg7HPBRDUARIsAMeR_0jp0zdcF3CBgFoq93KsDmFQznKt7Poxm02e9s1UTPzDwcgzwBh6nmEaApz4EALw_wcB [Accessed 22 Oct. 2017].



This theme recommends the following plugin: Jetpack by WordPress.com.
Begin installing plugin | Dismiss this notice

No activity yet!

- Ranorex.com. (2017). *Test Automation Tools | Automated Software Testing with Ranorex*. [online] Available at: https://www.ranorex.com/test-automation-tools.html?gclid=Cj0KCQjwg7HPBRDUARIsAMeR_0iPpTKN5iytEAVTCzclFhAwgT5Z6Zfs8yzjRr3jobD8T9-BAJKa6MIaAqBxEALw_wcB [Accessed 22 Oct. 2017].

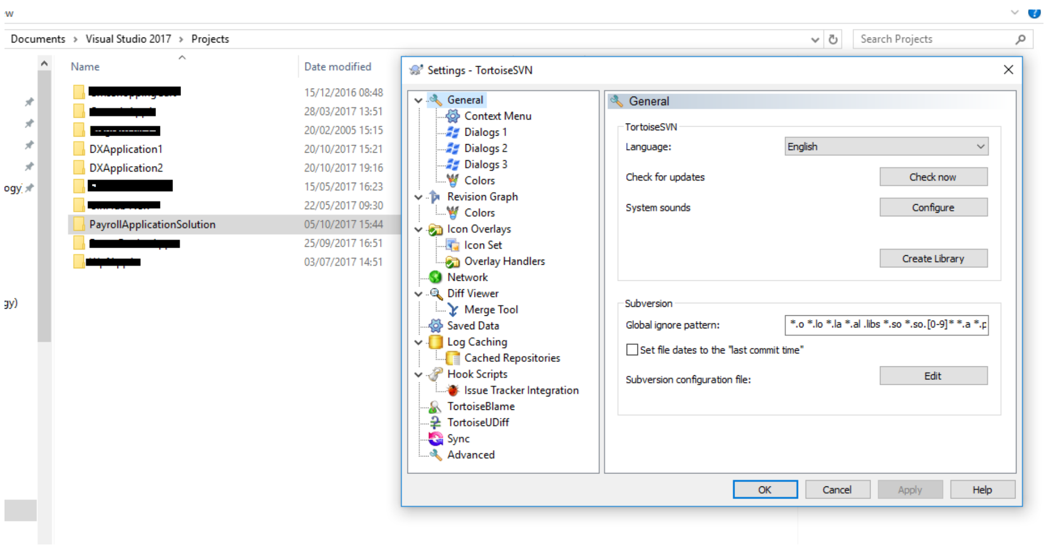


1.0 x Speed 1 x Repeat

#	Duration	Action		Button	Location	Repository Item
1	1570ms	Mouse	Click	Left	40;9	TextFirst_Name_
2	1730ms	Key Sequence	{LShiftKey down}...			TextFirst_Name_
3	1920ms	Mouse	Click	Left	40;10	TextLast_Name_
4	3510ms	Key Sequence	{RShiftKey down}...			TextLast_Name_
5	2010ms	Mouse	Click	Left	11;9	RadioButtonMale
6	2040ms	Mouse	Click	Left	27;7	ListItemMovie
7	230ms	Mouse	Click	Left	16;9	ButtonAdd
8	4340ms	Validate	AttributeEqual	Text	VIP count: 1	TextVIP_count__1
9	950ms	Mouse	Click	Left	62;8	CellFirst_Name_Ro
10	2150ms	Mouse	Click	Left	35;16	ButtonDelete
11	1110ms	Mouse	Click	Left	24;10	ButtonClose

Item	Path
TextFirst_Name_	text[@controlname='tbFirstName']/text[@accessiblename='First Name:']
TextLast_Name_	text[@controlname='tbLastName']/text[@accessiblename='Last Name:']
RadioButtonMale	container/radiobutton[@controlname='rbMale']
ListItemMovie	list/list/listitem[@accessiblename='Movie']
ButtonAdd	button[@controlname='btAdd']
TextVIP_count__1	statusbar/text[@accessiblename='VIP count: 1']

- Tortoisesvn.net. (2017). *Home · TortoiseSVN*. [Online] Available at: https://tortoisesvn.net [Accessed 24 Oct. 2017].



Name	Date modified
[redacted]	15/12/2016 08:48
[redacted]	28/03/2017 13:51
[redacted]	20/02/2005 15:15
DXApplication1	20/10/2017 15:21
DXApplication2	20/10/2017 19:16
[redacted]	15/05/2017 16:23
[redacted]	22/05/2017 09:30
PayrollApplicationSolution	05/10/2017 15:44
[redacted]	25/09/2017 16:51
[redacted]	03/07/2017 14:51

General

TortoiseSVN

Language: English

Check for updates

System sounds

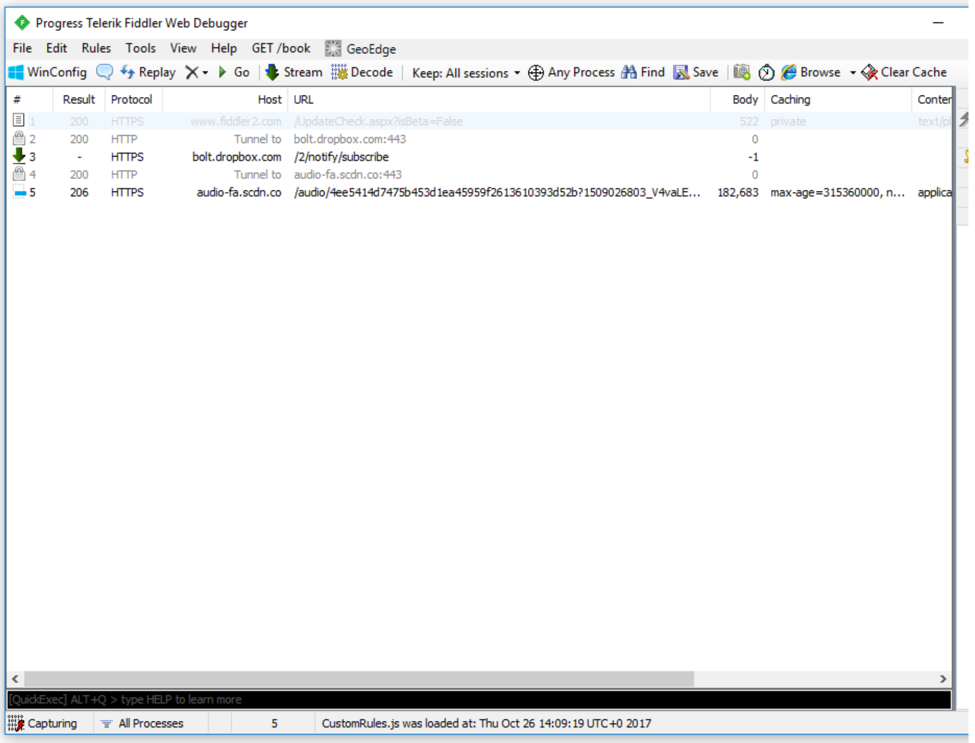
Subversion

Global ignore pattern: *.o *.lo *.la *.al .libs *.so *.so.[0-9]* *.a *.p

[ ] Set file dates to the "last commit time"

