

Open Source SCRUM Tool

Final Report

Submitted for the BSc in
Computer Software Development with Industrial Experience

April 2016

by

Christopher Derek Haworth

Word Count: 11,024

Table of Contents

Table of Figures	5
1 Introduction	7
2 Aim and Objectives	9
2.1 Project Aim	9
2.1.1 Objective 1 – Research Existing Solutions - Achieved	9
2.1.2 Objective 2 – Database Design – Achieved	9
2.1.3 Objective 3 – Authentication Investigation – Achieved	9
2.1.4 Objective 4 – Back End Creation – Achieved	9
2.1.5 Objective 5 – Front End Creation – Almost Achieved	9
2.1.6 Objective 6 – Reporting – Achieved	9
2.2 Secondary Objectives	10
3 Background.....	11
3.1 Traditional SDLC (Waterfall)	11
3.2 V Model	12
3.3 Spiral Model	13
3.4 SCRUM	14
3.5 Advantages and Disadvantages of SCRUM vs. Other Models	18
3.6 Tools for managing SCRUM projects.....	19
3.6.1 Trello	19
3.6.2 Microsoft Team Foundation Server	20
4 Technical Development.....	22
4.1 Design	22
4.1.1 System Layout	23
4.1.2 Database Design	25
4.1.3 API	28
4.1.4 User Interface	28
4.2 Implementation	29
4.2.1 Web API	29
4.2.2 Account Management System	34
4.2.3 ASP.NET Web User Interface.....	36
4.3 Testing	38
5 Evaluation	40
5.1 Project Achievements	40
5.2 Project Planning	42
5.3 Further Work	44
5.3.1: Known Issues	44
6 Conclusion	46
Appendix A: Glossary of SCRUM Terms	47

Appendix B: Initial Task List	48
Semester 1	48
Semester 2	49
Appendix C: Initial Time Plan	50
Appendix D: User Documentation	51
1. Introduction	52
2. Licence	53
2.1 GNU GENERAL PUBLIC LICENSE	53
2.1.1 Preamble	53
2.1.2 TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION	53
3. System Requirements	57
4. Environment Preparation	58
4.1 Domain	58
4.2 IIS Server	58
4.3 SQL Server	58
5. Installation and Configuration	59
5.1 SQL Server	59
5.2 IIS Server	59
5.3 Configuration	61
5.3.1 Web.Config	61
5.3.2 Helper.js	62
6. User Guide	63
6.1 Administration	63
6.1.1 Team Management	64
6.1.2 User Management	64
6.1.3 Log Viewer	66
6.2 Project Management	66
6.2.1 Project List	66
6.2.2 Project Details	67
6.2.3 Product Backlog Item Details	70
6.3 Reports	71
7. Frequently Asked Questions/Known Issues	73
7.1 Known Issues	73
7.2 Frequently Asked Questions	73
Appendix E: Technical Documentation	74
1. Introduction	75
2. API Overview	76
3. API Method Index	77
3.1 Project	77

3.2 Product Backlog	78
3.3 Iterations.....	78
3.4 Tasks.....	79
3.5 Users.....	80
3.6 Roles	81
3.7 Teams	82
3.8 Logs	82
4. Object Types.....	83
4.1 ProjectDTO.....	83
4.2 ProjectDetailsDTO.....	83
4.3 ProductBacklogItemDTO	86
4.4 ProductBacklogItemDetailsDTO	86
4.5 ChangePBIPriority (Taken From Fiddler)	87
4.6 IterationDTO.....	87
4.7 IterationDetailsDTO	87
4.9 SetIteration (Taken From Fiddler).....	88
4.10 BacklogItemTaskDTO.....	89
4.11 BacklogItemTaskDetailsDTO.....	89
4.12 UserDTO	89
4.13 UserDetailsDTO	89
4.14 UserToRole (Taken from Fiddler)	90
4.15 UserToTeam (Taken from Fiddler).....	90
4.16 RoleDTO	90
4.17 RoleDetailsDTO.....	90
4.18 TeamDTO.....	91
4.19 TeamDetailsDTO.....	91
4.20 TeamToProject (Taken from Fiddler).....	92
References	93

Table of Figures

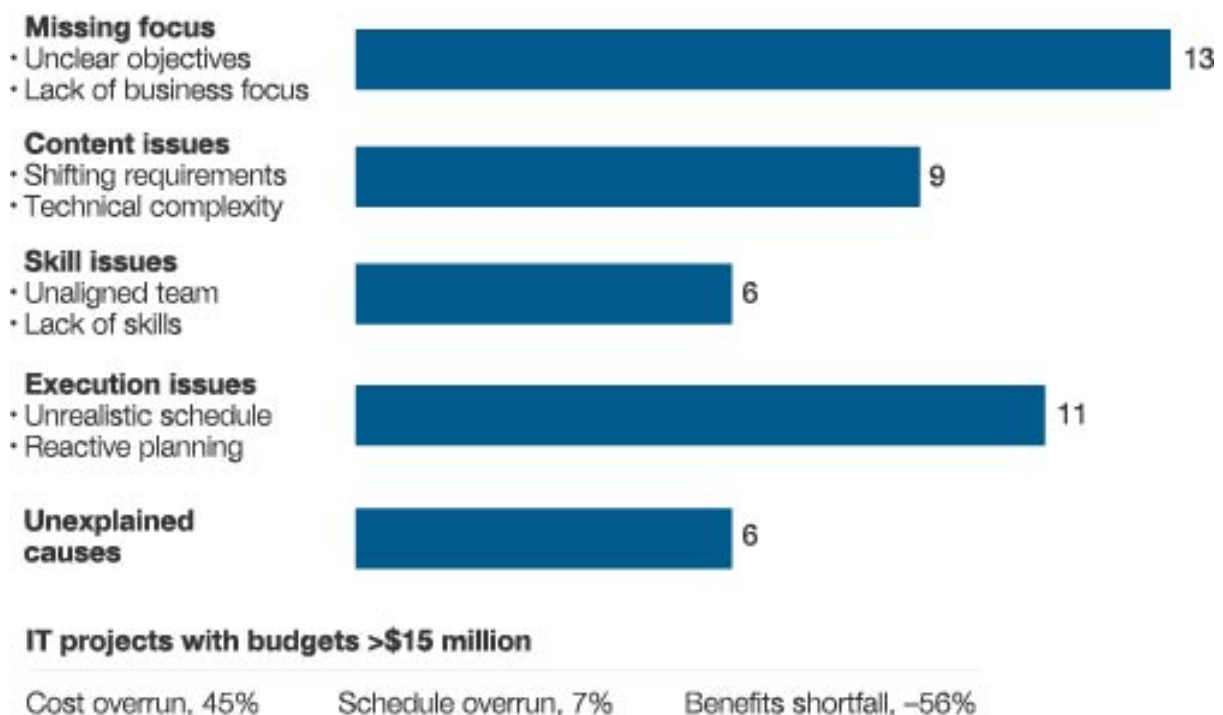
Figure 1. Rough causes of cost overruns in IT projects (Bloch, et al., 2012)	7
Figure 2. Typical Waterfall Process Model (Mahalakshmi & Sundararajan, 2013).....	11
Figure 3. Example V model process (Firesmith, 2013)	12
Figure 4. Double V model (Firesmith, 2013).....	12
Figure 5. Spiral model of the software process (Boehm, 1988)	13
Figure 6. SCRUM Process (Palmquist, et al., 2013).....	15
Figure 7. Typical Burndown Chart (International SCRUM institute, n.d.)	16
Figure 8. Typical Velocity Chart (Microsoft, n.d.)	16
Figure 9. Typical SCRUM Board	17
Figure 10. Trello welcome board (Trello Inc., n.d.)	19
Figure 11. Example of multiple people assigned to a card (Trello Inc., n.d.).....	20
Figure 12. Typical TFS welcome screen	20
Figure 13. Example of forecasting in TFS	21
Figure 14. System Architecture Diagram.....	24
Figure 15. Initial Database Design Part 1	25
Figure 16 Initial Database Design Part 2.....	26
Figure 17. Database layout after changes due to issues with EF. Exported from SQL Server Management Studio.	27
Figure 18. Initial UI mockup.....	28
Figure 19. API Class Diagram.....	30
Figure 20. Model Class Diagram.....	32
Figure 21. Final Database Schema	33
Figure 22. Class Diagram of the Account Management System.....	34
Figure 23. Flow Diagram of Authentication System.....	35
Figure 24. Example of the User Interface.....	37
Figure 25. Example Add Dialog.....	37
Figure 26. Active Directory Configuration for testing	39
Figure 27. Project Burndown.....	43
Figure 28. Project Velocity	43
Figure 29. Ordering Issue	45
Figure 30. Initial Time Plan Gantt Chart	50
User Guide Figures	
Figure 31. Running the DB Script on the Blank database to setup the schema.....	59
Figure 32. Adding the site in IIS Manager	60
Figure 33. Authentication Settings.....	61
Figure 34. Example Group Configuration	61
Figure 35. Home Page	63
Figure 36. Logging in with a user that does not have permissions to view the Administration section of the site.....	63
Figure 37. Administration Section.....	63
Figure 38. Team Management.....	64
Figure 39. Showing Teams that are assigned and not assigned to projects.	64
Figure 40. User Management (Email Address Masked for Privacy).....	65
Figure 41. Edit User Information Dialog	65
Figure 42. Team and Role Management from user control	65
Figure 43. Log Viewer	66
Figure 44. Project Index	67
Figure 45. Error Message if accessing project that you are not a member of	67
Figure 46. Project Details Page.....	67
Figure 47. Iteration Information	68
Figure 48. Add Backlog Item Dialog.....	68
Figure 49. New Backlog Item	68

Figure 50. Changing Priorities.....	68
Figure 51. Updated Priorities.....	69
Figure 52. Add to iteration.....	69
Figure 53. Edit PBI Dialog.....	69
Figure 54. Product Backlog Item Details Page	70
Figure 55. Add Task Dialog.....	70
Figure 56. Added New Task.....	71
Figure 57. Reports Page	71
Figure 58. Report page	72
API Documentation Figure	
Figure 59. API Class Diagram.....	76

1 Introduction

Planning software projects has always proved very difficult and time consuming, some methods not being very cost effective. “On average large IT projects run 45% over budget and 7% over time, while delivering 56% less value than predicted” (Bloch, et al., 2012). This causes a lot of projects to eventually fail due to different factors such as unclear objectives or changing requirements (Bloch, et al., 2012). Figure 1 illustrates a distribution of the major cause of software project failures.

Rough distribution by cause of the 45% of IT projects that experience cost overruns (for those with budgets >\$15 million in 2010 dollars), %



Source: McKinsey-Oxford study on reference-class forecasting for IT projects

Figure 1. Rough causes of cost overruns in IT projects (Bloch, et al., 2012)

Figure 1 highlights that shifting requirements is one of the top 3 causes for projects overrunning alongside unclear objectives and unrealistic schedules.

There are different models that can be used for managing software projects, with each model having benefits and disadvantages depending on the project in question. One such model is SCRUM which coincides with the Agile methodology (James, n.d.), that aims to adapt better to changing requirements for software projects.

SCRUM as a framework aims to mitigate these by using an iterative development model which actively involves the customers in the development process which allows the development teams to better respond to changes. This involves maintaining communication between both the development teams and the Product owner through a series of events as well as during the development and encourages both parties asking questions. This will be further explained later in the report.

For this project, one personal aim is to produce a system that can be used within an enterprise environment without having the large system requirements of a system like Microsoft Team Foundation, while still being able to be used to plan projects and report upon them. This will require having fast response times on slower hardware (e.g. 5-10s max for committing changes to the database).

In the following sections, I will set out a plan to build an open source SCRUM tool to help facilitate the SCRUM process, along with contrasting different software development planning methods with the SCRUM framework. As well as this there will be a detailed discussion of the different sections of this tool alongside an evaluation of the project and what would be beneficial to be done to improve the tool.

2 Aim and Objectives

2.1 Project Aim

To Create an Open Source Web Based SCRUM Management Tool using ASP.NET with a Web API based data access layer to allow for third party applications to interact with the site.

To facilitate the above aim, the following primary objectives will need to be achieved:

1. Research existing solutions
2. Design a suitable Database architecture
3. Authentication Investigation
4. Create a functioning Web API
5. Create a front end website
6. Investigate Graphing/Reporting Methods

2.1.1 Objective 1 – Research Existing Solutions - Achieved

For this objective I have researched existing SCRUM Tools/list-based planning tools, primarily focusing on both Trello¹ and Microsoft Team Foundation Server ²to cover both ends of the spectrum. This along with the authentication investigation has allowed me to come to a conclusion as to the scope of the application, which will be further discussed in the design section of this report.

2.1.2 Objective 2 – Database Design – Achieved

Following on from my research into existing solutions and forming a list of user stories and a specification for the application, A UML Database diagram has been designed and implemented on SQL Server, currently with minimal changes to the initial design.

2.1.3 Objective 3 – Authentication Investigation – Achieved

From further investigation into both existing solutions and use cases for the application, it has been decided that for this project, Active Directory will be used to facilitate the management of users. This does however limit the scope of the application towards businesses, due to the requirement of a functional Active Directory domain.

2.1.4 Objective 4 – Back End Creation – Achieved

This objective will be to create the Web API component of the system, implementing the core functionality for the application and the data access layer. This will be done before the front end as the front end will call into this system to get the data it needs.

2.1.5 Objective 5 – Front End Creation – Almost Achieved

This objective will be for the design and creation of the website that will serve as the interface to the application, it will require me to research jQuery and other interactive libraries for drag-and-drop functionality within the system.

2.1.6 Objective 6 – Reporting – Achieved

For this objective, investigation into different libraries and methods of graphing data on websites will need to be undertaken, leading into implementing this into the front end application.

¹ Trello Home Page: <http://www.trello.com>

² Microsoft Team Foundation Server Product Page: <https://www.visualstudio.com/en-us/products/tfs-overview-vs.aspx>

2.2 Secondary Objectives

In addition to the above objectives, there are also some secondary objectives that were set. Some of these objectives have been completed, however some of them have not been implemented due to time constraints. These are as follows:

1. Adjustable Sprint Board (addition of new columns to organize the board)
2. Having an area for the definition of done somewhere when the sprint board is visible as a checklist to go through before an item is moved into done.
3. Allow for further customization or use of set components, e.g. linking to adjusting the sprint board, being able to have a customized project that can have segments changed to support other SCRUM like frameworks e.g. Kanban.
4. High Performance, within 5-10 seconds to move an item around the sprint board/into sprints - Done.

Only the fourth secondary objective was able to be implemented. This will be further explained later in this report.

2.3 Deliverables

At the end of the project, the aim is to have a functional planning tool implemented which includes an API with documentation. The documentation will include both user and technical documentation. To comply with the initial aims of this being an open source tool, the source code for both the API and Web user interface will be made publically accessible in a GitHub repository, which can be found here: <https://github.com/Chronos664/os-scrum>

The documentation will cover installing the application and preparing the environment for running it along with expected inputs and outputs from the Web API allowing for interaction with 3rd party applications.

3 Background

“SCRUM is a management framework for incremental product development” (James, n.d.). SCRUM is part of the Agile framework³ of project management which is a contrasting system to the more traditional approach that waterfall takes. The Waterfall project management system is a sequential model, as shown below. Agile however focuses more towards getting software in the hands of the client faster through an iterative, customer collaboration approach (Beck, et al., 2001). There are multiple forms of Agile which have different benefits for different projects, for example, Extreme Programming (Wells, 2009), Dynamic System Development (Anon., 2011) and Lean (Poppendieck.LLC, 2015) to name a few (Freedman, 2010) which all have differences and similarities to SCRUM.

3.1 Traditional SDLC (Waterfall)

The Waterfall model is a sequential model whereby each phase must be completed before moving onto the next and the requirements are determined up front (Palmquist, et al., 2013). It is commonly split into a series of 5 sequential tasks pictured below:

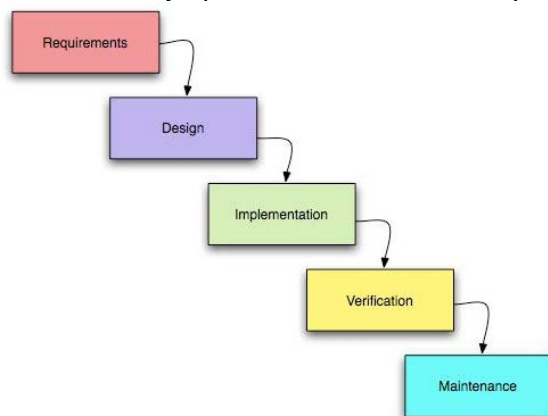


Figure 2. Typical Waterfall Process Model (Mahalakshmi & Sundararajan, 2013)

In this model, the project enters the distinct phases of the development life cycle (Requirements Capture, Design, Implementation, Testing, and Maintenance) in a sequential order. This methodology initially comes from manufacturing and mechanical engineering.

Depending on the project this can cause problems. If the project has a set of fixed requirements and a fixed time frame, then this approach is easier to implement with less overhead than SCRUM for example. If during the project a request for change is placed to the developers, then the cycle will have to be repeated from the start.

Some advantages of the waterfall method include the use of formal checking and approval before moving onto the next phase of development (Palmquist, et al., 2013), and for companies that require large amounts of documentation, for example in safety critical industries like the fire service.

Some key issues however with the waterfall model are that “customers will not get satisfaction, requirements will be in pending, no profit, waste of time” (Mahalakshmi & Sundararajan, 2013). This is due to the sequential nature and change requires re-evaluation of the previous segments.

³ Agile Manifesto: <http://agilemanifesto.org/>

3.2 V Model

The V model is a variant of the waterfall model that builds on top of it and focuses on verification and validation (Firesmith, 2013).