Subversion configuration file:

- Telerik.com. (2017). *Download Fiddler Web Debugging Tool for Free by Telerik*. [online] Available at: https://www.telerik.com/download/fiddler [Accessed 26 Oct. 2017].



#	Result	Protocol	Host	URL	Body	Caching	Conter
1	200	HTTPS	www.fiddler2.com	/UpdateCheck.aspx?isBeta=False	522	private	text/p
2	200	HTTP	Tunnel to	bolt.dropbox.com:443	0		
3	-	HTTPS	bolt.dropbox.com	/2/notify/subscribe	-1		
4	200	HTTP	Tunnel to	audio-fa.scdn.co:443	0		
5	206	HTTPS	audio-fa.scdn.co	/audio/4ee5414d7475b453d1ea45959f2613610393d52b?1509026803_V4vaLE…	182,683	max-age=315360000, n…	applica

```
[QuickExec] ALT+Q > type HELP to learn more
Capturing    All Processes    5    CustomRules.js was loaded at: Thu Oct 26 14:09:19 UTC+0 2017
```

6 Appendix

6.1 Project Proposal

FINAL Project
Proposal-ThinScale I

6.2 Employee Guide

EMPLOYEE QUICK GUIDE

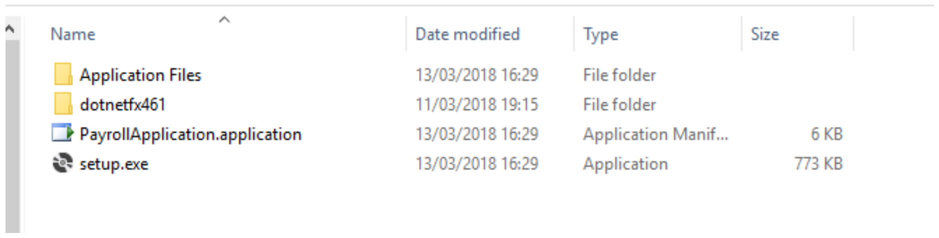
STEP 1 – INSTALLATION

From the Fun Box folder, click Download and save the file to your desire location.

Picture TBD

STEP 2

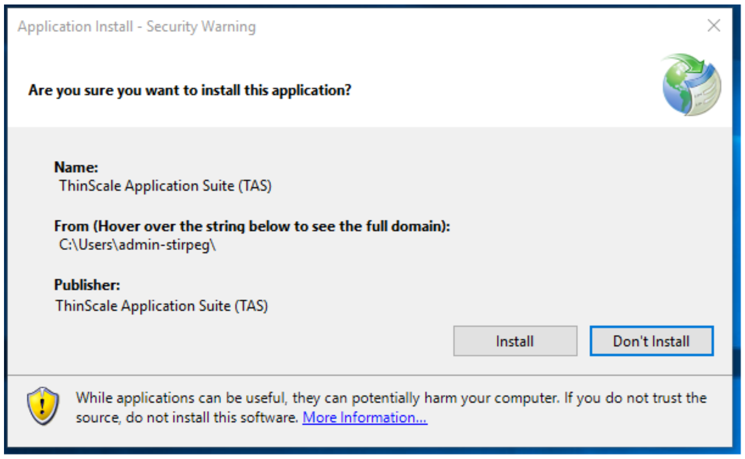
Once the file has been downloaded, please unzip it and click the setup.exe file to begin installation.



Name	Date modified	Type	Size
Application Files	13/03/2018 16:29	File folder	
dotnetfx461	11/03/2018 19:15	File folder	
PayrollApplication.application	13/03/2018 16:29	Application Manif...	6 KB
setup.exe	13/03/2018 16:29	Application	773 KB

STEP 3

Click install to begin installation. Depending on your computer speed it might take few minutes.



Application Install - Security Warning

Are you sure you want to install this application?

Name:
ThinScale Application Suite (TAS)

From (Hover over the string below to see the full domain):
C:\Users\admin-stirpeg\

Publisher:
ThinScale Application Suite (TAS)

While applications can be useful, they can potentially harm your computer. If you do not trust the source, do not install this software. More Information...

6.3 Project Plan

Final-Gantt Chart.mpp

6.4 Monthly Journals

Reflective Journa September.docx

Reflective Journal October.docx

Reflective Journal November.docx

Reflective Journal January.docx

Reflective Journal February.docx

Reflective Journal March.docx

Reflective Journal April.docx

Reflective Journal May.docx

6.5 Database Schema

Db Schema.txt

6.6 Meetings

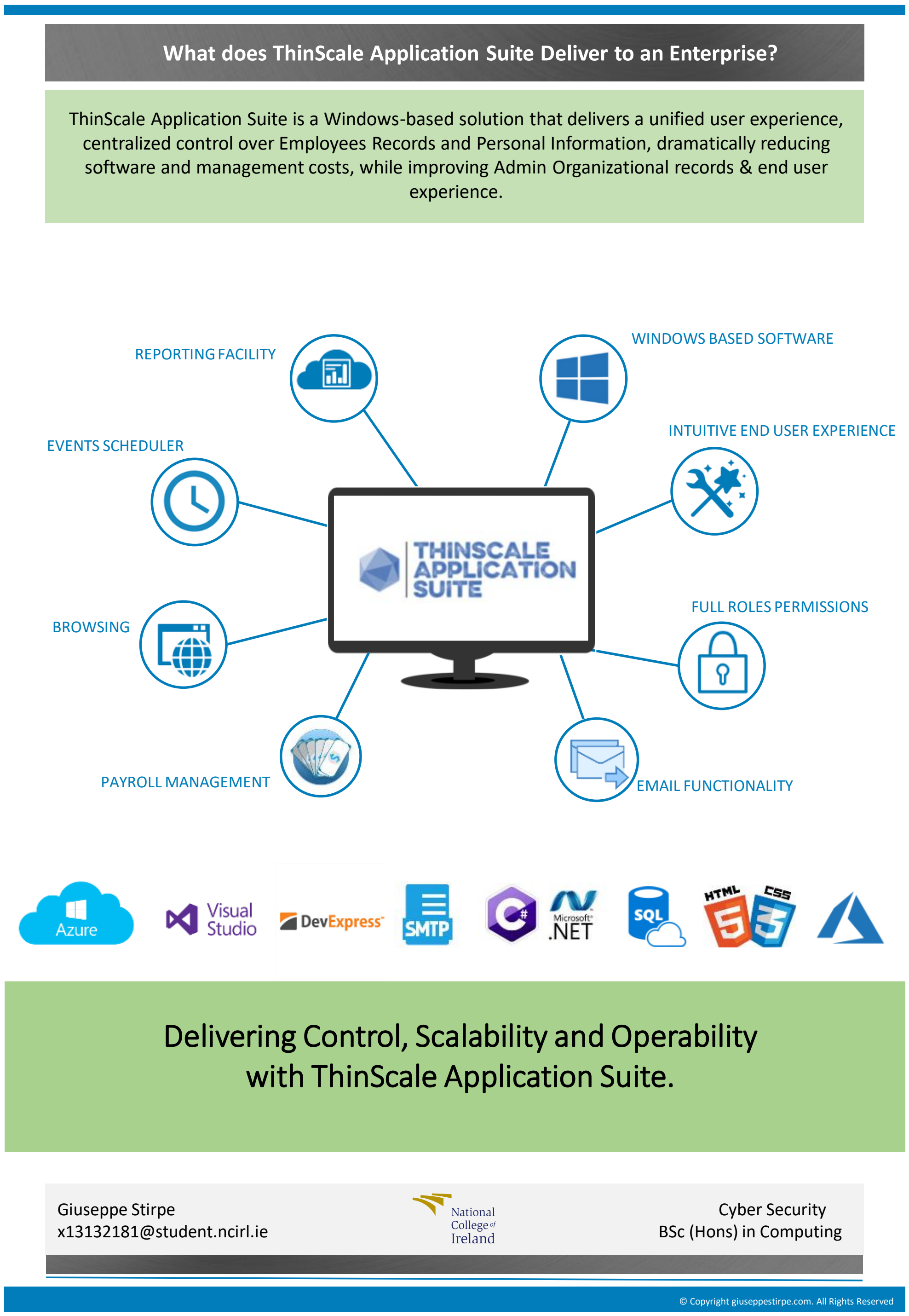
Meeting Agenda.docx

Meeting Agenda 2.docx

Meeting Agenda 3.docx

Meeting Agenda 4.docx

6.7 Poster



What does ThinScale Application Suite Deliver to an Enterprise?

ThinScale Application Suite is a Windows-based solution that delivers a unified user experience, centralized control over Employees Records and Personal Information, dramatically reducing software and management costs, while improving Admin Organizational records & end user experience.

REPORTING FACILITY
WINDOWS BASED SOFTWARE
EVENTS SCHEDULER
INTUITIVE END USER EXPERIENCE
THINSCALE APPLICATION SUITE
BROWSING
FULL ROLES PERMISSIONS
PAYROLL MANAGEMENT
EMAIL FUNCTIONALITY
Azure
Visual Studio
DevExpress
SMTP
Microsoft .NET
SQL
HTML 5
CSS 3

Delivering Control, Scalability and Operability with ThinScale Application Suite.

Giuseppe Stirpe
x13132181@student.ncirl.ie

National College of Ireland

Cyber Security
BSc (Hons) in Computing

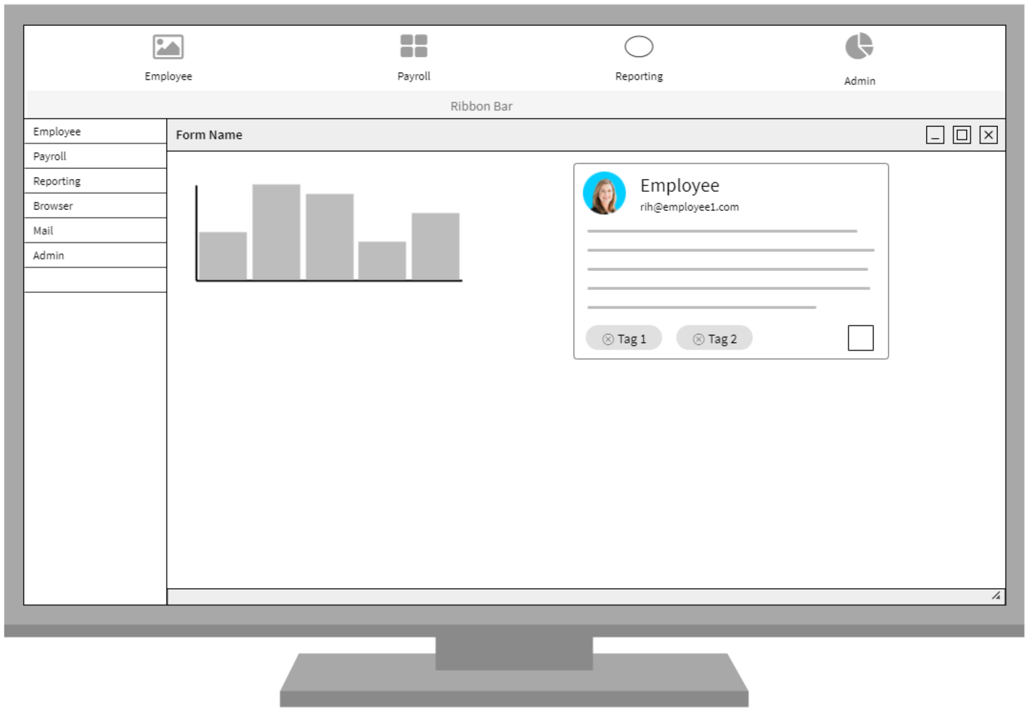
© Copyright giuseppestirpe.com. All Rights Reserved