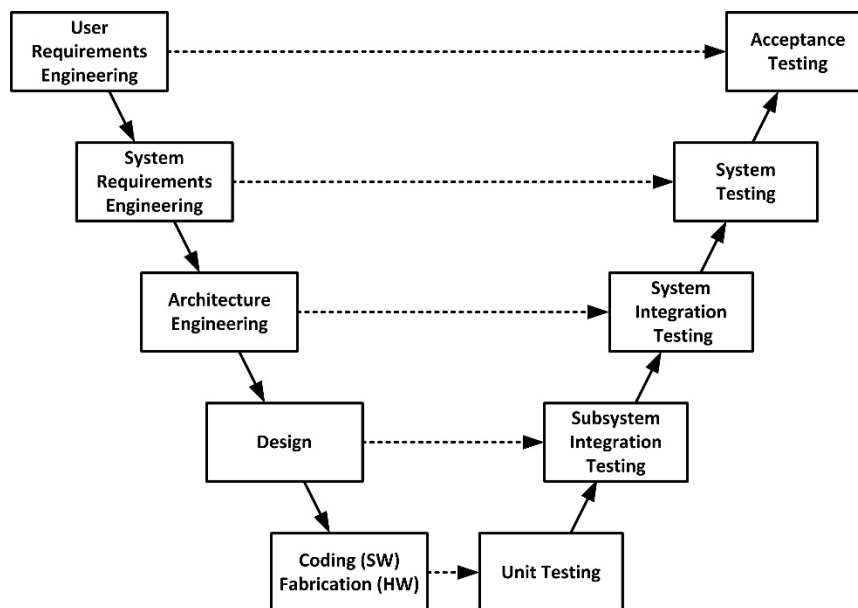


Figure 3. Example V model process (Firesmith, 2013)

With this model, the activities on the right verify or validate the product of the work on the left (Firesmith, 2013). As seen from the waterfall diagram (figure 2), this is quite similar in the stages required.

This model can then be further enhanced by adding tests to each process in the sequence, this is known as the double V model. (shown in Figure 4) (Freedman, 2010)

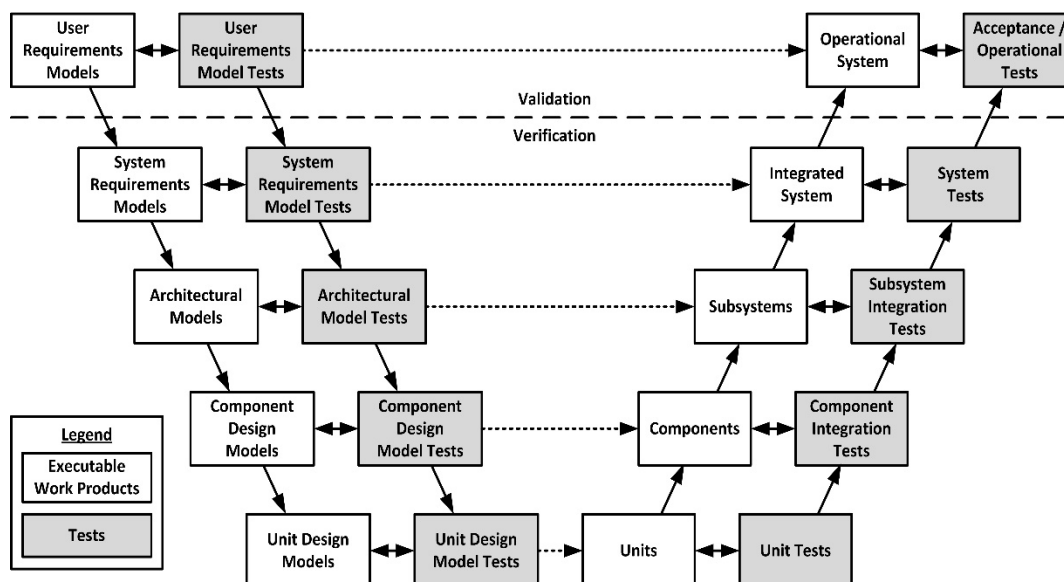


Figure 4. Double V model (Firesmith, 2013)

This model allows testing to be executed as the work products are created and these tests will form part of the validation before moving onto the next stage of the process. (Firesmith, 2013). This therefore starts to add a more iterative approach to software development as issues can be caught at the testing phase. This model can also be applied on a smaller

scale to manage an iteration of work as well as an overall project, which in the case of this model are referred to as short-duration V's. (Firesmith, 2013).

Whereas these models clearly represent the process flow required for sequential software project management, it can be applied to iterative process flows for managing what is required within an iteration as stated before however if the model is applied to a large complex system managed by an Agile method, there will need to be greater coordination across teams to provide consistent, testable components and subsystems that can be integrated (Firesmith, 2013).

3.3 Spiral Model

The spiral model is a risk driven approach to software development and incorporates many of the strengths of other models and attempts to resolve many of their difficulties (Boehm, 1988)

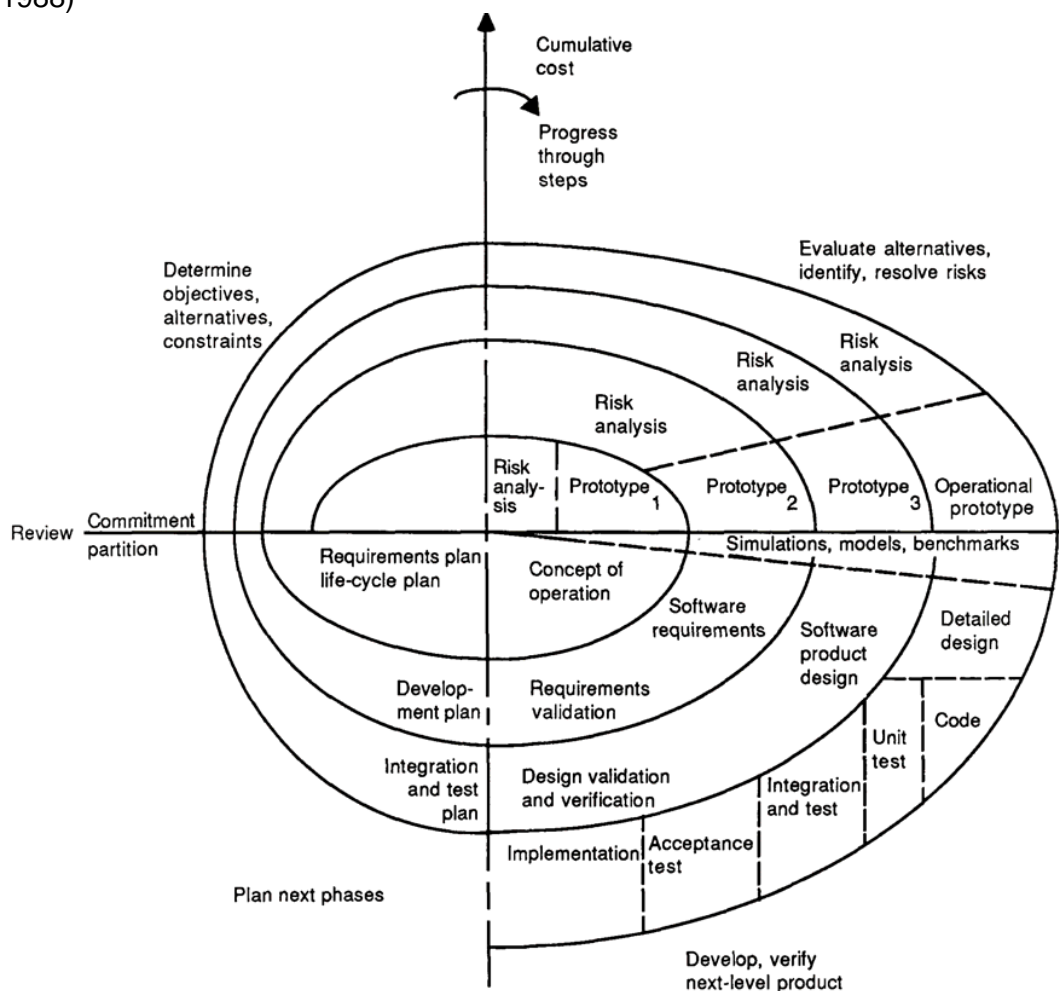


Figure 5. Spiral model of the software process (Boehm, 1988)

This model is based on various refinements of the waterfall model applied to large government software projects (Boehm, 1988) where managing risk is essential to the project.

Figure 5 represents the cumulative cost incurred in accomplishing the steps to date; and the angular dimension represents the progress made in completing each cycle of the spiral (Boehm, 1988). This therefore reflects a more iterative approach to software development

featuring multiple prototypes, at which point a risk analysis is carried out and then the prototype is validated.

There are 4 main sections to this model as show in figure 5:

1. Determine the objectives, alternatives and constraints for the cycle
2. Evaluate alternatives, identify and resolve risks
3. Develop, verify next level product
4. Plan next phase

Given the risk driven approach to the model, progress through the cycles is determined by the level of risk in the current cycle for example, if performance or user-interface risks are causing issues in the current cycle, the next cycle may be an evolutionary development one (Boehm, 1988) whereby alternatives are researched and the program is refined. If however the previous prototyping cycles have been able to resolve the risks identified, the model will then follow the basic waterfall approach modified as appropriate for an incremental approach (Boehm, 1988).

This model therefore is suited towards large mission-critical projects where minimizing risk is essential and can also be very flexible due to the incremental approach taken (Sparrow, 2011).

However, in contrast to this, the spiral model is less suited towards small projects or projects with less risk. This is due to the large overhead with handling risk assessments for small projects or if the project has low risk. Along with this, given the amount of documentation in the intermediate stages of the project, the spiral model makes management of the project very complex (Sparrow, 2011).

3.4 SCRUM

In 1995 Ken Swaber initiated a new management model based on Agile, called SCRUM. (Mahalakshmi & Sundararajan, 2013) and is an alternative method of managing the software development process. It comes under the Agile methodology banner as it contains the same concepts that Agile aims for (Mahalakshmi & Sundararajan, 2013).

SCRUM introduces different roles in comparison to Waterfall, and is geared more towards self-organizing teams rather than a hierarchical approach of a line manager. It favors getting the product into the hands of the customer vs. large documentation unlike traditional software development life cycles e.g. Waterfall.

SCRUM trades the traditional phases of waterfall for an incremental and iterative approach which focuses on high value features and allows for feedback sooner. (James, n.d.)

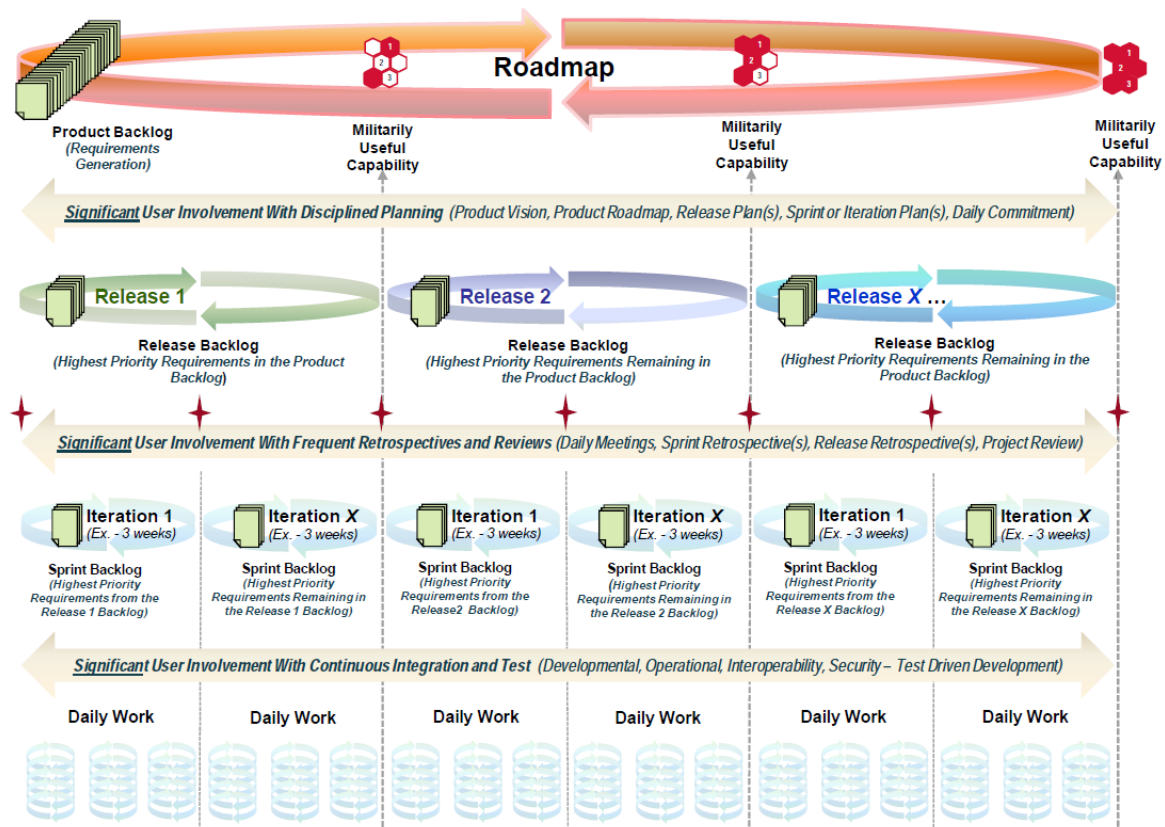


Figure 6. SCRUM Process (Palmquist, et al., 2013)

Figure 6 highlights the different events and processes during the SCRUM cycle. Starting with the Product Backlog, there are a set of iterations (Sprints) for each product release. At the end of each iteration there should be a product increment that the Product Owner could release.

A sprint or iteration, which is a fixed length block of time less than a month, e.g. 2 weeks, which will have a set of backlog items committed to be “done” in the sprint by the team, and should result in an iteration of the product that is releasable to the client.

Within a sprint, there are a series of events that take place as part of the SCRUM framework, these are:

- **Daily SCRUM/Standup:** This event is a short meeting, at the start of the working day, where the team will gather round, standing for 15 minutes or less depending on the size of the team, and discuss what each member has accomplished during the previous day and if there is anything that is impeding the tasks.
- **Sprint Review Meeting:** This takes place at the end of each sprint, can be held around the time of a Retrospective, whereby the team will demonstrate the current increment of the product to the product owner. This can also offer a chance for the product owner to review the current backlog. This meeting can also be attended by any of the project stakeholders.
- **Sprint Retrospective Meeting:** Takes place at the end of the sprint. This meeting is between the team and the SCRUM Master to review the work done during the sprint and how well the team have adhered to the SCRUM principles, this is to see what went well in the sprint as well as what the team felt did not go as well. This is used to then improve the iteration process as well as reinforce the positive outcomes of the sprint.

- **Sprint Planning Meeting:** This is where the Product Owner and the team negotiate which Product Backlog Items they will commit to finishing within the sprint (James, n.d.). It is important that this decision is jointly made between the team and the Product Owner to reduce technical debt. After which point these items are then “committed” to the sprint backlog to be worked on for the current sprint.
- **Backlog Refinement Meeting:** This is where the backlog items are assessed by the team to ensure that they are understood. The backlog items will also be given an estimate as to the required effort relative to each other. If the task is given an effort score that is too large for a single sprint (Calculated from the average velocity of previous sprints) the task will be broken down into smaller items.

Along with the Above events, SCRUM also has a series of items to aid in the planning and reporting of the state of a project, some of these are as follows (For more information please refer to the glossary of SCRUM terms in Appendix A):

- **SCRUM Board:** Also known as a sprint board. This is an area in which the items of work for the current sprint are tracked. Within a SCRUM environment there are three columns, Not Started (or To Do), In Progress, and Done. Figure 9 (over the page) shows an example of this taken from another project.
- **Burndown Chart:** This is a graph of effort against time for a sprint and shows the stakeholders if the team are on track to finishing the tasks within the sprint, or can show the overall state of the project. E.g.



Figure 7. Typical Burndown Chart (International SCRUM institute, n.d.)

- **Velocity Chart:** This graph is a sum of all the estimated effort of the work items for each sprint plotted as a bar chart of completed effort against sprints. This can be used to calculate the average amount of work that can be completed by a team in a sprint e.g.

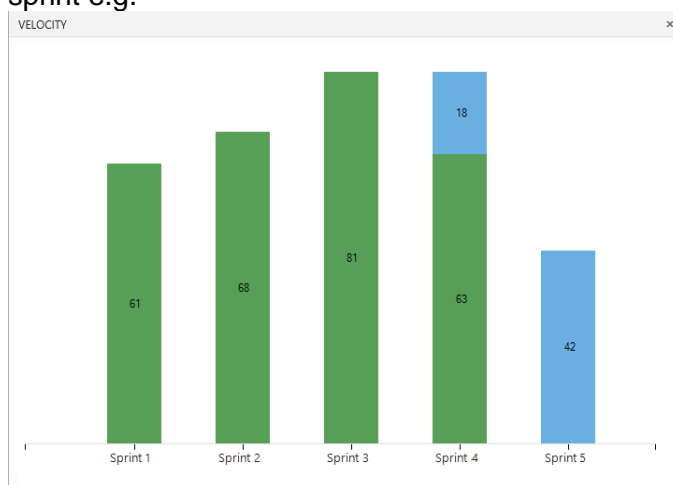


Figure 8. Typical Velocity Chart (Microsoft, n.d.)