6.8 Customer Questionnaire

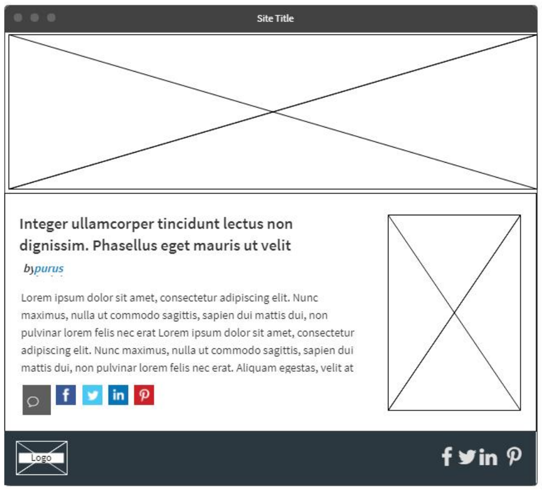
Customer Software Evaluation Form

Title of software package / program: ThinScale Application Suite	
Criterion **Customer Name:**	**Scale from 0 to 5** 0 Very poor 1 Poor 2 Ok 4 Good 5 Very Good
Is the level of language that the program offers clearly indicated?	
Is it easy to start the program?	
Is it easy to register into the program	
Is it easy to login into the form?	
Is the user interface easy to understand? (For example, is the screen layout clear and easy to interpret?)	
Is it easy to navigate through the different forms?	
Are icons that are used to assist navigation clear and intelligible?	
Is the ribbon bar useful?	
Is the left Navigation Bar useful?	
Can you easily quit something that is beyond your ability?	
Are the grammar and vocabulary used in the program accurate?	
If the program includes pictures, are they (a) relevant, (b) an aid to understanding?	
Is the program relevant to your national / regional / departmental programme of study?	
Is the program response to your actions accurate?	
Is the text inside the forms legible?	
Does the scheduler form easy to use?	
Is it easy to book your own events?	
Is it easy to check other events?	
Is the browser form easy to use?	
Is the email form easy to use?	

6.9 Application Wireframe



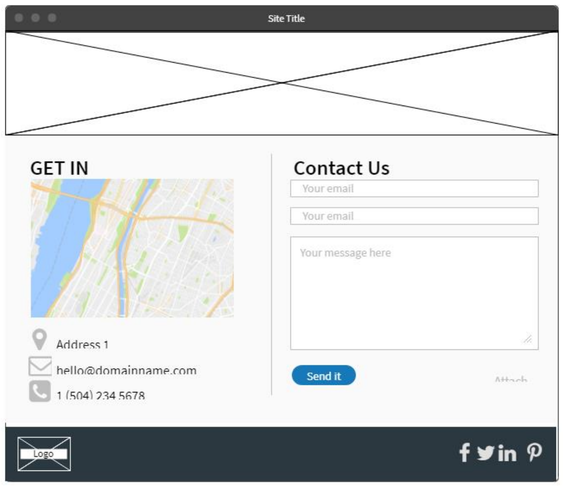
6.9.1 Web Site Wireframe



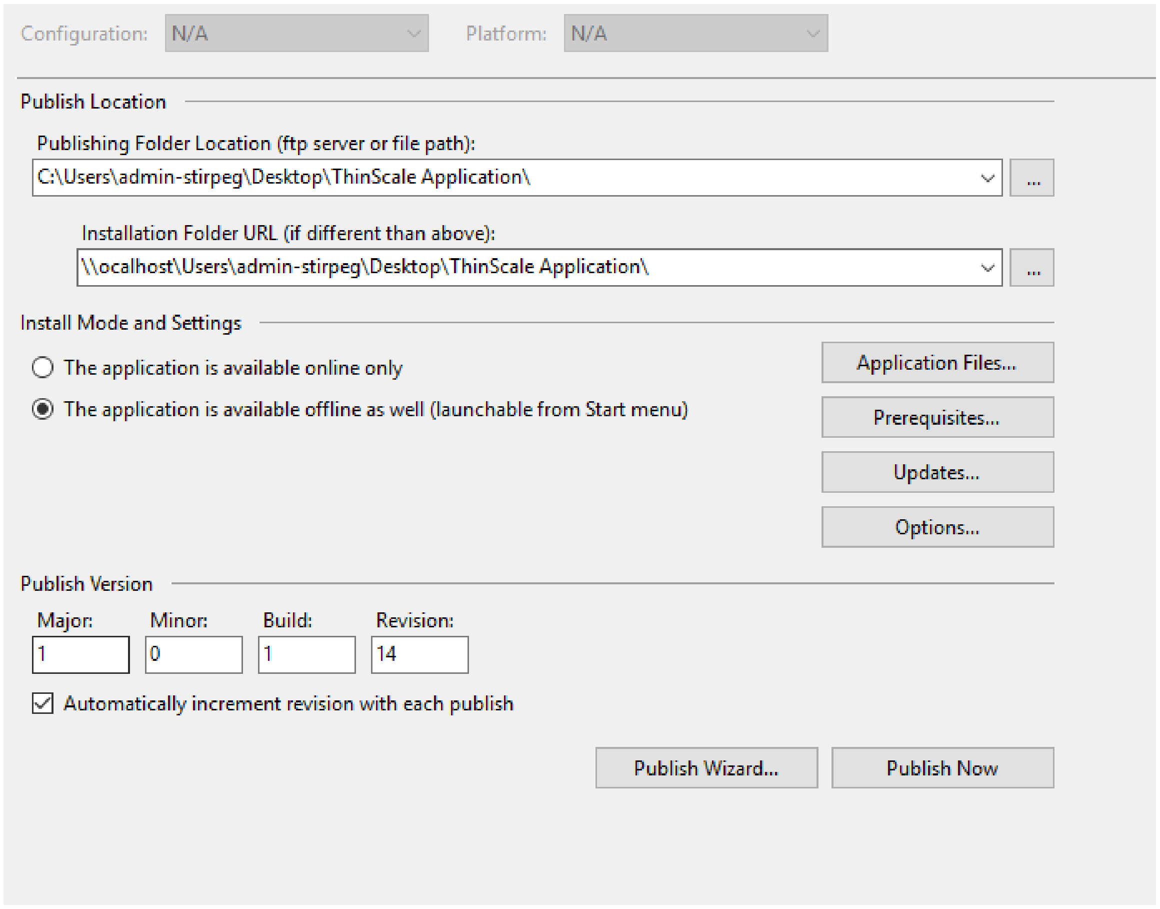
Integer ullamcorper tincidunt lectus non dignissim. Phasellus eget mauris ut velit

by purus

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem felis nec erat Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem felis nec erat. Aliquam egestas, velit at



7.0 Application Releases, Files and Visual Studio



Configuration: N/A Platform: N/A

Publish Location

Publishing Folder Location (ftp server or file path):
C:\Users\admin-stirpeg\Desktop\ThinScale Application\

Installation Folder URL (if different than above):
\\ocalhost\Users\admin-stirpeg\Desktop\ThinScale Application\

Install Mode and Settings

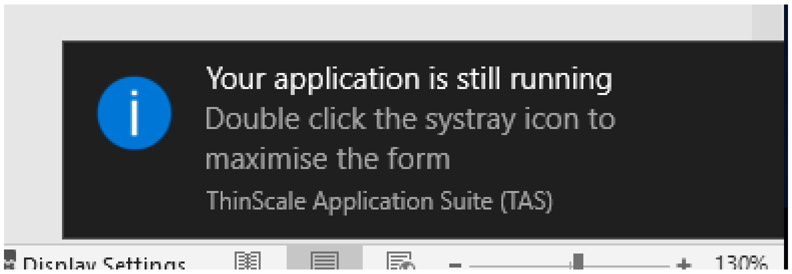
○ The application is available online only
● The application is available offline as well (launchable from Start menu)

Publish Version

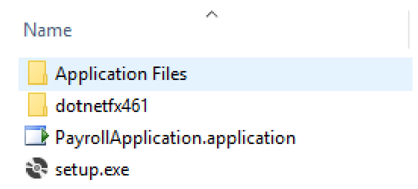
Major:	Minor:	Build:	Revision:
1	0	1	14

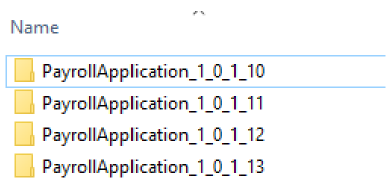
☑ Automatically increment revision with each publish

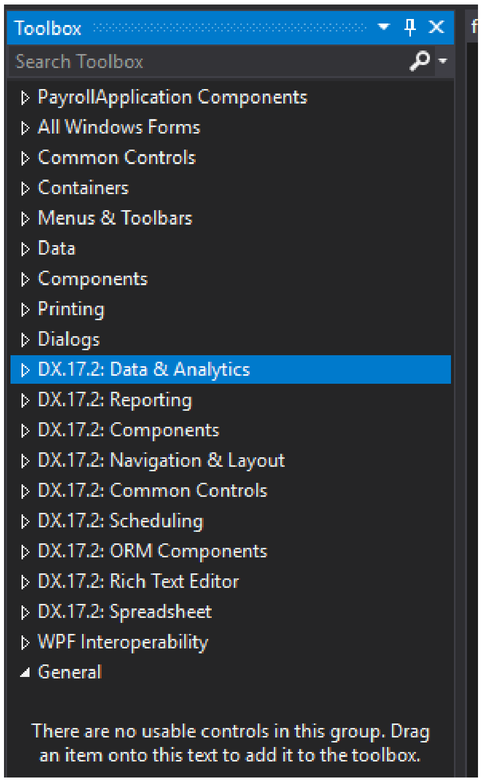
NotifyIcon when application is minimised.



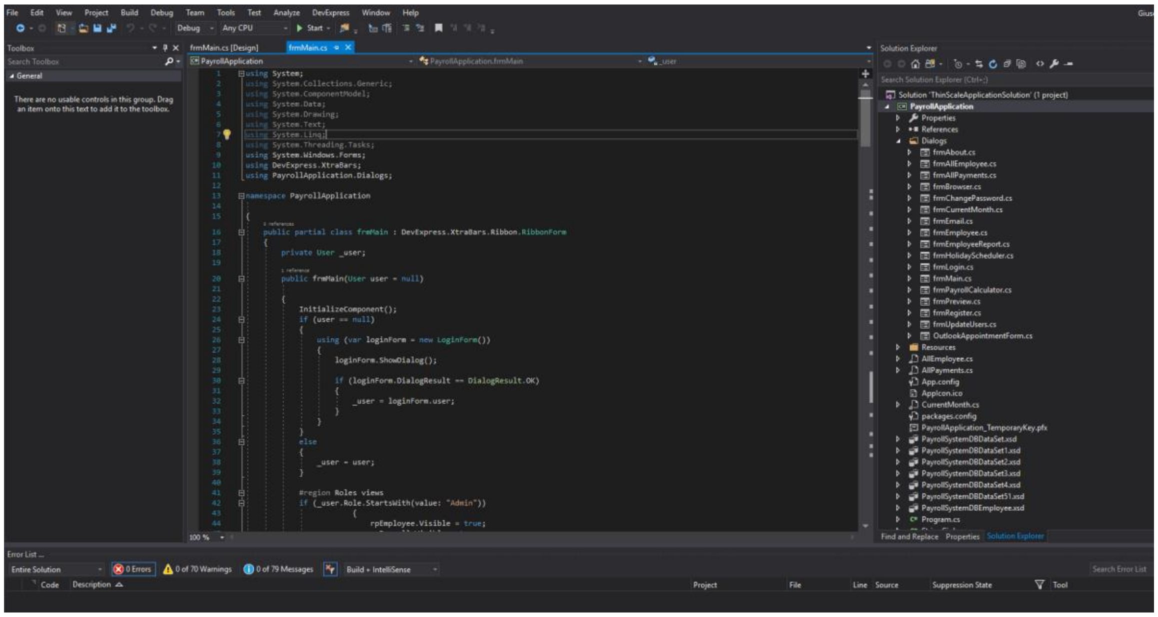
Your application is still running
Double click the systray icon to maximise the form
ThinScale Application Suite (TAS)




Name
PayrollApplication_1_0_1_10
PayrollApplication_1_0_1_11
PayrollApplication_1_0_1_12
PayrollApplication_1_0_1_13



There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.

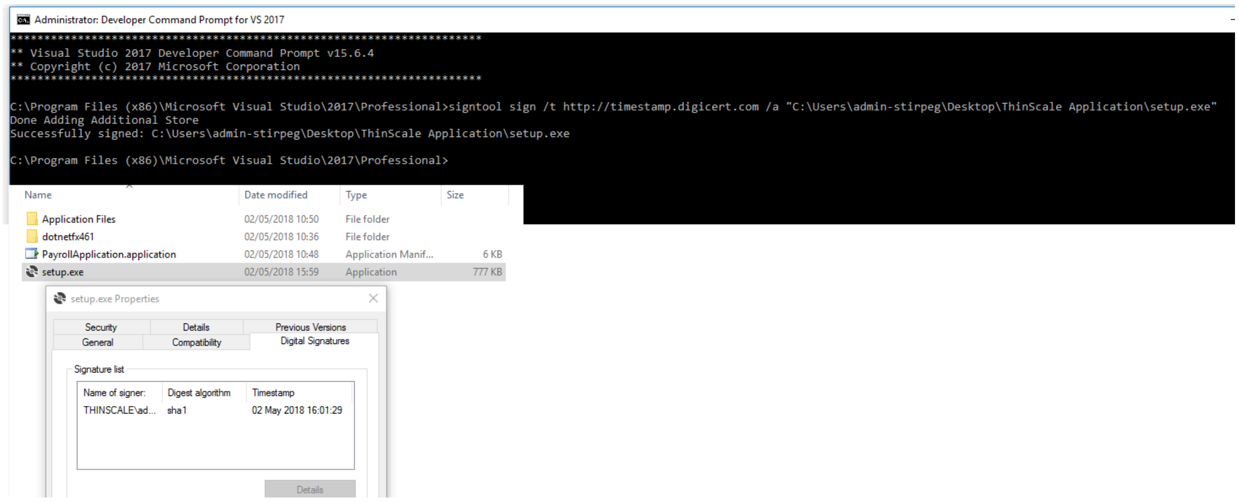


There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;
using DevExpress.XtraBars;
using PayrollApplication.Dialogs;