	TO DO	IN PROGRESS 5 h	DONE
Outlook Integration Test Project +			<div>Create Outlook Test Project Christopher D. ...</div> <div>Create Template Parser for Configuration of Template</div>
Create Test Server +			<div>Create HTTP Server to handle Snom Incoming requests Christopher D. ...</div> <div>Setup Message Queue Manager to handle Messages</div>
Create Test Client 3 h +		<div>Test Form 3 Christopher D. ...</div>	
Design communications framework for inter-pc communication using Message Queuing +			<div>Create Message Queue Manager Christopher D. ...</div> <div>Create Message Types for MSMQ Christopher D. ...</div>
Design Database for windows service and address book 2 h +		<div>Create Shared Data Access Layer for Database 2 Christopher D. ...</div>	<div>Create Basic Database Layout Christopher D. ...</div> <div>Design Database Layout Christopher D. ...</div>
Investigate Snom Configuration +			<div>Create Action Link Mappings for Project on test Device Christopher D. ...</div>

Figure 9. Typical SCRUM Board

3.5 Advantages and Disadvantages of SCRUM vs. Other Models

Depending on the project, for example, complex work involving knowledge creation and collaboration such as new product development (James, n.d.) and for web based feature rich systems where client input has a large impact, SCRUM can provide many benefits over more traditional models like Waterfall while incorporating some of the other models discussed like the V Model. Some of these benefits are as follows:

- Greater customer interaction
- More flexible to adapt to change
 - This however is also present in other models that follow an iterative approach for example the spiral model.
 - It can also be mitigated to an extent within the V Model during the verification phase of the system.
- Changing requirements can be implemented faster. This is due to the short time boxed cycles in SCRUM and the greater collaboration between the developers and the customers, customer feedback is essential within SCRUM and can be used to initially reduce the changing requirements or to minimize the change required
- From a business perspective, the cost of change is greatly reduced due to both the lack of repeated stages and being able to communicate between the client and development teams more openly for requirements.

However, there are also disadvantages to using SCRUM and some types of projects that do not work as well in this more fluid management system. Some of these disadvantages are as follows:

- For smaller projects SCRUM is harder to implement and stick to due to the differing structure
- Reduced documentation on the product as the focus is more on the product and getting it to the client
 - This however can be mitigated to an extent during a retrospective whereby the stakeholders can be involved and demos of the product can be given.
 - Also sprints of documentation can be planned in should documentation become an issue.
- “Team work is highly essential” (Mahalakshmi & Sundararajan, 2013)
 - Due to the team centric nature of SCRUM, the team members will need to function well in a team otherwise this can cause issues for the overall performance of the team, however this problem is inherent to any team based development system, though due to the self-management aspect of SCRUM can be more apparent.
- As stated previously team work is essential, should rifts form within the team, this can be detrimental to progress, however this can be handled within the daily scrum and retrospectives.
- SCRUM is harder to fit alongside some project management structures, for example PRINCE 2, due to the lack of imposed structure
 - However, some of the discussed models can be used within an iteration and could therefore be implemented alongside SCRUM within the workplace.

As previously stated, depending on the project in question and the industry, models like the Spiral model and Waterfall fit better than SCRUM, due to having a fixed specification and safety requirements, for example military contracts where certain requirements are forced onto the company developing the software. Also for software support teams, SCRUM can be harder to implement due to the nature of support, however other Agile methods like Kanban or Scrumban, that adjust some of the principles can be used in place in this situation to allow for better management.

3.6 Tools for managing SCRUM projects

There are a range of different tools that can be used to manage SCRUM based projects. These range from list-based systems like Trello, which provides a service that can be used as a team SCRUM board, to a more feature rich system like Microsoft Team Foundation Server (TFS), which, alongside its source control system, provides various project management templates and reporting for these. Despite the varying range of tools, some commercial (e.g. Visual Studio Online (TFS), ProjectCards⁴, ScrumDesk⁵), some open source (e.g. Digaboard⁶, Explainpmt⁷ and Agile Express⁸) this report will focus on both Trello and Team Foundation Server as these two tools have had the most impact in terms of design decisions for this project along with personally being well versed in both systems.

3.6.1 Trello

Trello is a freemium list-based cross-platform web-based utility which launched in September of 2011 with apps for the web and iPhone (Trello Inc., n.d.).

“Trello is the free, flexible, and visual way to organize anything with anyone” (Trello Inc., n.d.). Trello’s scope is targeted towards anyone, rather than specifically businesses looking at implementing SCRUM, and therefore does not provide the reporting elements of a business project management solution like TFS, however this comes as a much cheaper alternative given that anyone can make a basic account for free.

When you first sign into Trello you are greeted by default with the Welcome Board:

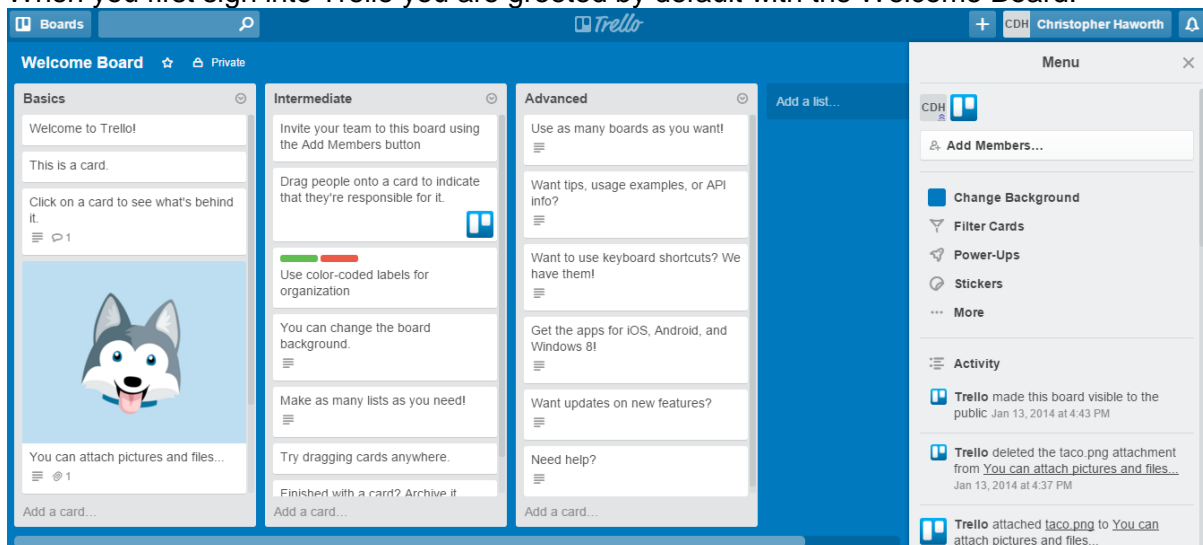


Figure 10. Trello welcome board (Trello Inc., n.d.)

Due to the broad scope, Trello is designed to be easy to use, and therefore the welcome board gives you an introduction to Trello’s features. Each one of the cards that are displayed (Items in the list) can be clicked on to view more details. These can also be dragged around into different lists, therefore creating a SCRUM Board, where you can have 3 lists (Not Started, In Progress, Done).

⁴ ProjectCards Home Page: <http://www.projectcards.com/>

⁵ ScrumDesk Home Page: <http://www.scrumdesk.com/>

⁶ Digaboard GitHub: <https://github.com/Magentron/Digaboard>

⁷ Explainpmt GitHub: <https://github.com/explainpmt/explainpmt>

⁸ Agile Express Home Page: <http://agileexpress.sourceforge.net/>

Trello also gives you the option to create teams, and have shared boards between these teams. This gives you the option to assign a card to a person, or multiple people for example:

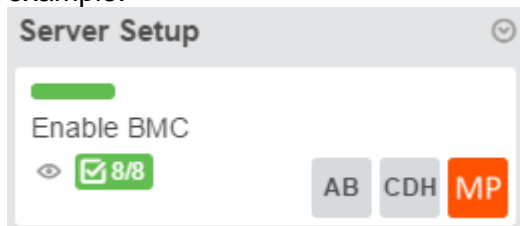


Figure 11. Example of multiple people assigned to a card (Trello Inc., n.d.).

Because of Trello's ease of use and more public aim. It is not in itself a SCRUM management tool, however it can be made to function as one. However, by doing so, the tradeoff is that the reporting element of SCRUM is not available due to the broad scope.

3.6.2 Microsoft Team Foundation Server

In contrast to Trello, who's only focus is a list-based tool which can be used to implement SCRUM, Microsoft Team Foundation Server is an enterprise grade source control system with integrated project management features that include both traditional software development lifecycles and Agile life cycles like SCRUM.

Due to this, again in contrast to Trello, despite TFS being a web based tool, it requires you to purchase a license and install locally within the organization, or use a freemium version hosted by Microsoft called Visual Studio Online, therefore the scope of this solution is more focused towards businesses that have an existing Microsoft server environment due to its requirements on MSSQL and Windows Server with Active Directory.

When you first start TFS as an administrator you are greeted with the following screen:

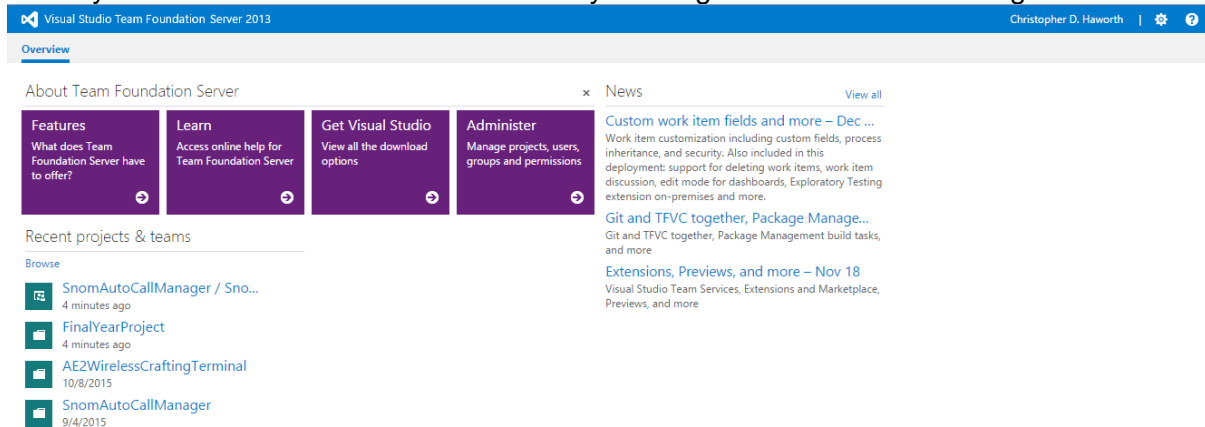


Figure 12. Typical TFS welcome screen

This will be a little bit different as I have existing projects within this TFS instance.

To get started with TFS one will need to have Visual Studio installed on a machine in order to create a project as there is no way of creating one from the web interface unlike Trello which is completely managed from the web interface.

However, in contrast to Trello, because this system is aimed at businesses, the system is generally not used for small personal projects due to the requirements stated above. Though in contrast, due to the business nature of TFS it does have reporting capability for SCRUM and therefore is a more rounded tool for SCRUM management for example, Burndown and

velocity charts can feed into forecasting for future sprints:

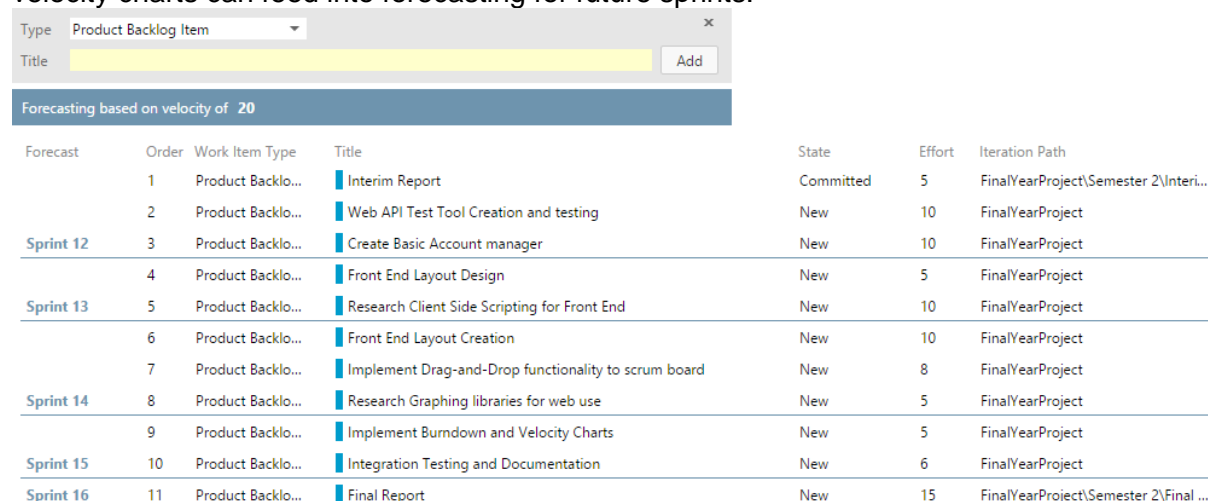


Figure 13. Example of forecasting in TFS

In contrast the two tools both have their place depending on the scope of the user's aims, however for larger projects, TFS has the upper hand due to it being built as a project management solution. The downside to this is also that TFS is targeted at software development companies due to its source control element, whereas Trello is available to anyone as a Collaborative List Manager.

Within this tool, personally the aim is to encapsulate the lightweight side of Trello while bringing in the reporting elements of TFS to allow it to be used as a standalone planning tool that isn't tied to any source control system.

4 Technical Development

4.1 Design

Initially a set of user stories based on the objectives of the project were devised, these are as follows:

1. As an Administrator I want to be able to create projects for the team to use
2. As a stakeholder I want to be able to create new items on the Product Backlog
3. As a stakeholder I want to be able to set the priority of items on the Product Backlog
4. As a team we want to commit items from the Product Backlog to the current iteration
5. As a team member I want to create a task for a Product Backlog item on the Sprint Board
6. As a team member I want to be able to move tasks across the Sprint Board into done
7. As a stakeholder I want to be able to view the projects status via reports on the amount of completed tasks
 - o Burndown charts
 - o Velocity graphs
8. As a third party developer I want to be able to interact with the data created on the website within my own application

Following on from this, a specification was then constructed for the project as follows:
Create a multi-user Web based SCRUM management tool with the ability to have multiple projects per instance that Administrators can manage for multiple teams within the organisation.

The application will need to support access by multiple different users to facilitate the management of the projects, these different users will have different levels of access depending on what their roles are within the team, for example an external stakeholder will only have very limited access to the application whereas the Team Administrator will have unrestricted access to the platform.

The system will require to visualise different elements of the SCRUM process (Product Backlog, SCRUM Board) which multiple roles will require different levels of access to, for example an external client will need to be able to view the Product Backlog in order to prioritise it for the team.

Following on from this, the teams need to be able to create both product backlog items as well as tasks once the items are committed to a sprint/iteration. The application will require per project a way of planning out the sprint/iteration times and keep track of when they are to facilitate planning.

Another key component that is required is the ability to view certain reports like burndown charts and velocity graphs to be able to ascertain how well the team is doing with a project and to estimate if they will finish the project on time, this will need to be per sprint and per project phase.

The system will need to facilitate external 3rd party programs through the use of a Web API that developers can hook into to create applications that can interact with the system.

The Web API is the core component of this project, therefore, as I will go into more detail about during the project management review section of this report, I have spent the majority of Semester one solely working on the API in a test driven development structure.

4.1.1 System Layout

Figure 14 shows the architecture layout for the application and shows that the primary hub for the application is the API itself.

The API is a series of stateless methods which return JSON objects that allow for interaction with the data access layer (described in section 4.1.2). This is also the only element that can directly interact with the database.

On top of the API and the User Interface sits the Account Management system. This uses Active Directory to authenticate users that connect to the system. Active Directory is also used to provide half of the role provisions that the system has, defined by the groups that the user is present in within the domain. The other half of this is provided by an explicit role system defined within the account manager (more information on this in section 4.2.2).

The Web User Interface sits at the top of the system as the primary method of interacting with the system. Each user will need to be authenticated through the Account Management system, this is the same for both the API and Web UI, and each section has limiting usability depending on what role the user has within the system.

When the system is hosted within IIS, two methods of access are exposed (the API and Web UI). The Web User Interface calls into the functions of the API to provide database operations and data retrieval. This separation could lead to the API and the web user interface being separated into different applications to host on different servers depending on the workload for the system in the future.

The Following sections will describe the initial designs for each of the components of the system (Database, API, User Interface).