namespace PayrollApplication

{
    public partial class frmMain : DevExpress.XtraBars.Ribbon.RibbonForm
    {
        private User _user;

        public frmMain(User user = null)

        {
            InitializeComponent();
            if (user == null)
            {
                using (var loginForm = new LoginForm())
                {
                    loginForm.ShowDialog();

                    if (loginForm.DialogResult == DialogResult.OK)
                    {
                        _user = loginForm.user;
                    }
                }
            }
            else
            {
                _user = user;
            }

            #region Roles views
            if (_user.Role.StartsWith(value: "Admin"))
                        {
                            rpEmployee.Visible = true;
```

```
Solution 'ThinScaleApplicationSolution' (1 project)
  PayrollApplication
    Properties
    References
    Dialogs
      frmAbout.cs
      frmAllEmployee.cs
      frmAllPayments.cs
      frmBrowser.cs
      frmChangePassword.cs
      frmCurrentMonth.cs
      frmEmail.cs
      frmEmployee.cs
      frmEmployeeReport.cs
      frmHolidayScheduler.cs
      frmLogin.cs
      frmMain.cs
      frmPayrollCalculator.cs
      frmPreview.cs
      frmRegister.cs
      frmUpdateUsers.cs
      OutlookAppointmentForm.cs
    Resources
    AllEmployee.cs
    AllPayments.cs
    App.config
    AppIcon.ico
    CurrentMonth.cs
    packages.config
    PayrollApplication_TemporaryKey.pfx
    PayrollSystemDBDataSet.xsd
    PayrollSystemDBDataSet1.xsd
    PayrollSystemDBDataSet2.xsd
    PayrollSystemDBDataSet3.xsd
    PayrollSystemDBDataSet4.xsd
    PayrollSystemDBDataSet51.xsd
    PayrollSystemDBEmployee.xsd
    Program.cs
```

7.1 Digital Signature with a SHA1 Certificate/Digest Algorithm/Timestamp



Administrator: Developer Command Prompt for VS 2017

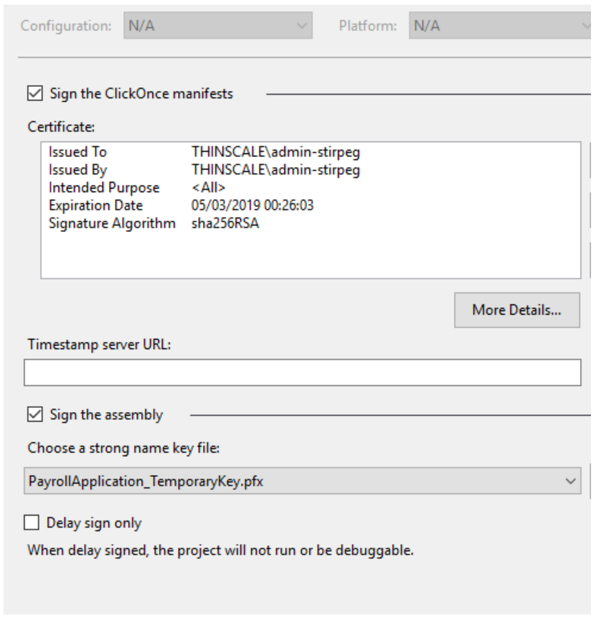
```
**********************************************************************
** Visual Studio 2017 Developer Command Prompt v15.6.4
** Copyright (c) 2017 Microsoft Corporation
**********************************************************************

C:\Program Files (x86)\Microsoft Visual Studio\2017\Professional>signtool sign /t http://timestamp.digicert.com /a "C:\Users\admin-stirpeg\Desktop\ThinScale Application\setup.exe"
Done Adding Additional Store
Successfully signed: C:\Users\admin-stirpeg\Desktop\ThinScale Application\setup.exe