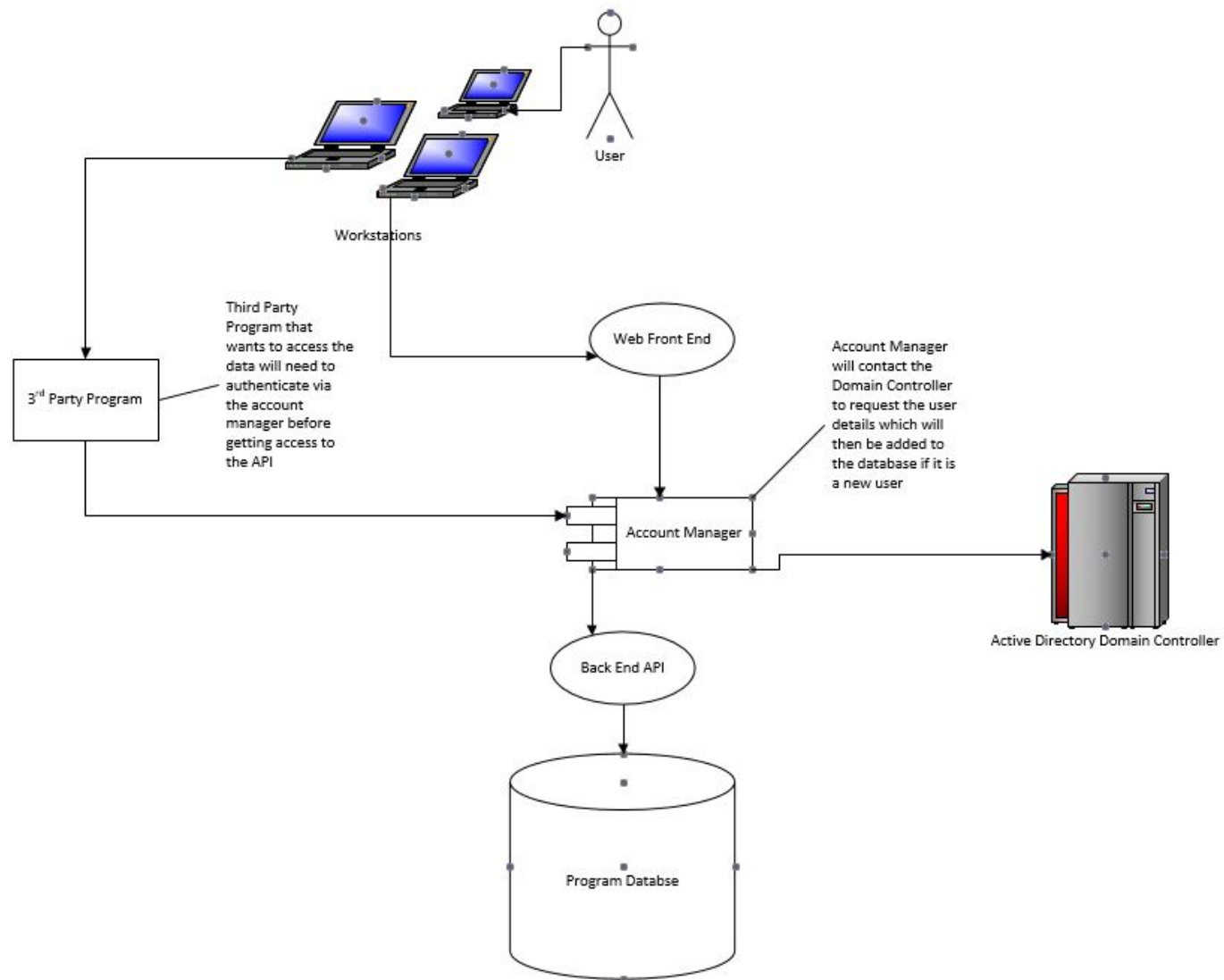


Figure 14: System Architecture Diagram

4.1.2 Database Design

Following on from the specification I proceeded to build the following database design, this also served as a point of reference for designing the main API controllers as each controller has been mapped to a specific table for functions related to the data there.

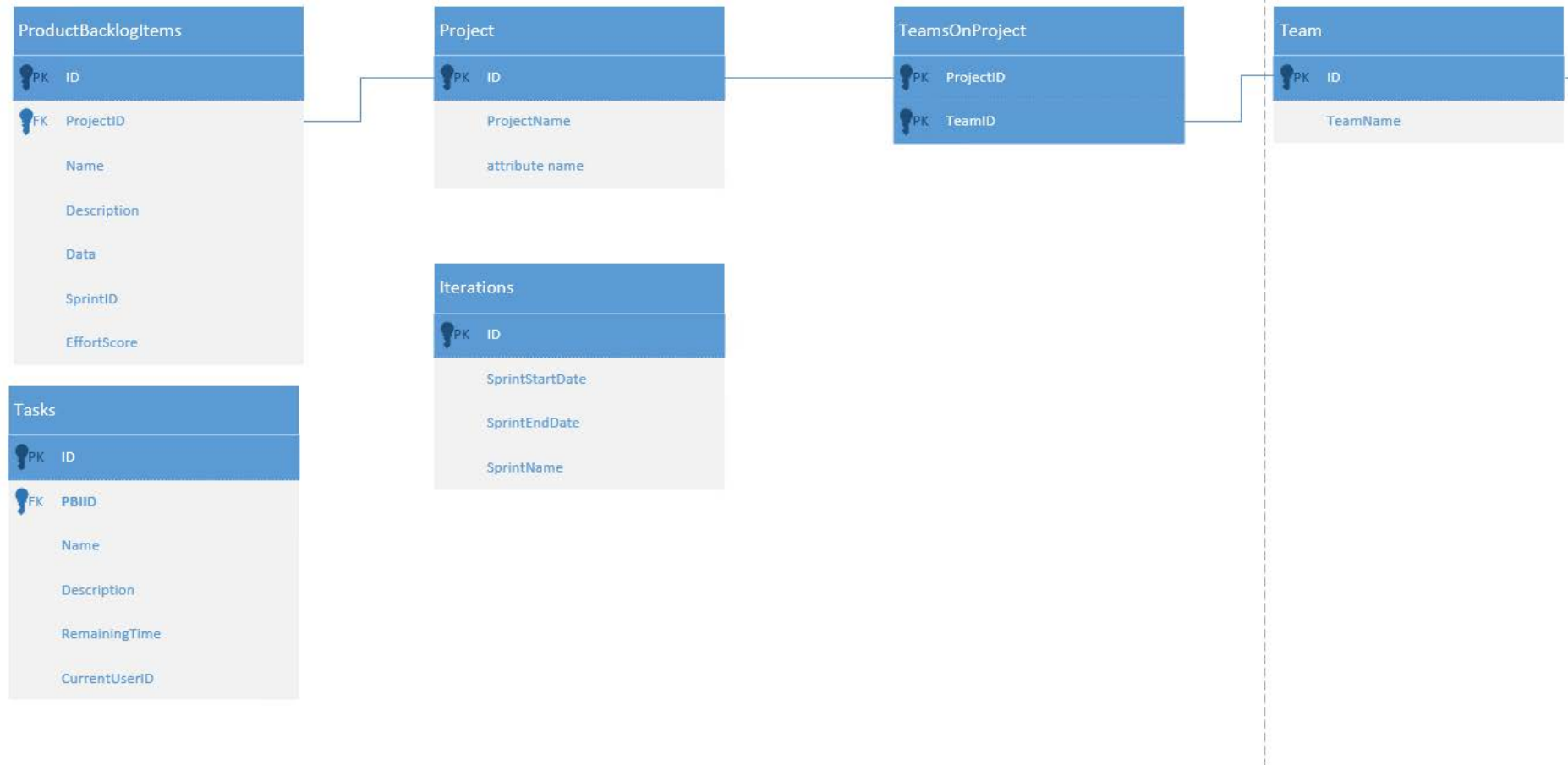


Figure 15. Initial Database Design Part 1

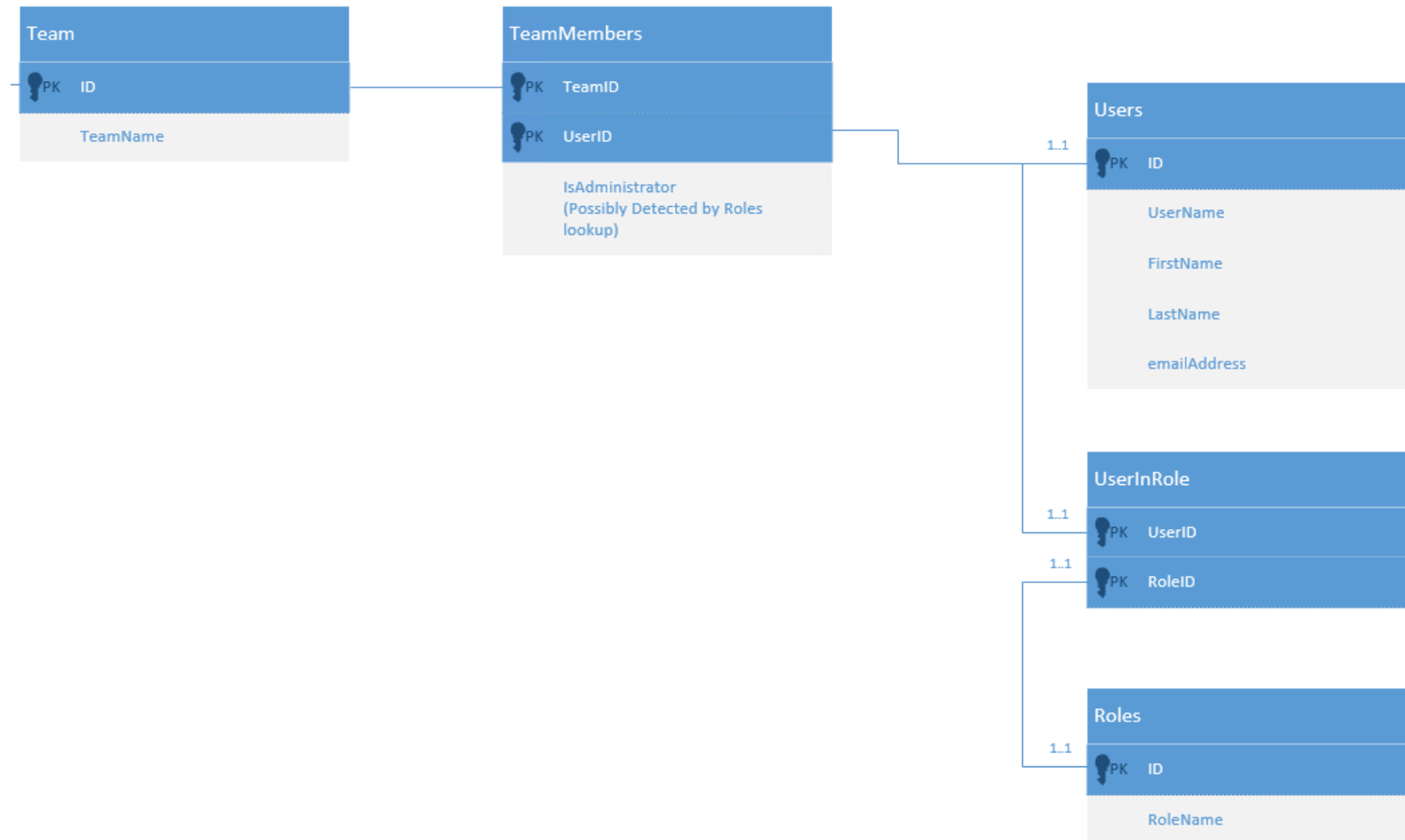


Figure 16 Initial Database Design Part 2

This database design, shown by the above two figures, has mostly remained the same since the start of the database, however due to an issue in entity framework with composite primary keys and foreign keys referencing only part of the PK, the keys on some of the tables have had to be adjusted to allow for constraints to be placed for Entity Framework navigational properties, therefore the database now looks like the following:

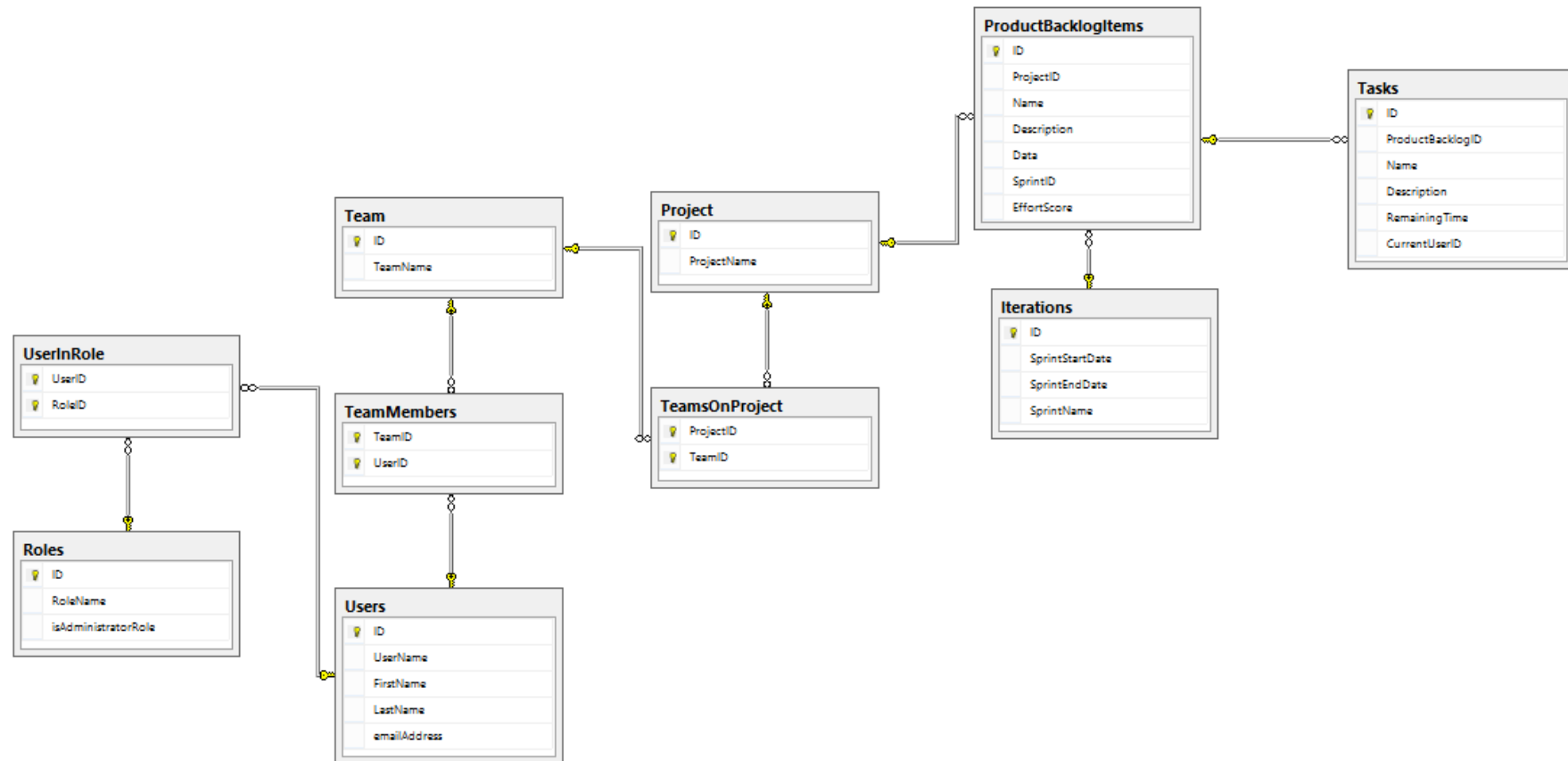


Figure 17. Database layout after changes due to issues with EF. Exported from SQL Server Management Studio.

4.1.3 API

With regards to the structure of the API itself, there is a controller that is mapped to each of the main tables in the database to provide access to the required components of the data

As for the test framework, the integrated test framework present in Visual Studio for unit testing was chosen. At present there are tests for the data access layer that provides the methods for the controller to hook into for manipulating the database.

It has been built like this to allow for integration into mocking frameworks or separating the database components into a shared library that can then be imported into other applications for working with the database directly. This can also allow for testing with mocking frameworks, however this has not currently been implemented.

During the initial design phase, a decision was made towards using Active Directory for the account management for this project. This does however limit the scope towards business due to the requirement of a functional Active Directory domain. Due to the target audience of the system being aimed at businesses that operate under the SCRUM Framework, this limiting scope does not have too much of an effect on the overall project.

4.1.4 User Interface

For the user interface the initial plan was to use a series of partial views and dialogs to provide the features required for the system with the basic layout like the following:

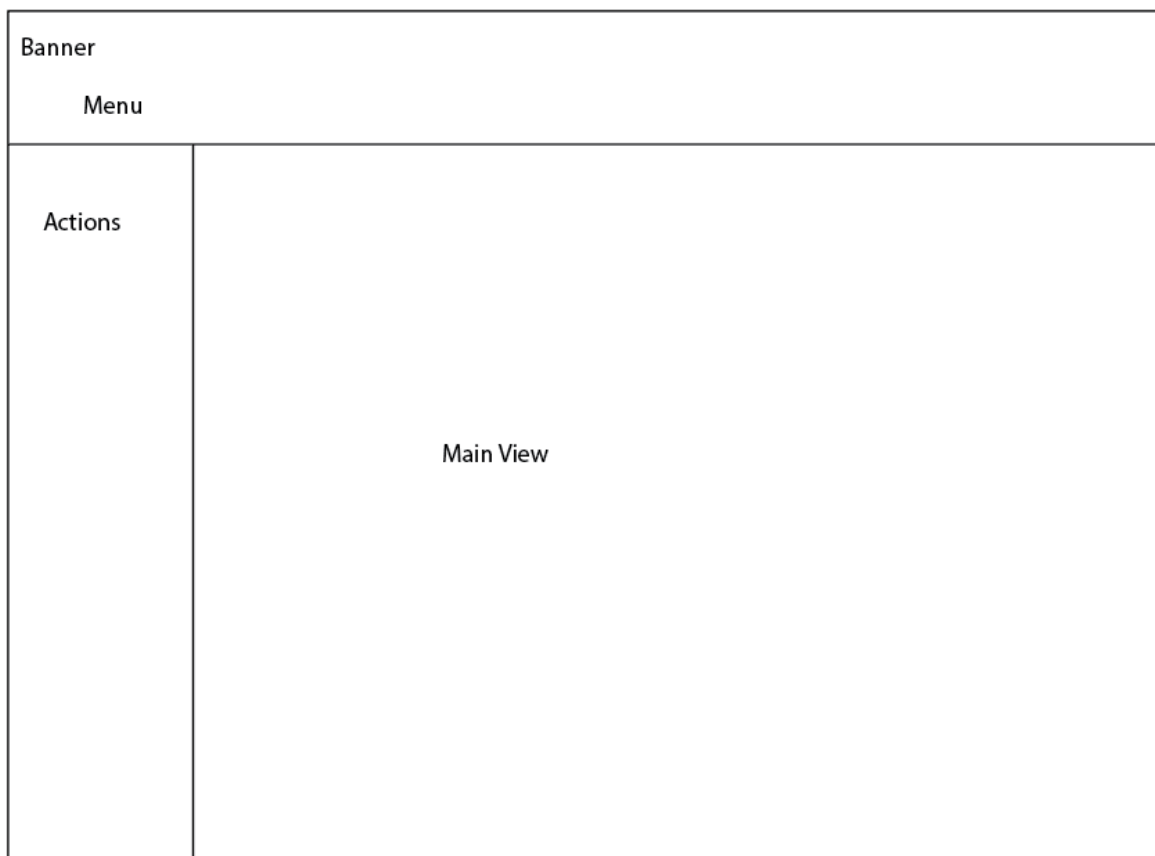


Figure 18. Initial UI mockup

In this design, the first screen that will be displayed in the main view will be the current iteration Scrum board so that the user can get a quick glance straight away without having to

navigate through multiple menu interfaces as to what they are working on and what is in the pipeline.

During the start of the second semester this design will be refined before implementation of the front end.

This element of the design is influenced by the ease of use of Trello vs TFS, whereby for the project an aim is to incorporate some of that ease of use but with more powerful features like the reporting elements.

The front end will use a large amount of interactive features provided by jQuery, such as drag and drop cards on the Scrum board to move the task into the next stage. jQuery will also be used to provide the information dialogs for editing specific tasks without leaving the main page. This approach is similar to both TFS and Trello which both implement the drag and drop dialog based interface.

4.2 Implementation

This section will be split between the three components of the system, the Web API, the Account Management System and the ASP.Net User Interface.

4.2.1 Web API

The Web API component in the system is the primary method of accessing the underlying database and forms the core system of the application as shown in figure 15. This uses a series of classes that represent the different models in the system to interact with the dataset.

Communication to and from the Web API is handled via JSON, this was chosen over using XML due to the integration with jQuery in the user interface section of the project. Further information as to the responses and requests to the API is available in Appendix E.

The API Itself is split into a series of 8 classes that map over to the base models present in the system and communicates via the data access layer to the database.

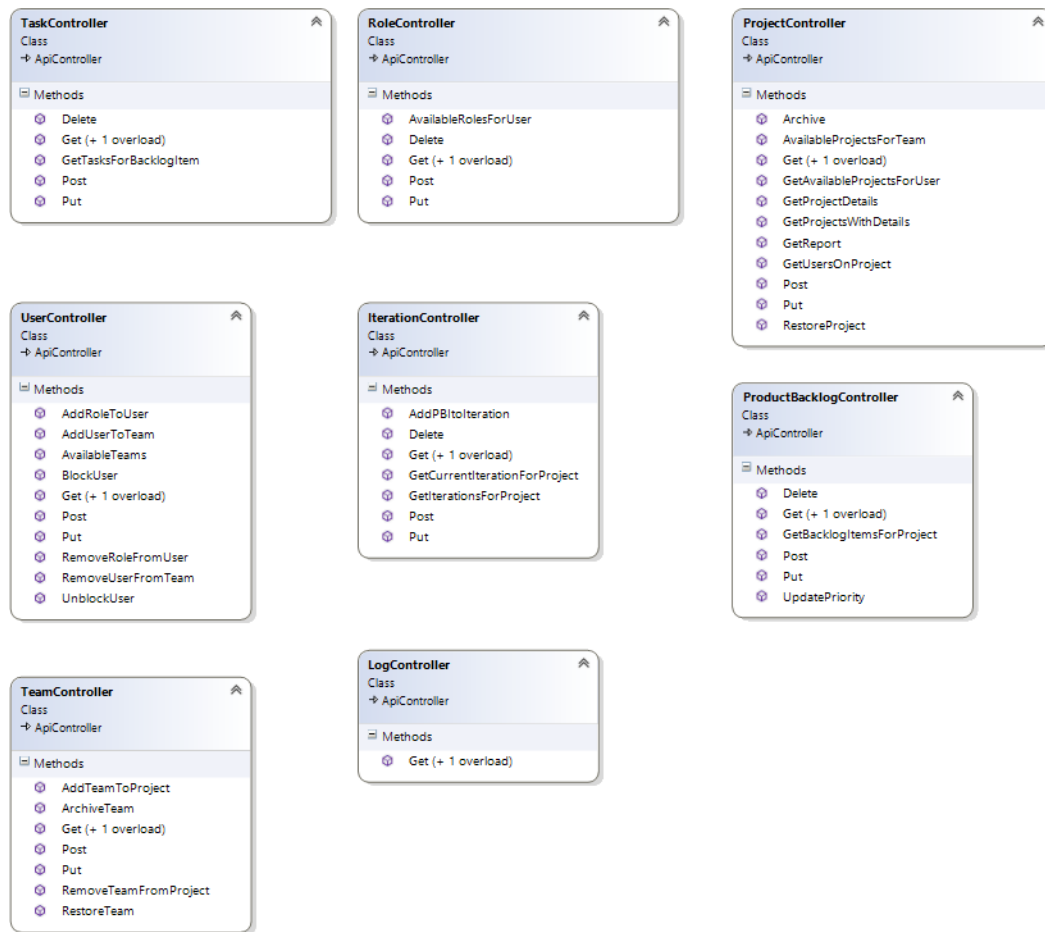


Figure 19. API Class Diagram

Figure 19 shows a class diagram of the API structure. As stated previously these classes (excluding the Log Controller) map to the core tables of the database and provide CRUD functionality on these tables. The API also has additional helper functions to provide access to commonly required elements of the User Interface, for example the `GetCurrentIterationForProject` method of the `IterationController`, which returns the current iteration (located by date) of the selected project passed as an integer id to the method.

Each method in the API that returns data from the database returns it in the form of a Data Transaction Object to avoid issues with circular references during the serialization process, these objects are part of the data model, shown in figure 20 over the page.

The data model of the project moves away from the MVC pattern to an extent as there is some logic present in the model itself to allow for data conversion, this is provided by the interface `IModelContext` which all of the core model classes implement.

This interface has three methods defined:

1. `GetDetails`
2. `ToDTO`
3. `UpdateItem`

Starting with `ToDTO`, this method will return the appropriate model transaction object for serialization, these objects have any relationships removed from them to allow for the

serialization to take place so therefore will only return the data that is actually present in the underlying database and none of the virtual data that is added in by Entity Framework.

The GetDetails method extends the existing data transaction objects to add the virtual collections/objects that are present due to the relationships in the data tables, and to avoid the circular reference issues will return the relevant DTO's for these objects.

Both of these methods are used in the Generic List extensions provided by the list utilities class which is also shown in figure 20 below the interface. These two methods are the equivalent of a ToList function on a collection of model objects. An example use case of this would be the following: *db.GetProjects.ToDTO<ProjectDTO>()*; Which would get all the projects from the database in a list then convert that list into a ProjectDTO list for sending to the client. The same can be used for the GetDetails<>() method to return a list of ModelDetailsDTO. Both of these methods extend IEnumerable<T>.

Finally, UpdateItem is called to allow the model to update the fields itself, this is to save repeating code if the model would need to be updated in multiple places. This is used primarily in the Data Access Layer class.

Each of the API methods also require authentication to use them at varying different levels, this is supplied by the Account management system which will be covered in more detail in the next section. For more information on the different authentication levels please refer to the Technical Documentation in Appendix E.

Due to various changes during the development of the project, the database went through another iteration of changes with the final schema being shown in figure 21.

The first noticeable difference is the addition of the log table. Towards the end of the development of the User Interface and finalizing the API, a logging system was implemented which logged errors, warnings and debug messages to this database table to then be viewed from a page in the User Interface, along with via the Web API. This was handled in this manner to centralize the data storage rather than writing log files to the server's storage drive which over time could get quite large in size.

Secondly there were changes to the UserInRole table, which will be explained in the section on the Account Management System, as well as linking iterations directly to product backlog items to facilitate features of the User Interface.

Despite the circular reference present between project, product backlog items, and iterations, the previously explained functions of the IModelContext interface avoids issues with serialization due to the data transaction objects. This is also highlighted in figure 20 when looking at the DTOs for the classes in question.

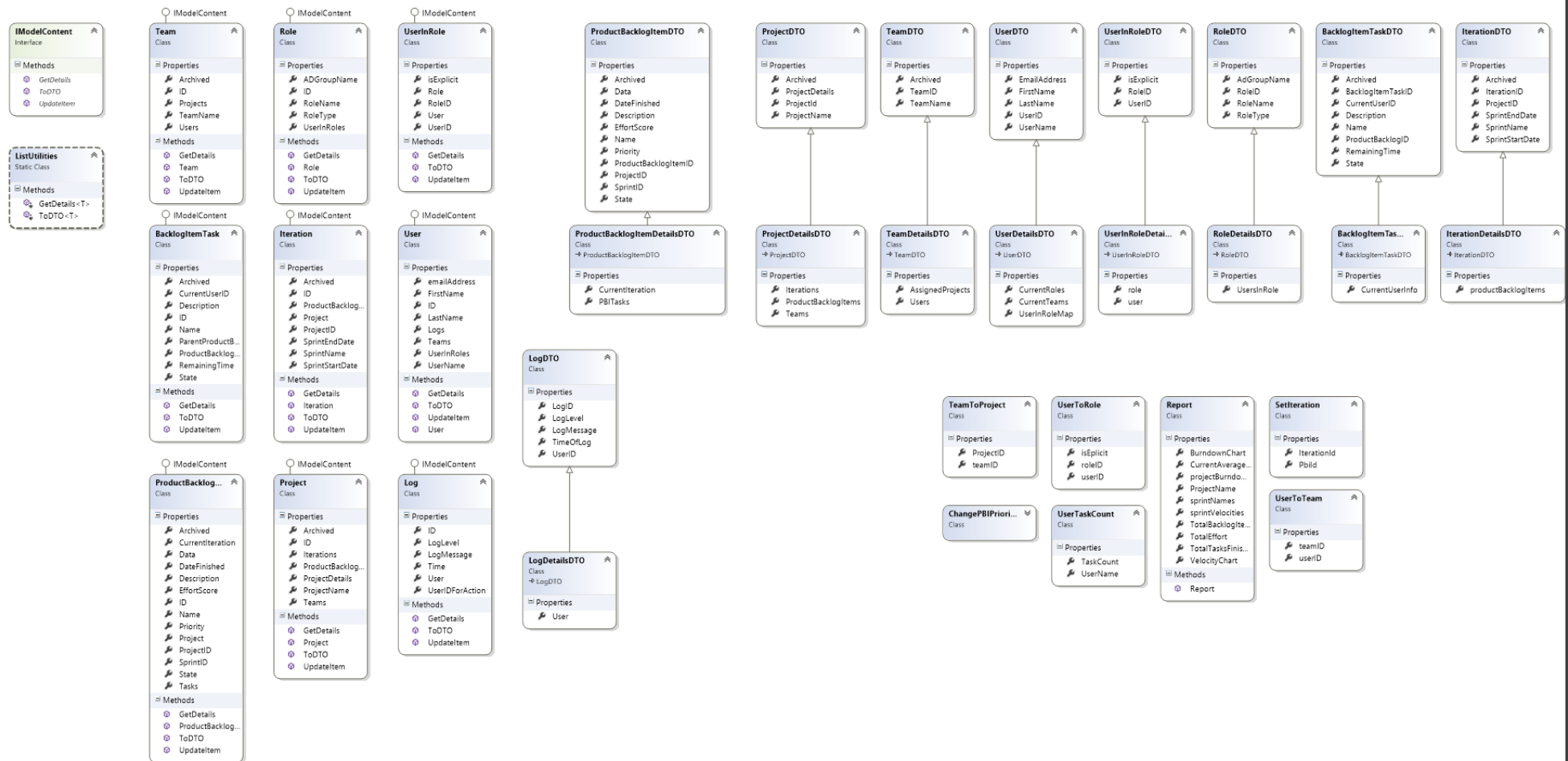
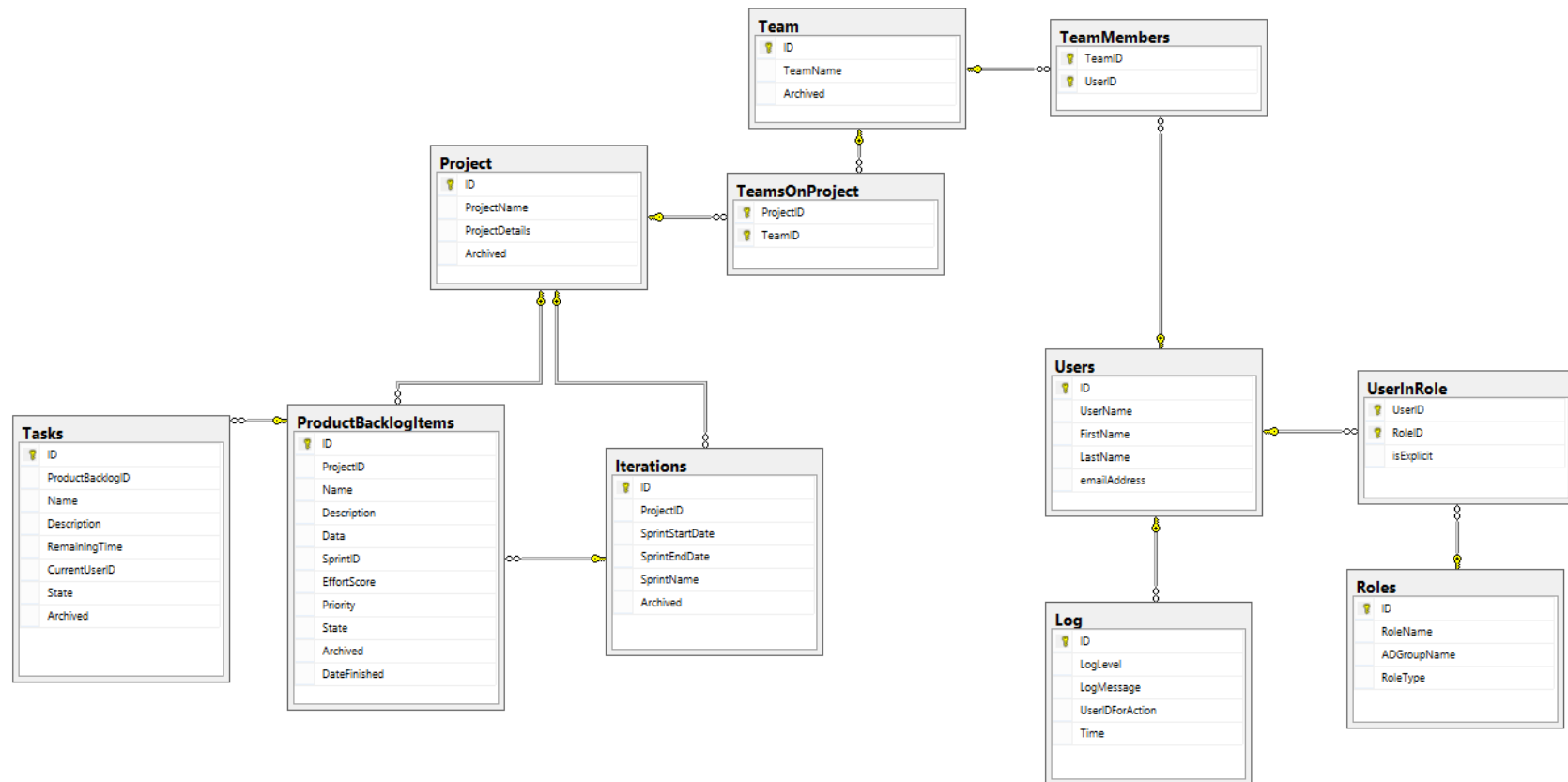


Figure 20. Model Class Diagram

*Figure 21. Final Database Schema*

4.2.2 Account Management System

The Account Management System sits between both the API and the User Interface as well as on top of the User Interface and provides User Authentication and role provision from Active Directory as well as providing its own role provisioning and user blacklisting system.

This is handled through a series of 3 filters that handle different elements of the Authentication process, these are as follows:

1. User Authentication – Base Class
2. Team Authentication – Checks if a user is valid and is a member of a specific team
3. API Authentication – Extends User Authentication to allow for authentication at the API level

Both the Team and API authentication systems check the user's authorization using the user authentication system, which in turn checks the required roles within both Active Directory and the system's own role provisioning system.

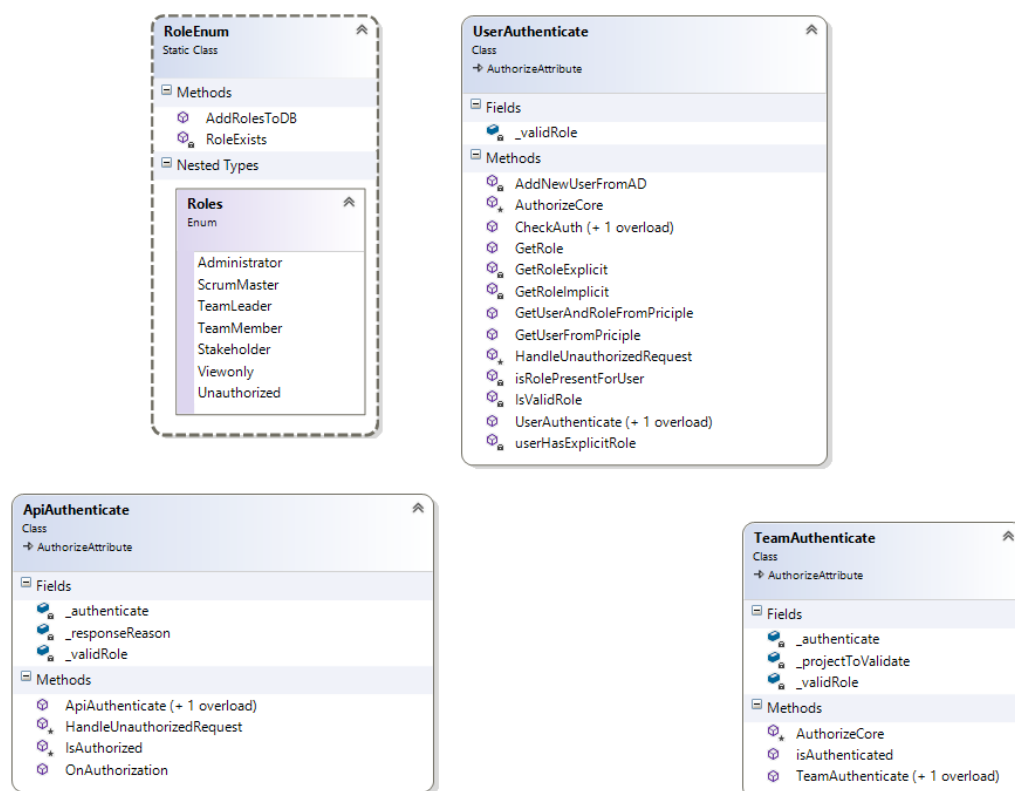


Figure 22. Class Diagram of the Account Management System

Due to differences in namespaces between MVC controllers and API controllers in the .NET framework, two different base classes needed to be extended for authenticating the API and User Interface.