C:\Program Files (x86)\Microsoft Visual Studio\2017\Professional>
```

Name	Date modified	Type	Size
Application Files	02/05/2018 10:50	File folder	
dotnetfx461	02/05/2018 10:36	File folder	
PayrollApplication.application	02/05/2018 10:48	Application Manif...	6 KB
setup.exe	02/05/2018 15:59	Application	777 KB

setup.exe Properties

Signature list

Name of signer:	Digest algorithm	Timestamp
THINSCALE\ad...	sha1	02 May 2018 16:01:29



Configuration: N/A Platform: N/A

☑ Sign the ClickOnce manifests

Certificate:

Issued To	THINSCALE\admin-stirpeg
Issued By	THINSCALE\admin-stirpeg
Intended Purpose	<All>
Expiration Date	05/03/2019 00:26:03
Signature Algorithm	sha256RSA

Timestamp server URL:

☑ Sign the assembly

Choose a strong name key file:

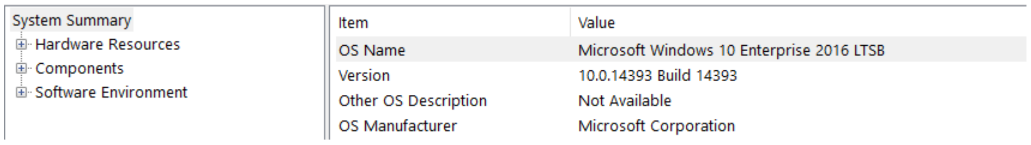
PayrollApplication_TemporaryKey.pfx

☐ Delay sign only

When delay signed, the project will not run or be debuggable.

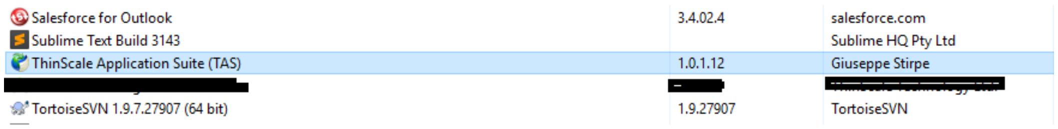
7.2 Testing VMs

W10 Virtual Machine



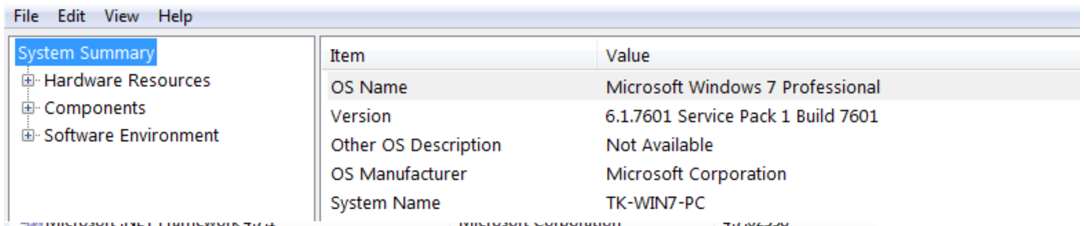
System Summary
- Hardware Resources
- Components
- Software Environment

Item	Value
OS Name	Microsoft Windows 10 Enterprise 2016 LTSB
Version	10.0.14393 Build 14393
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation



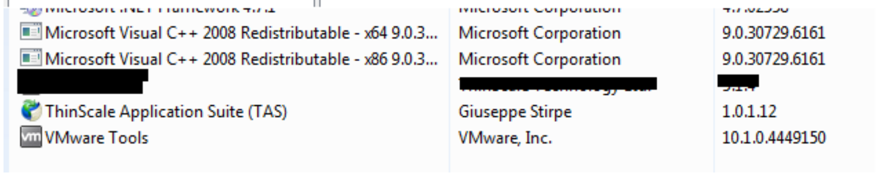
Salesforce for Outlook	3.4.02.4	salesforce.com
Sublime Text Build 3143		Sublime HQ Pty Ltd
ThinScale Application Suite (TAS)	1.0.1.12	Giuseppe Stirpe
[redacted]	[redacted]	[redacted]
TortoiseSVN 1.9.7.27907 (64 bit)	1.9.27907	TortoiseSVN

W7 Virtual Machine



System Summary
- Hardware Resources
- Components
- Software Environment

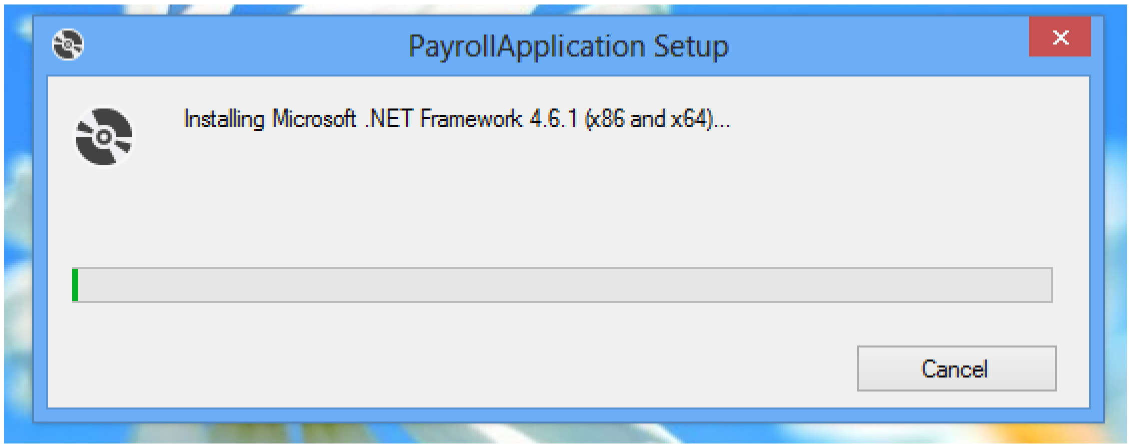
Item	Value
OS Name	Microsoft Windows 7 Professional
Version	6.1.7601 Service Pack 1 Build 7601
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	TK-WIN7-PC



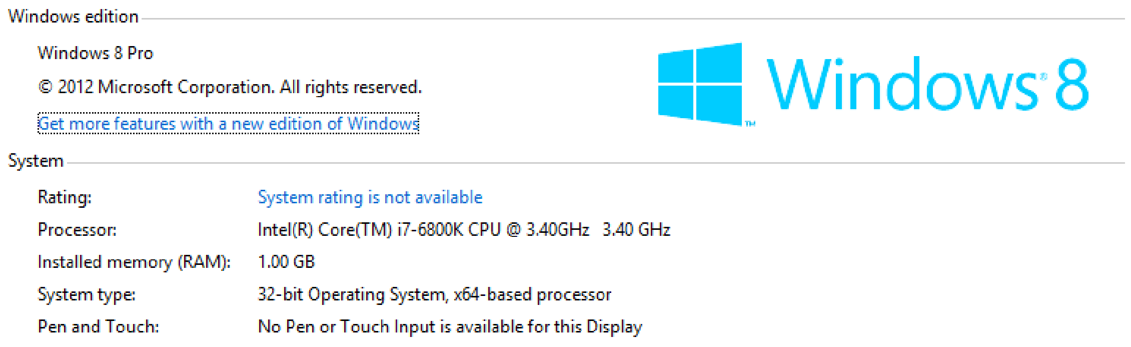
Microsoft .NET Framework 4.7.1	Microsoft Corporation	[illegible]
Microsoft Visual C++ 2008 Redistributable - x64 9.0.3...	Microsoft Corporation	9.0.30729.6161
Microsoft Visual C++ 2008 Redistributable - x86 9.0.3...	Microsoft Corporation	9.0.30729.6161
[redacted]	[redacted]	[redacted]
ThinScale Application Suite (TAS)	Giuseppe Stirpe	1.0.1.12
VMware Tools	VMware, Inc.	10.1.0.4449150

W8 Virtual Machine

.Net framework was missing and the application performed the installation prior my application to meet the requirements


PayrollApplication Setup

Installing Microsoft .NET Framework 4.6.1 (x86 and x64)...


Windows edition

Windows 8 Pro

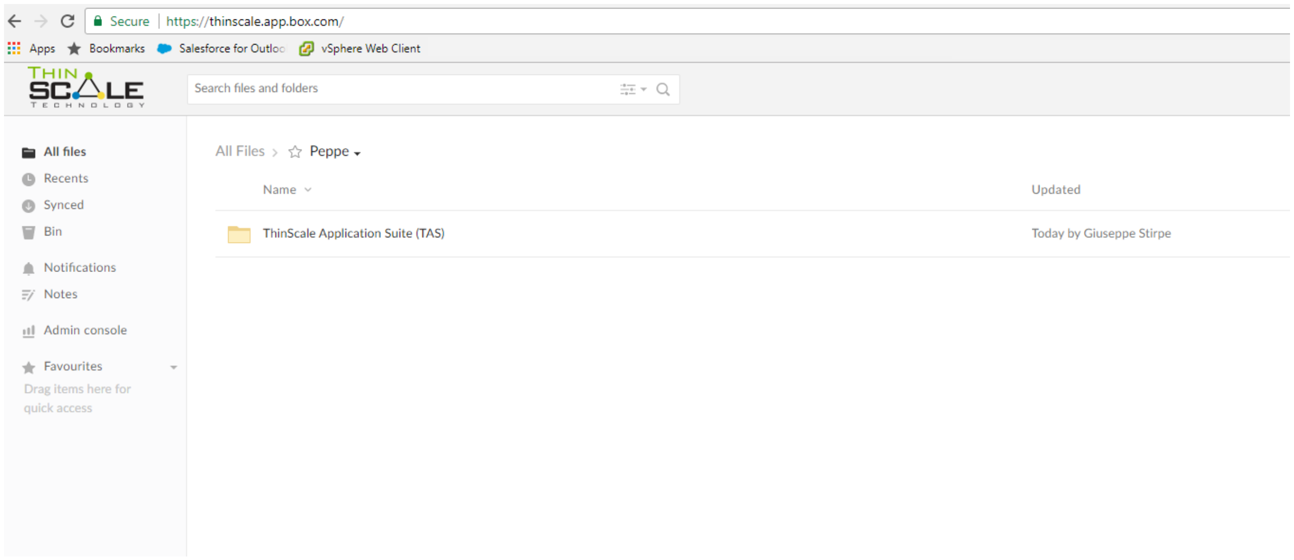
© 2012 Microsoft Corporation. All rights reserved.

Get more features with a new edition of Windows

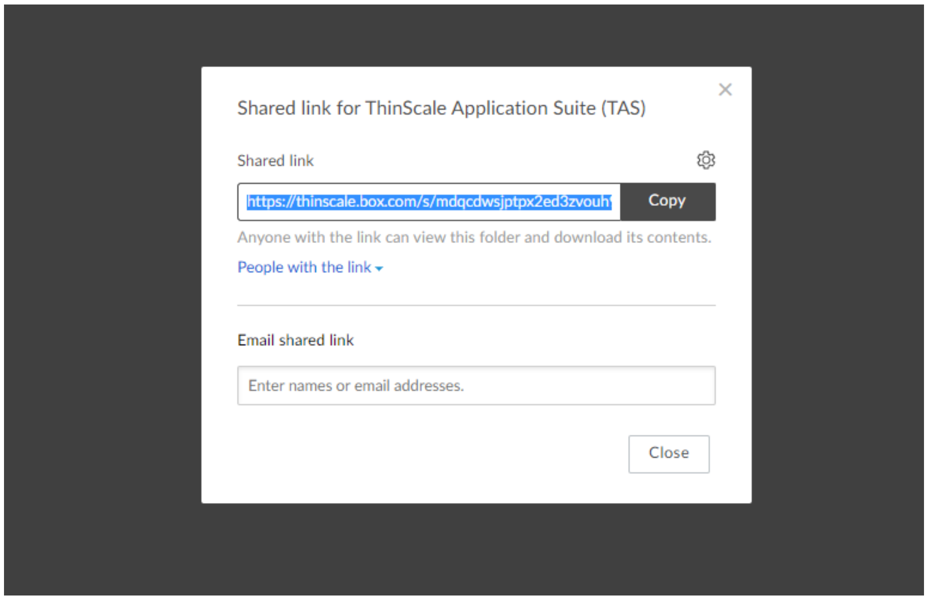
System

Rating:	System rating is not available
Processor:	Intel(R) Core(TM) i7-6800K CPU @ 3.40GHz 3.40 GHz
Installed memory (RAM):	1.00 GB
System type:	32-bit Operating System, x64-based processor
Pen and Touch:	No Pen or Touch Input is available for this Display

7.3 Box Folder