As stated previously both the `ApiAuthenticate` and `TeamAuthenticate` filter call back to the `UserAuthenticate` filter to confirm the user's level of access to the initial element of the system initially, then in the case of the `TeamAuthenticate` system, will then check to see if the user is either a SCRUM Master or Administrator in which case they will be automatically granted access, or if the user is a member of the team on that project.

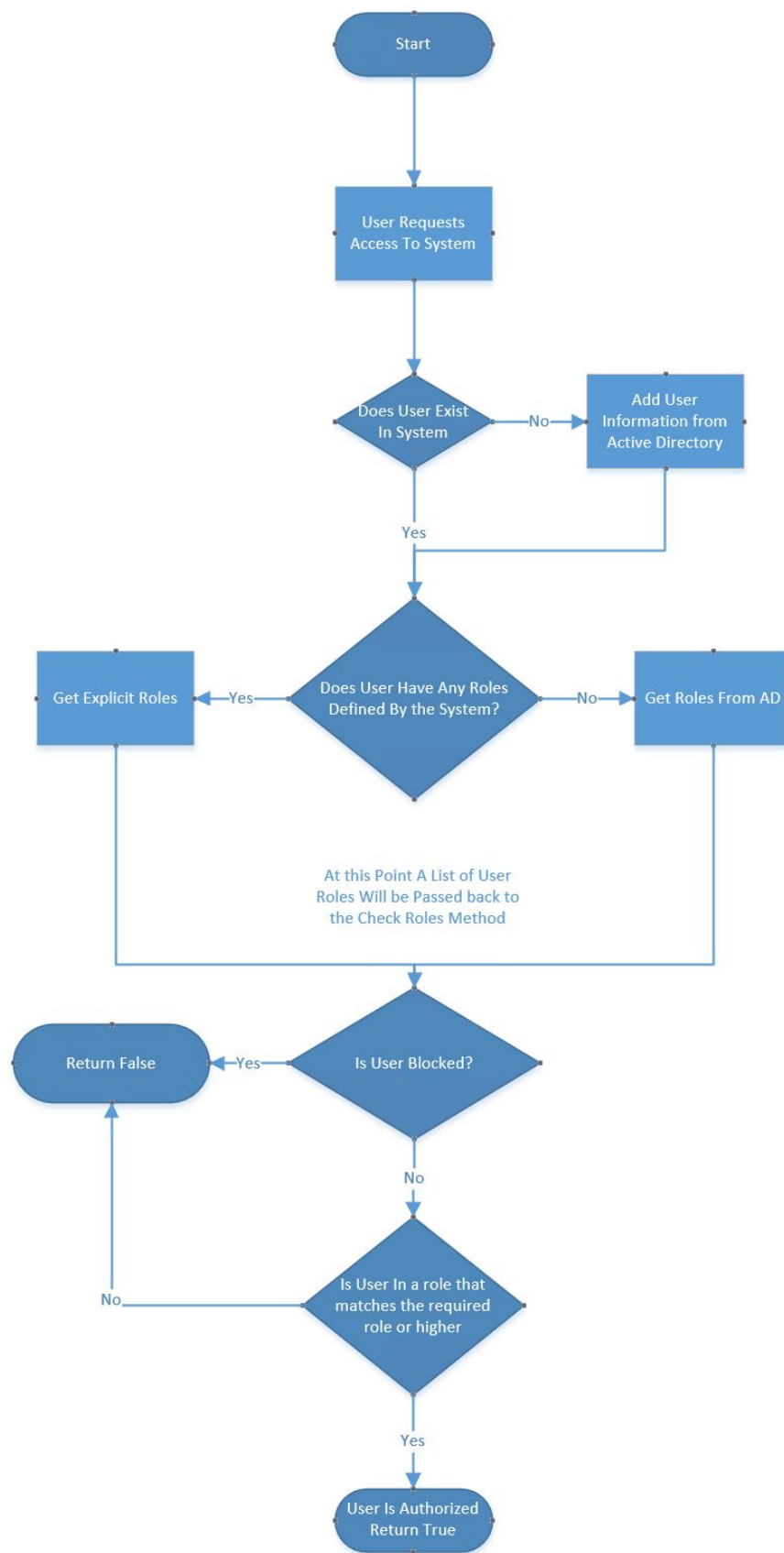


Figure 23. Flow Diagram of Authentication System

Figure 23 shows the process in which the Account Management System checks to see if a user is both a valid user and if the user is authorized to view specific content in the system.

When the user requests access to any protected content, initially the system will check to see if it knows the user first (if the user entry is present in the database), following on from this, if it doesn't know the user, the domain controller will be queried to request the user information to add to the database, then will continue to check the user roles in the order of the flow diagram above.

Due to this role management system the database required some changes to accommodate the required information, further changing it from the initial design as shown in figure 21.

The primary change here was the inclusion of the isExplicit bit in the user in role table, which allows for defining roles within the system, if this bit is true, then the role has been granted to the user within the scope of the application, otherwise the role is implicitly granted via the Active Directory groups specified in the Roles table.

The system was designed in this way to offload work from enterprise system administrators, for example, if a user required permissions within the application for a short space of time, an administrator of the system could grant the user the required role within the scope of the application rather than having to modify the user's domain permissions. This also allows for the blocking of users without restricting their access to other domain resources, say for example if the user moves into a different area of the company that doesn't require access to the system, the user could be moved into the unauthorized group in Active Directory, or an administrator in the system could block the user within the system, therefore not requiring any changes in the domain.

4.2.3 ASP.NET Web User Interface

The ASP.NET web user interface is the primary method of accessing the application, it consists of 4 Main Sections:

1. Home/Scrum Board
2. Projects
3. Reports
4. Administration (Only visible if you are logged into an account that has a permission level of SCRUM Master or higher)

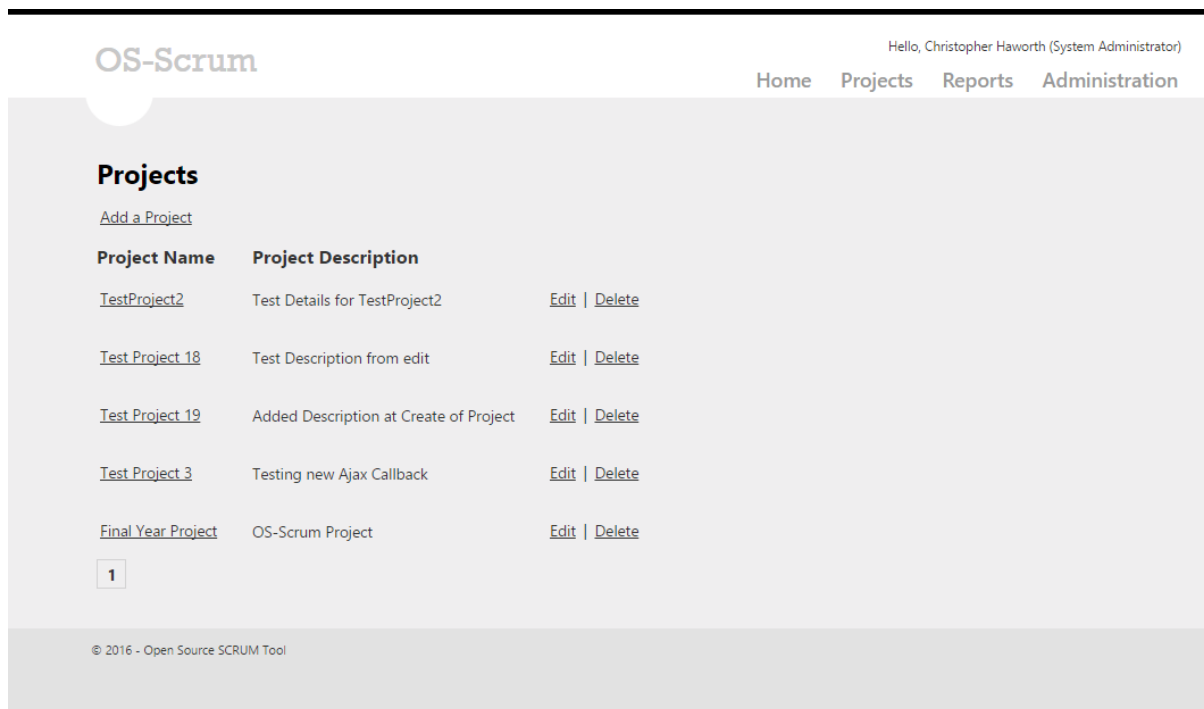


Figure 24. Example of the User Interface

Figure 24 shows an example of the web interface of the project (currently showing the project page). It uses a bootstrap based layout, similar to that of the default asp.net template. This design was used to try to remove some of the hassle of developing a layout from scratch, however could be overhauled at a later date to be more lightweight.

The user interface uses a large amount of jQuery to accomplish most of the tasks in the system aside from the initial displaying of some of the data. It makes heavy use of jQuery dialogs for any kind of add or edit in the program e.g.

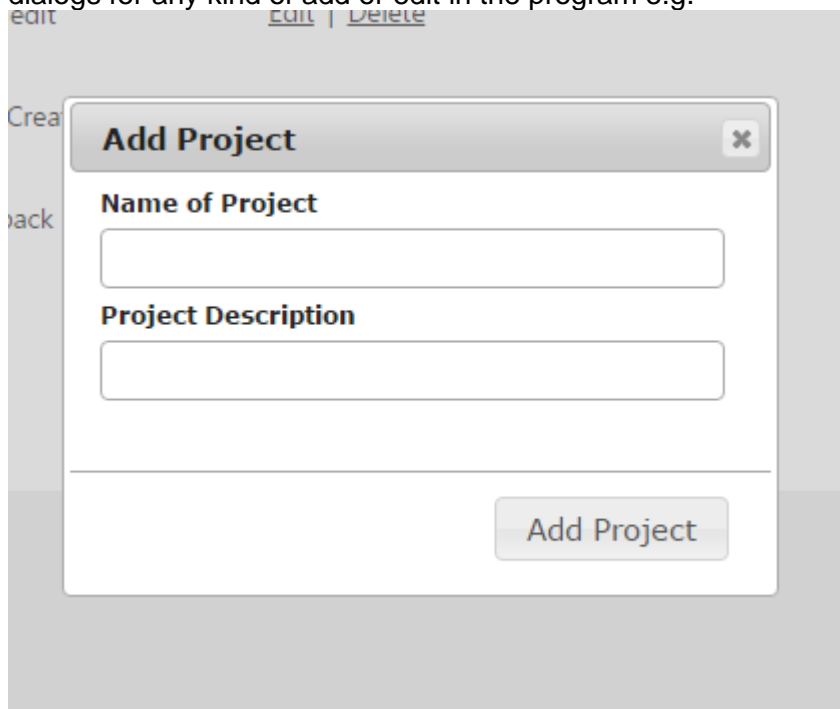


Figure 25. Example Add Dialog

For the dialog system, the choice of dynamically constructing the form was chosen over building new views to minimize the amount of requests made to the server and offload some of the work to the client. The dialogs all use a custom java script object helper designed for the project, to allow for an object oriented way to create the required components of a dialog.

Each model that required some form of editing has a dialog "class" associated with it in the client side scripts, so that if a different page required an add project dialog for example, all that would need to be added to the click event of the link is: *var AddProjectDialog = new ProjectDialog("add"); AddProjectDialog.generatedDialog("open");*

This system was implemented also to reduce the amount of time spent making new dialogs as code was implemented to build the div to render in an OO manner e.g.

```
var AddData = {  
    dialogID: "AddProject-dialog",  
    dialogTitle: "Add Project",  
    dialogItems: [  
        new TextBox("projectNameTextBox", true, "Name of Project"),  
        new TextBox("projectDescription-TextBox", true, "Project Description")  
    ]  
};
```

When passed to the generateDialog method along with the dialog settings this would build the dialog shown in figure 25.

All updates/additions to the data are done via ajax called back to the API so that there is a single point of access to the dataset. This was built this way due to how much of a core component the API is within the project, so rather than duplicate the existing code for the database, the API was used to handle the updates, therefore the Web Interface behaves like any other application that would have access to the API.

Moving through the User Interface, jQuery Sortable is used for setting the priority of the product backlog items in the project. When an element is dragged into a different position in the table, the backlog item ids are collected into an array and passed back to the API to update the priorities based on the new position in the array. A couple of options were considered including this one, one of which would have been a series of multiple requests to the API to handle updating the priority. The array method though was chosen due to it being a lighter weight and minimizes the amount of requests made to the server.

With regards to the reporting objective of the project, Chart.js was chosen for the graphing library due to it being open source and allows for interactive charts to be created and rendered client side, which means the server does not have to build and send over a static jpeg or png with the chart.

One area which could have been done differently, and if more time was available, is how the charts are handled in the system. Currently the chart data and colour is decided server side via a .NET wrapper constructed for the project and passed back through via the API to the client to display. In retrospect this was not the best approach to take for this, and does not fit the overall theme of the project being open source and easy for 3rd party applications to interact with the system. Instead of this what should have been done would be to send the data in a generic format not configured for Chart.js, then build the actual chart from that data.

4.3 Testing

Throughout the project most of the testing was ad-hoc based on the feature that was being worked on at the time, and then ad-hoc integration testing with regards to the User interface,

which in retrospect was a bad way to go through doing the testing, and proper test documents should have been created and used after each major feature was implemented.

The project started with a test driven development approach, and the data access layer was initially unit tested. However, since a mocking framework was not implemented, the testing required a clean database and was therefore not very reliable.

During the implementation of the API, testing proved to be quite difficult using various tools to pass the data required to the API, these included using the likes of Fiddler and telnet sessions. Towards the end of the API development a small Windows Forms test application was created to run an integration test on the API.

Throughout the development of the application, despite the ad-hoc nature of the testing, the testing was carried out in a test environment running Windows Server 2012 R2 along with a separate SQL Server (MS SQL 2014), both set up in a domain environment to mimic the target environment for the application so issues that wouldn't have been picked up on a development machine could be tested better in the target environment of the application.

For the scope of testing, a series of dummy accounts and groups were created within the Active Directory domain as shown below:

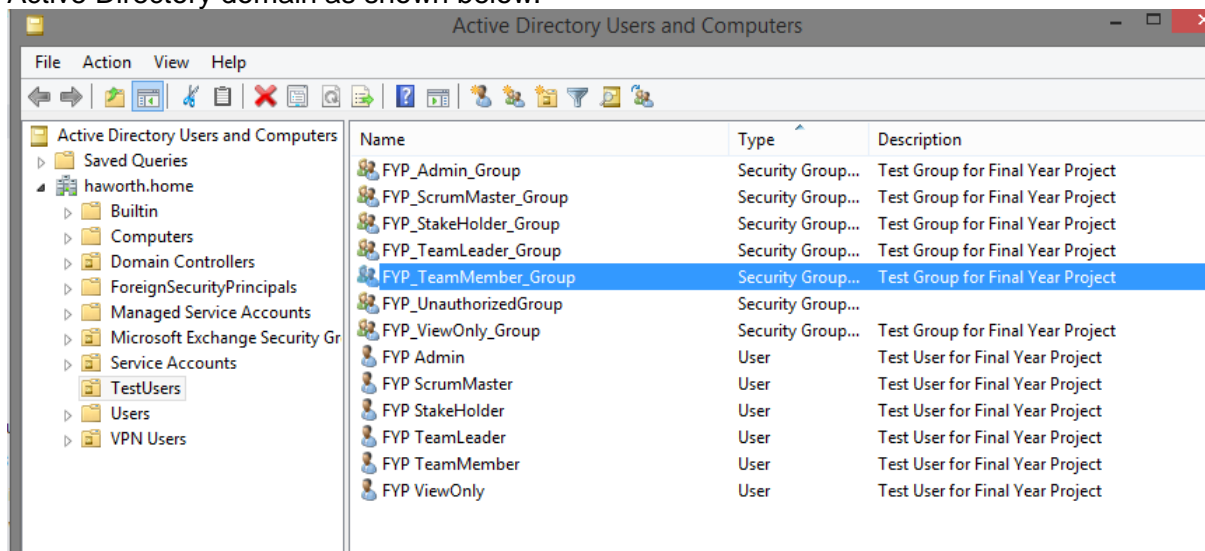


Figure 26. Active Directory Configuration for testing

Each of the users in Figure 26 were mapped to the respective groups, this allowed for testing the application at different levels of authentication without having to modify the code. However, testing at different levels with the same user is possible by assigning explicit role mappings within the database or administration panel.

As previously stated some unit tests were implemented in the project. 5 tests were initially put in place for Projects on the Data access layer, and the initial structure was put in for 5 more on the Backlog item section of the DAL, however those tests were not finished off. For the tests that were completed, Overall code coverage on the project was 8.64% with 4.18% coverage on the DAL and 38.7% of the Models. Despite the low test coverage in the unit tests, testing still took place, in more of an ad-hoc method, whereby each feature was tested in a manual integration test within the prepared test environment. By doing this it also enabled testing of general performance, as the test environment was configured in such a way that the specifications were low (Core I3-4010U as the Hyper-V host, VM's Configured with 2 Virtual Cores and for IIS, 1GB of ram, SQL, 2GB of RAM).

5 Evaluation

5.1 Project Achievements

In this section, a breakdown of the objectives will be discussed and how the objectives have been met or could have been improved upon, and if they were not fully fulfilled, an explanation will be given as to the issues encountered while working on those elements of the application.

Looking back at the initial objectives set out for the project, the main objectives have been mostly fulfilled.

Objective 1: Research Existing Solutions – Achieved

For this objective research was conducted into existing applications that meet the initial aim of the project, Multiple different tools were found ranging from PHP based applications to Ruby + MySQL open source/cross platform toolsets.

This objective also made the decision to focus the project to an enterprise setting given how the user management system has been implemented using Active Directory, in a similar manner to Microsoft Team Foundation Server.

The research focus for existing applications focused on both Trello and MSTFS which restricted the scope of the objectives somewhat by not taking other solutions into detailed consideration.

Objective 2: Database Design – Achieved

With regards to the database design, it has undergone a couple of changes from the initial design phase, however the structure has remained mostly consistent throughout the project. This objective also links into Objective 4 (building the back end of the application) as to the implementation phase of the database.

Objective 3: Authentication Investigation - Achieved

Investigation into what authentication system to use, along with the initial research phase, directed the project towards an enterprise setting using Active Directory for user and role management. Moving into the implementation of this, a multi-role provider system has been implemented in the system that allows for role provision via the domain and also within the scope of the application by the app administrators. This is used by both the API and the Web UI to authenticate users that access the system. This was initially planned in for Semester 1 however slipped into the start of Semester 2 after spending a large amount of time focused on the API and initial designs of the system.

Objective 4: Back End Creation - Achieved

The API has become the core component of the application. Most of the work that the User interface does has been routed through this API to maintain separation between the logic and presentation layer, following an MVC based pattern.

This was initially built using Test Driven Development for the Data Access Layer component however was not continued in that manner resulting in manual testing. In retrospect this shouldn't have been the case and has led to one of the main delays in the development of the application and testing the different components via multiple tools and raw requests via telnet.

To improve this, a testing framework should have been implemented at the start and continued past the DAL (which could have used a mocking framework for unit testing) rather than tests that were not well written relying on the state of a blank database.

As stated previously, testing was one of the largest time sinks in the project, and a large number of issues was encountered when initially developing the API, which led to the creation of an additional test application that called into the API. This is another area which slipped during the development of the project, as once the initial Web UI was implemented, development on this small test application stopped, and testing shifted to using the Web UI. This again links back to why using TDD would have been more beneficial to keeping the project on schedule, and could have been integrated with the Continuous Integration system that was used during development (GitLab + TeamCity).

Along with this, issues were encountered when building the account management system. During the implementation of Active Directory group checking, an issue was found whereby the `UserPrincipal.IsMemberOf(group)` method was returning invalid data, which was failing authentication when testing different accounts at different authentication levels to the main account. This required manual checking of the available Active Directory groups and comparing them against the list of groups that the user is a member of.

Despite this, the objective has been completed. The core components of the API were implemented in Semester 1 and any additional functions that were required by the UI were implemented in place.

Objective 5: Front End Creation – Partially Achieved

In building the user interface, initially there was a debate between if the UI should access the database directly through the DAL or make calls at runtime to the API for access to the data. After weighing each point, it was decided that it would call back to the API for its data access features. This allows for a more loosely coupled design, and can further improve performance on larger data by separating the user interface from the API and running on more powerful hardware. This can also lead to increased security, whereby if the system was split, the API could be configured to not be externally accessible for direct calls, and only accessible from the UI. In the implementation some of the tables are populated via direct access and some are populated at runtime via ajax calls to the API. For a point of further development, all of the tables should be populated via ajax call and therefore completely decoupling the Web UI from the rest of the system.

During development of some of the jQuery based user interactions, issues were encountered with combining both jQuery draggable and jQuery sortable when assigning backlog items to sprints as well as changing priorities. Initially the design was to have the ability to drag a backlog item onto an iteration on the left of the page, however because of this issue, a dialog box was introduced instead to facilitate the adding of tasks to iterations.

Issues were also encountered when designing the SCRUM board feature, which as of writing is only partially implemented. Currently, the initial layout is there with the core jQuery components present, however this will need integrating into the main site as it currently resides on a test page. The issues that were encountered was that a div cannot have a preset size if there is no content, which lead to issues with creating a droppable area. This issue has been avoided by displaying a box around the droppable area when a task is moved over it. Another issue that was encountered was that the task div would randomly jump over the page if it was not in the correct place to be dropped. This was handled by the use of a check in the drop function to ensure that it was placed in the correct zone, otherwise it would reset the div to its original position.

Objective 6: Reporting – Done

For reporting, research was carried out into multiple methods of displaying charts within a web page. Chart.js was chosen due to its open source nature and ease to implement within the solution while still being able to dynamically generate the charts on the client side and allow for some interaction with the charts rather than a static image generated server side and then sent to the client.

Secondary Objectives

During the course of the project, the only secondary objective that was completed was relating to the overall performance of the system which was a core personal aim of this project.

This was achieved by designing the system in such a way that there is a minimal amount of data that is required when making any kind of update transaction to the server. As previously stated during section 4.3, the test environment was set up to represent a low powered server configuration which made this objective easier to test.

The result of which is that the overall system + database server (2 servers) has a memory footprint of 3GB of ram (including running 2 instances of Windows Server 2012 R2 for both IIS and SQL Server).

This is in contrast to previous experiences with Team Foundation Server, whereby the configuration used at the start of the project (8gb SQL, 4GB TFS Server) took quite a long time (20-30s per transaction) to update elements of the project.

The following table lists the initial features based on the created User Stories and if they have been implemented at the time of this report.

<u>Feature</u>	<u>User Story for Feature</u>	<u>Support in the Tool</u>
Creating Projects for Teams	1	Yes
Creating Product Backlog Items	2	Yes
Set Priority of Backlog Items	3	Yes
Commit Backlog Items to Iterations	4	Yes
Create Tasks	5	Yes
Create Tasks from SCRUM Board	5	Partially Implemented
Move Tasks Across SCRUM Board to change State	6	Partially Implement, Not Currently Functional
View Reports	7	Yes
Interact Via 3 rd Party Applications	8	Yes, Support Documentation Created explaining how the API functions (See Appendix E).

5.2 Project Planning

SCRUM was used to plan out the project. This was chosen due to being able to plan out the project around other University work therefore being able to respond to increasing workloads effectively. Initially the sprints were quite short in length (2-4 days) to account for time that was not spent on campus. Therefore, initially only one task a week was aimed for, resulting initially in a low average velocity and Slower burn down. This is shown in figure 27, generated from the reporting section of the project.

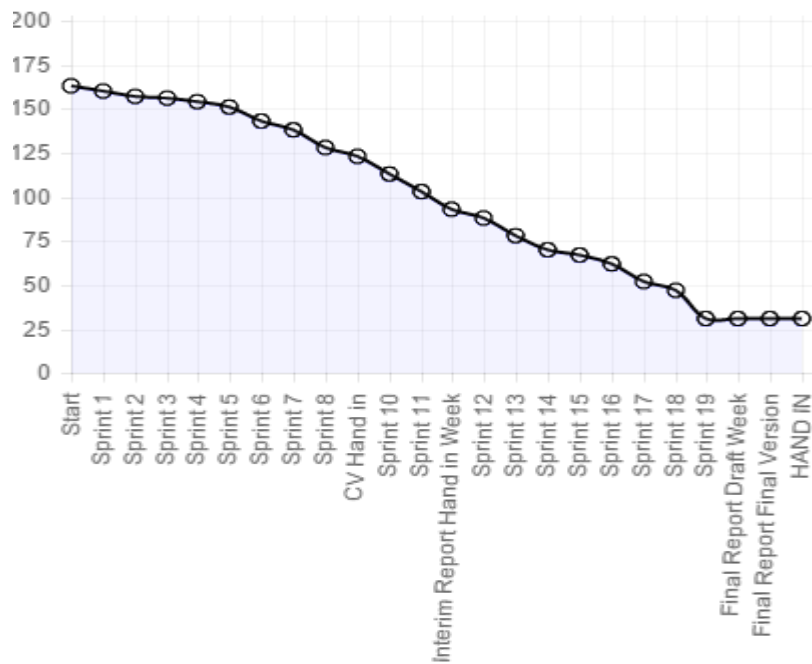


Figure 27. Project Burndown

Figure 27 does however show a steady pace mostly throughout the project up to Sprint 19 (Easter holidays) whereby more time was available to work on the project. This is also shown in the velocity chart in figure 28.

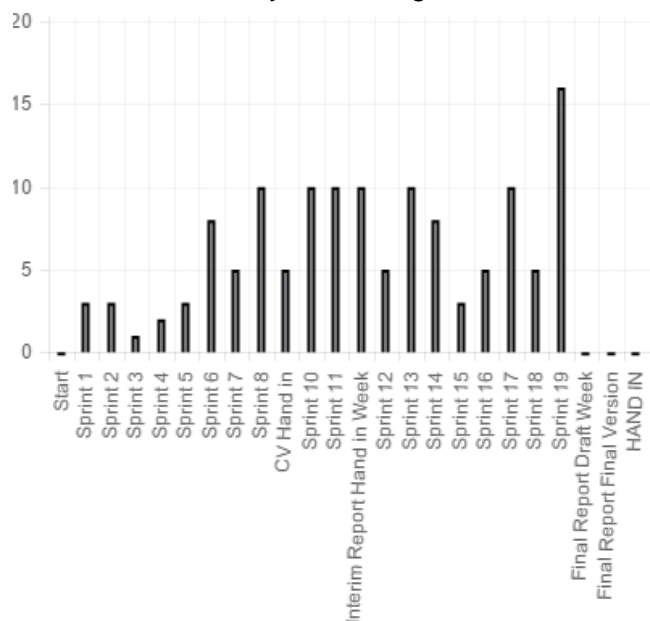


Figure 28. Project Velocity

Figure 28 also shows that the average velocity is approximately 6 (6.6). This is a low velocity score, for example in a team of 6 for a 2-week sprint there would be an average velocity of about 40. However, since the sprints were quite short in length as mentioned before (2-4 days), and only one person working on the project, this would be consistent with an average velocity if working in a team of 6.

In retrospect, some of the effort scores assigned to the initial task list were not representative of the actual work required to complete the item in question, and needed to be broken down across multiple sprints. This could have been avoided by spending more time to break down each task rather than leaving broad tasks like “Build User Interface” on the project backlog.

The Initial time plan and task list (See Appendix B and C) remained mostly static with very few changes, with the only major change being that the account management system was pushed into Semester 2, this was due to having to fit other coursework and exam revision around the project, however has not caused any significant delays in the overall project, as initially time was set aside in case of any overruns.

5.3 Further Work

Initially the interactive SCRUM board on the website home page will need to be finished. This will require migrating the layout from the test page to the site home page and modifying the existing scripts to allow the user to select multiple projects. Along with this, the board will also need finishing off by adding ajax calls to the API to update the state of the data. This should take a sprint to finish implementing.

Moving into the future, a core area of improvement would be in the User Interface on the website as some initially planned elements were removed from the pipeline due to time constraints and level of knowledge with libraries used. One area of improvement here would be on the project details page, whereby when adding a backlog item to an iteration currently requires opening a dialog, dragging the appropriate item to the name of the iteration on the left of the page would make interaction faster and more streamlined.

Another point of improvement will be to adjust how the reporting method in the API packages up the data exclusively in a format to use with Chart.js, Whereas for the UI this is somewhat useful, in the scope of the application it would be better to send the raw data separately then construct the graphs client side (for more information on the current system, please refer to the previous section and the technical documentation).

Following on from this and touched upon briefly in section 5.1, a couple of sprints should be set aside to work on implementing unit testing and automatic integration testing across the project, this should include using a mocking framework on the data access layer along with User Interface testing. This could then be fed into a continuous integration package in an enterprise setting to streamline deployment.

5.3.1: Known Issues

Currently the biggest of the known issues is present on the project details page, whereby the iteration list despite being a static list supplied from the model at the page load, will shuffle around when a product backlog item is added to an iteration. In testing ordering does not make a difference here. An example of this is shown in figure 29.



Figure 29. Ordering Issue

This list should be in the same order as figures 27 and 28, yet when adding a backlog item to an iteration using the dialog, which shows them in the order that they were added into the system, the list will randomly change and the sprint that the item was added into will be moved. This list is populated at page load with a for each loop which gets all iterations associated from the project selected. In testing I have tried modifying the loop to order by the start date of the iterations, both ascending and descending without success.

Another issue that appeared during testing is that sometimes when the dialog box is used and the Add/Update button is clicked, sometimes the system will not record the change and no error will be logged as to why, however if the page is refreshed and the operation is tried again then this time it will work without any issues.

6 Conclusion

Looking back at the initial aim of the project, To Create an Open Source Web Based SCRUM Management Tool using ASP.NET with a Web API based data access layer to allow for third party applications to interact with the site, the project deliverables have met that aim, however there is still room for improvement.

A core personal aim to this project was to have the end system perform well without a large system overhead, this was stated as a secondary objective however in testing in my test environment this goal has been met, with a total overhead of less than 3gb of ram required to host both the database and website, where in contrast for an average performing team foundation server system with everything running on one server recommending 8 GB of ram (Microsoft, n.d.). TFS can run with less ram, however performance will suffer.

Working on this project has allowed me to further enhance my experience with enterprise grade system management components (Active Directory, SQL Server) as well as learning more about JavaScript and jQuery and how best to implement different components of the system to balance the work between client and server. Alongside varied source control systems, with the University supplied SVN service, my own internal Team Foundation Server (Used initially when planning the project and initial source control repository) then moving onto using GitLab with build agents provided by TeamCity for the end of the project, with each system having its own quirks and benefits from both a performance and system overhead perspective.

In constructing the project, quite a few challenges were posed in the development, some down to lack of initial knowledge of systems like jQuery and others down to having to work with intricate systems like Active Directory when integrating authentication into both the ASP.NET UI and the Web API components of the project.

This project has included multiple technical challenges, primarily focused around integration with Active Directory and separately authenticating the Web API from the User Interface. This required multiple stages of testing to see how the browser would handle making subsequent calls to different components of the application with regards to passing credentials. Another technical achievement is the use of a dual authorization system which combines the Active Directory groups with the defined roles within the system, and allows non-domain administrators to explicitly define roles to users within the scope of the system. This was achieved by multiple levels of checking and custom Authorization filters to provide this checking. A personal challenge in this project was its heavy use of the jQuery library in the user interface. As stated in the previous paragraph, there was a distinct lack of initial knowledge of how jQuery functions, alongside basic JavaScript knowledge. Therefore, throughout the duration of this project, heavy investment was spent in both time and also some investment in purchasing training packages in this area to increase personal knowledge in the fields.

Appendix A: Glossary of SCRUM Terms

- **Product Owner:** “The Product Owner is responsible for maximizing the value of the product and the work of the team” (Schwaber & Sutherland, 2013). The Product Owner is also responsible for priorities of tasks.
- **Development Team:** “A Cross-Functional, Self-Organizing/Self-Managed Team” (James, n.d.), which carry out the tasks of the Sprint
- **SCRUM Master:** “The SCRUM Master is responsible for ensuring SCRUM is understood and enacted” (Schwaber & Sutherland, 2013). Will be one of the points of contact if there is a clash of interests between the team and Product Owner if there are issues with a task that requires escalation.
- **Product Backlog:** A list of tasks for the project. The tasks can also be referred to as Product Backlog Items. The Product Backlog where possible should be visible and editable to all Stakeholders and will be re-prioritized by the product owner (James, n.d.)
- **Product Backlog Item:** Also referred to as a task, this is often written as a user story in the form of “As a (user of the system) I want to be able to do (x feature) for (y output)”. It will also have an effort rating which can be represented as a number of hours/days, which is decided during the Backlog Refinement Meeting by the Development Team
- **Sprint:** (Also known as iteration) is a fixed length block of time less than a month e.g. 2 weeks, which will have a set of backlog items committed to be “done” in the sprint by the team, and should result in an iteration of the product that is releasable
- **Sprint Backlog:** A subset of the Product Backlog that is decided and committed to being finished within a Sprint during the Sprint Planning Meeting.
- **“Done”:** In the case of done, the organization or team will have their own definition as to what done means, this could include for example:
 - Fix bug
 - Document
 - Test
 - Commit Changes to Source Control
 - Managed Ticket

Appendix B: Initial Task List

Semester 1

#	Task Name	Description	Duration (days)
1	Initial Report	Constructing the Initial Report Deliverable	2
2	Research Existing Solutions	Investigate existing Scrum tools to build more of an understanding as to what is required	2
3	Database Design	Create initial un-normalized database for application	1
4	Database Normalizing	Proceed to normalize database from task 3	2
5	Authentication methods	Investigate different Authentication Methods	2
6	Authentication System	Build Basic Account Manager (Completed During Semester 2)	4
7	Back End Web API	Create Data Access Layer (CRUD) for the Web API	1
8	Back end Web API	Create Basic API for accessing the database	2
9	Back End Web API	Build Web API	4
10	CV Hand in	Update CV and create Cover Letter	4
11	Back End Web API	Build Testing Tool and Test API	4

Key:

Item Completed.

Item In Progress.

Item Not Started.