ThinScale Application Suite (TAS) — Today by Giuseppe Stirpe



Shared link for ThinScale Application Suite (TAS)

Shared link

https://thinscale.box.com/s/mdqcdwsjptpx2ed3zvouh

Anyone with the link can view this folder and download its contents.

7.4 Project Declaration

Declaration Cover Sheet for Project Submission

SECTION 1 *Student to complete*

Name: **Giuseppe Stirpe**
Student ID: **X13132181**
Supervisor: Vikas Sahni

SECTION 2 Confirmation of Authorship

The acceptance of your work is subject to your signature on the following declaration:

I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature: [illegible] Date:______13/05/2018

NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College's Disciplinary Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.

Complete the sections above and attach it to the front of one of the copies of your assignment,

What constitutes plagiarism or cheating?

The following is extracted from the college's formal statement on plagiarism as quoted in the Student Handbooks. References to "assignments" should be taken to include any piece of work submitted for assessment.

Paraphrasing refers to taking the ideas, words or work of another, putting it into your own words and crediting the source. This is acceptable academic practice provided you ensure that credit is given to the author. Plagiarism refers to copying the ideas and work of another and misrepresenting it as your own. This is completely unacceptable and is prohibited in all academic institutions. It is a serious offence and may result in a fail grade and/or disciplinary action. All sources that you use in your writing must be acknowledged and included in the reference or bibliography section. If a particular piece of writing proves difficult to paraphrase, or you want to include it in its original form, it must be enclosed in quotation marks

and credit given to the author.

When referring to the work of another author within the text of your project you must give the author's surname and the date the work was published. Full details for each source must then be given in the bibliography at the end of the project

Penalties for Plagiarism

If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the college's Disciplinary Committee. Where the Disciplinary Committee makes a finding that there has been plagiarism, the Disciplinary Committee may recommend

- that a student’s marks shall be reduced
- that the student be deemed not to have passed the assignment
- that other forms of assessment undertaken in that   academic year by the same student be declared void
- that other examinations sat by the same student at the same sitting be declared void

Further penalties are also possible including

- suspending a student college for a specified time,
- expelling a student from college,
- prohibiting a student from sitting any examination or assessment.,
- the imposition of a fine and
- the requirement that a student to attend additional or other lectures or courses or undertake additional academic work.