Semester 2

#	Task Name	Description	Duration (days)
12	Interim Report	Constructing the Interim Report Deliverable	4
13	Front End Design	Design the Layout for the Front End Website to access the application	2
14	Front End Research	Research client side scripting for Drag-and-Drop features of the Scrum Board	4
15	Front End Creation	Build the Front end Website Layout and features	4
16	Drag-And-Drop	Implement Drag-and-Drop Features to front end	3
17	Reporting Research	Researching Graphics Libraries for the website	2
18	Reporting	Implement Burndown and Velocity Charts	2
19	Testing	Integration Testing and Program Documentation	3
20	Preparation for Final Report	Preparing for final report, collecting code samples, references and any other requirements	15
21	Final Report	Creating Final Report	10

Appendix C: Initial Time Plan

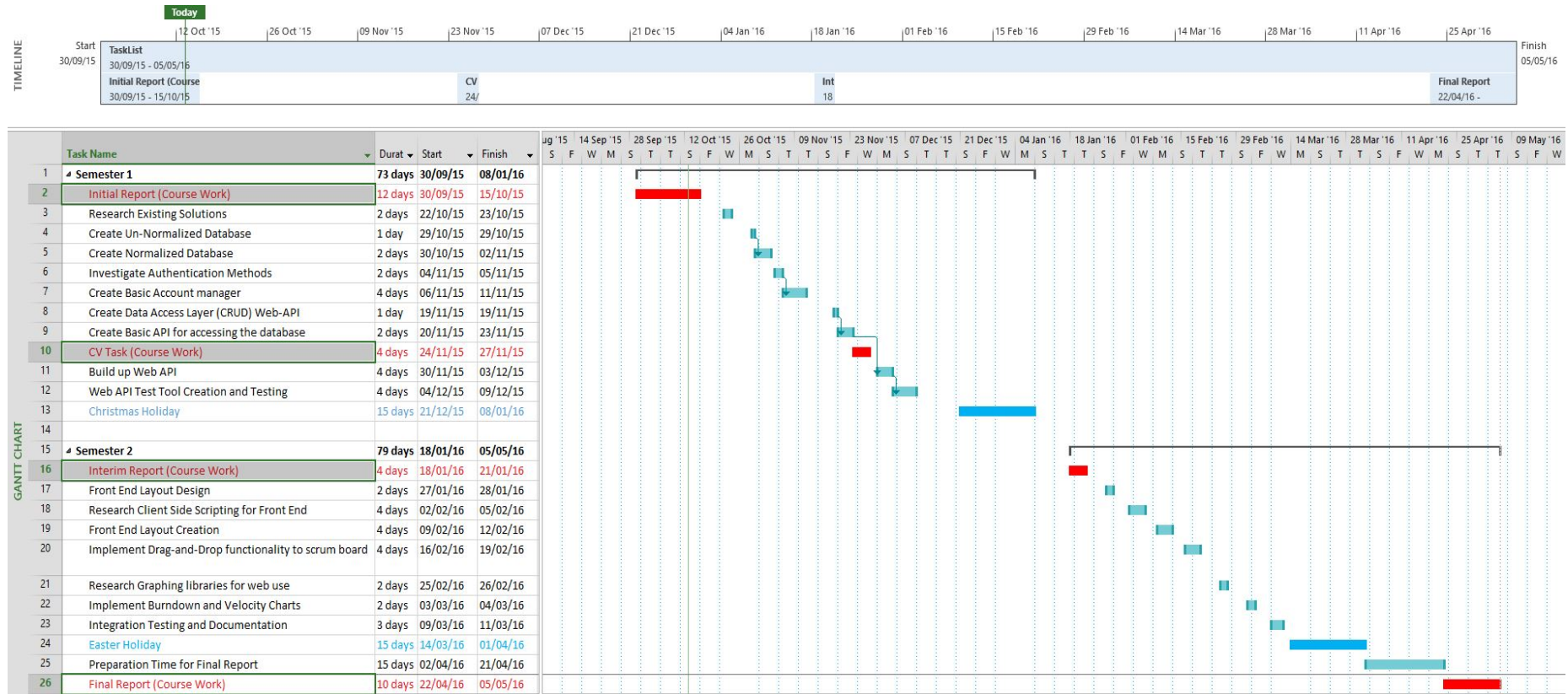


Figure 30. Initial Time Plan Gantt Chart

Appendix D: User Documentation

The Contents page has been removed from the documentation due to it conflicting with the existing Table of contents of this report



OS-Scrum

Installation and User Guide Version 1

Christopher D. Haworth
4-18-2016

1. Introduction

In this document I will explain how to setup and use the OS-Scrum system.

This document should also be accompanied with a Technical Document which will go into more technical details as to how the system works should the user wish to change some of the features.

This Application is released as an open source platform and licensed under the GNU General Public License Version 2. No warranty will be given for the use of the software.

2. Licence

2.1 GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.

51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

2.1.1 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

2.1.2 TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program

(independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License.

(Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control

compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License.

However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

3. System Requirements

To run this system an existing Active Directory domain is required as this application makes use of this for its user authentication system.

The Web Server will need to be running a recent version of IIS (Tested on version 7) with ASP.NET 4.5 installed. Along with this there will need to be a SQL Server available to host the database. This can be on the same server, but will most likely be present somewhere else on the network. The application has been tested on Microsoft SQL Server 2014 however should work on 2008 R2.

Hardware Requirements:

- IIS Server
 - OS: Windows Server 2012 R2
 - Will require IIS Installed and the server will need to be part of the domain
 - CPU: Minimum of a dual core CPU
 - Tested on an Intel Core i3-4010U virtual machine at 1.7Ghz
 - RAM: 1GB
 - Should you need to run SQL on the same server this will need to be increased
 - Storage: 18MB
 - Again this will need to be more if you plan on installing SQL on the same machine
 - Network Connection
- SQL Server
 - Follow the recommended specifications for a production/or testing server from Microsoft available here: <https://msdn.microsoft.com/en-us/library/ms143506%28v=sql.110%29.aspx?f=255&MSPPError=-2147217396>
 - In testing a SQL server running on the same Hyper-V host as the IIS Server only required 2GB of RAM and a dual core CPU to function correctly.

4. Environment Preparation

4.1 Domain

The Application has a series of Active Directory groups that it will use for checking to see if a user is authenticated and authorized to access the relevant resources of the system. There are 7 groups that will be required to be created, however the naming is up to you, these group names will need to be added into the Web.config file, this will be covered in a different section.

The seven groups are as follows:

1. Administrator
2. Scrum Master
3. Team Leader
4. Team Member
5. Stake Holder
6. View Only
7. Unauthorized

An administrator will have full control of the system and be able to grant users explicit permissions within the scope of the application, more information on this will be covered later in this document.

A Scrum Master can do everything an Administrator can do apart from user management, this would be a normal role for standard management features i.e. Managing what teams are on what projects etc.

A Team Leader has the ability to add new sprints/edit existing sprints within projects that their team is assigned to.

A team member and stake holder have similar permissions, apart from the fact that a team member can edit tasks, add tasks to sprints and remove tasks, whereas a stake holder cannot.

A stakeholder account would be given to the client to be able to access the system and view the project that they are interacting with to aid in collaboration.

A View Only Account is the lowest level of permission in the system and is used to view the list of projects. A view only account can be added to a team which would allow them to see the backlog however it does not allow for changes to the underlying data (apart from to update the user that logs in if required via the API).

Finally, any account in the Unauthorized group is the equivalent of blocking the user from accessing the system.

Any requests that do not match the required level of access will be logged with the user that initiated the request and the requested resource to allow for auditing.

4.2 IIS Server

The IIS server will need to be set up with ASP.NET 4.5 installed along with its dependencies (.NET Framework 4.5). Along with this, the appropriate firewall ports will need to be opened for the application. The IIS Server will also need to be on the domain to allow for the use of domain based authentication.

4.3 SQL Server

The SQL Server can be collocated on the same machine as the IIS server, however it is recommended to have a separate system to run this within a production environment, therefore ensure that this machine's database server is accessible from the IIS server, and the IIS Server has permissions to access the database. (For setting up the database please refer to the installation section of this document).

5. Installation and Configuration

Depending on how you acquire this software there will be slightly different steps required to install the system.

If you download the source code directly from GitHub (Recommended) then you will need a development machine running Visual Studio 2013 minimum to publish the package files, then follow the steps outlined here starting from 5.1

If you have downloaded a prebuilt package (Also available on GitHub), after extracting the package, follow the steps below.

5.1 SQL Server

Within the extracted zip, in the resources folder, there is a script to set up the schema for the database on the SQL Server. This will need to be ran on an empty database to prepare the system first.

Also at this point ensure that the account used for accessing the SQL Server from the IIS instance has permissions to read and write to this database.

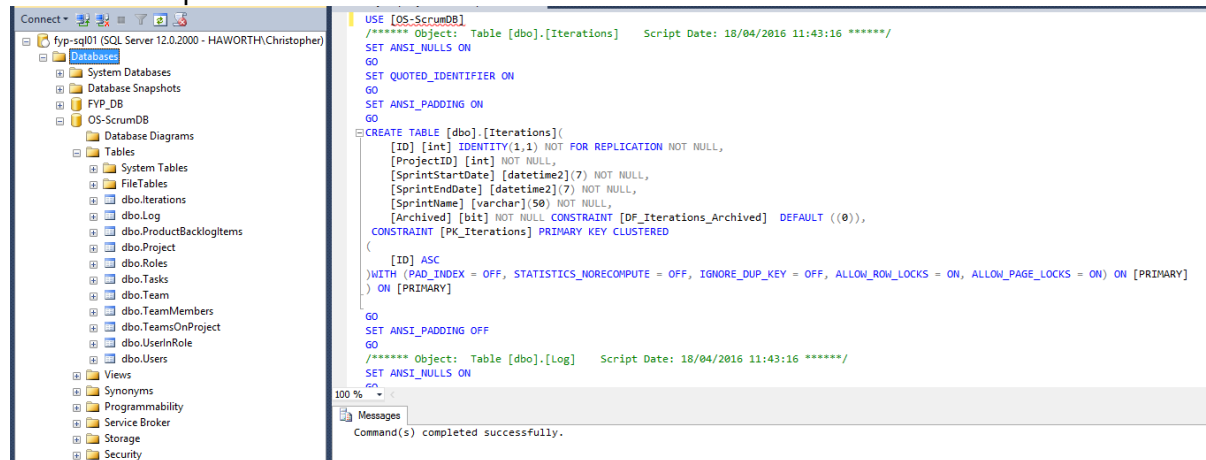


Figure 31. Running the DB Script on the Blank database to setup the schema

Once the script has finished, you should see something similar to the following, (there may be other databases present than this screenshot however).

At this point you are now ready to proceed to setting up the IIS Server, which will be explained in the following section.

5.2 IIS Server

To Setup the IIS Server, firstly copy the extracted zip file to a folder on the IIS server, e.g. C:\inetpub\os-scrum. Then open IIS Manager and add a new site

The screenshot shows the 'Add Website' dialog box in IIS Manager. The dialog is titled 'Add Website' and has a blue header bar. It contains the following fields and controls:

- Site name:** A text box containing 'OS-Scrum'.
- Application pool:** A dropdown menu showing 'OS-Scrum' and a 'Select...' button.
- Content Directory:**
 - Physical path:** A text box containing 'C:\inetpub\os-scrum' and a browse button (...).
 - Pass-through authentication:** Two buttons: 'Connect as...' and 'Test Settings...'.
- Binding:**
 - Type:** A dropdown menu set to 'http'.
 - IP address:** A dropdown menu set to 'All Unassigned'.
 - Port:** A text box containing '80'.
 - Host name:** An empty text box with the example 'www.contoso.com or marketing.contoso.com' below it.
- Start Website immediately:** A checked checkbox.
- Buttons:** 'OK' and 'Cancel' buttons at the bottom right.

Figure 32. Adding the site in IIS Manager

Figure 2 shows a default configuration for the Website assuming that port 80 is free and the path to the files is correct. It is recommended to uncheck the start website immediately option, so that the system can be configured.

Following on from this, the site will need to be configured to use windows authentication. To do this, go into the Authentication section and configure it to look like figure 3:



Authentication

Group by: No Grouping		
Name	Status	Response Type
Anonymous Authentication	Enabled	
ASP.NET Impersonation	Disabled	
Basic Authentication	Disabled	HTTP 401 Challenge
Digest Authentication	Disabled	HTTP 401 Challenge
Forms Authentication	Disabled	HTTP 302 Login/Redirect
Windows Authentication	Enabled	HTTP 401 Challenge

Figure 33. Authentication Settings

You may have different options available depending on what features of IIS you have installed, however so long as you have both anonymous authentication and windows authentication, this is not an issue. Now we can move onto configuration of the website.

5.3 Configuration

There are two files that will need updating to allow for the application to work, these are the Web.config file, located in the root of the website, and the Helper.js file located in Resources/Scripts.

5.3.1 Web.Config

In the Web.Config File there are 9 items that will require changing, 8 of them are located in the AppSettings.

1. ADDomainName: This will need to be set to the name of the Active Directory Domain that the User Groups are configured on
2. AdminGroup: Name of the Administrator Group
3. ScrumMasterGroup: Name of the Scrum Master Group
4. TeamLeaderGroup: Name of the Team Leader Group
5. TeamMemberGroup: Name of the Team Member Group
6. StakeHolderGroup: Name of the Stakeholder Group
7. ViewOnlyGroup: Name of the View Only Group
8. UnauthorizedGroup: Name of the Unauthorized Users Group

The group names will need to be inputted the same as they are in Active Directory for example:

FYP_Admin_Group	Security Group...	Test Group for Final Year Project	<add key="AdminGroup" value="FYP_Admin_Group" />
FYP_ScrumMaster_Group	Security Group...	Test Group for Final Year Project	<add key="ScrumMasterGroup" value="FYP_ScrumMaster_Group" />
FYP_StakeHolder_Group	Security Group...	Test Group for Final Year Project	<add key="TeamLeaderGroup" value="FYP_TeamLeader_Group" />
FYP_TeamLeader_Group	Security Group...	Test Group for Final Year Project	<add key="TeamMemberGroup" value="FYP_TeamMember_Group" />
FYP_TeamMember_Group	Security Group...	Test Group for Final Year Project	<add key="StakeHolderGroup" value="FYP_StakeHolder_Group" />
FYP_UnauthorizedGroup	Security Group...	Test Group for Final Year Project	<add key="ViewOnlyGroup" value="FYP_ViewOnlyGroup" />
FYP_ViewOnly_Group	Security Group...	Test Group for Final Year Project	<add key="UnauthorizedGroup" value="FYP_Unauthorized" />

Figure 34. Example Group Configuration

Following on from this, the next section that will need to be edited will be the database connection string at the bottom of the document. This is best done in the config file rather than IIS.

There are two or three elements that will need changing depending on if you use a windows user account to access the SQL Server or a SQL User.

1. Data source: location of the database
2. Initial Catalog: Name of the Database (Aka OS-ScrumDB)

3. If you are using Windows Authentication, then the Integrated Security=True is left in, otherwise if you are using SQL Authentication, this will need to be replaced with:
integrated security=false: user id={Username here};password={password here};

Once this is done, the final piece of configuration can then be done.

5.3.2 Helper.js

Within Helper.js there is a path to the API, e.g. <http://localhost:4874/api>. This will need to be updated to point to the address of the website set up in 5.2 followed by “/api”. This is used for contacting the Web API from within the User Interface.

Once this file has been updated, Start the website in IIS Manager then setup is complete.

6. User Guide

When you first navigate to the site as a valid user, you will be presented with the following screen:

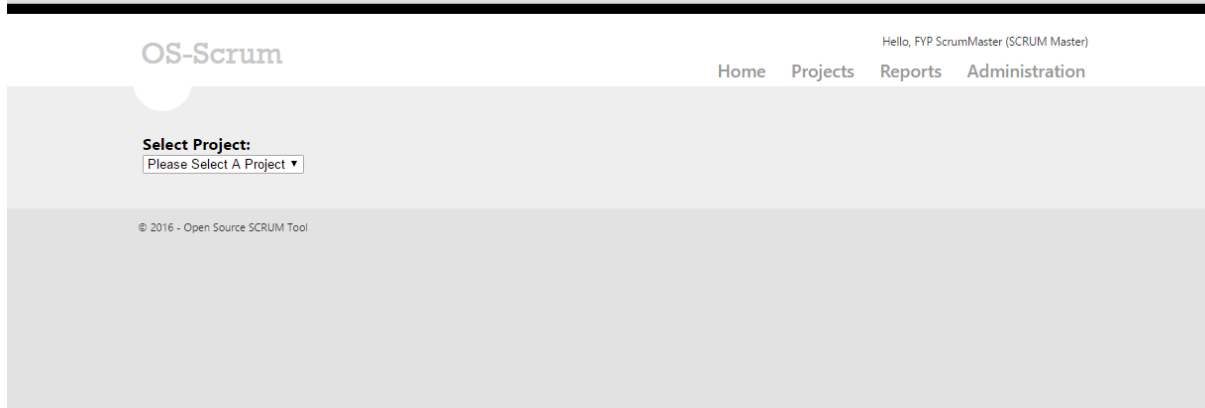


Figure 35. Home Page

It is recommended that initially you log in as an Administrator to be able to set up all of the required teams.

6.1 Administration

The first area you will want to visit on a clean installation would be the administration section of the website, please note however if you do not have the required permissions to access this page, the link will not display as shown below:

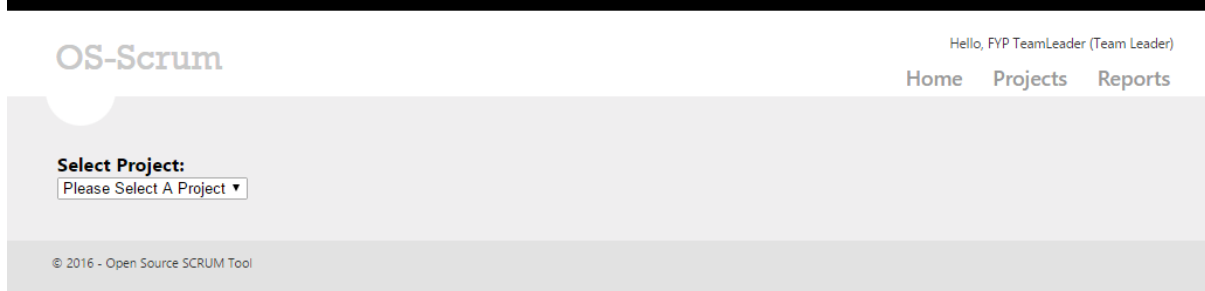


Figure 36. Logging in with a user that does not have permissions to view the Administration section of the site

When you enter the administration section of the website you are presented with a list of available options, as shown on the next page:

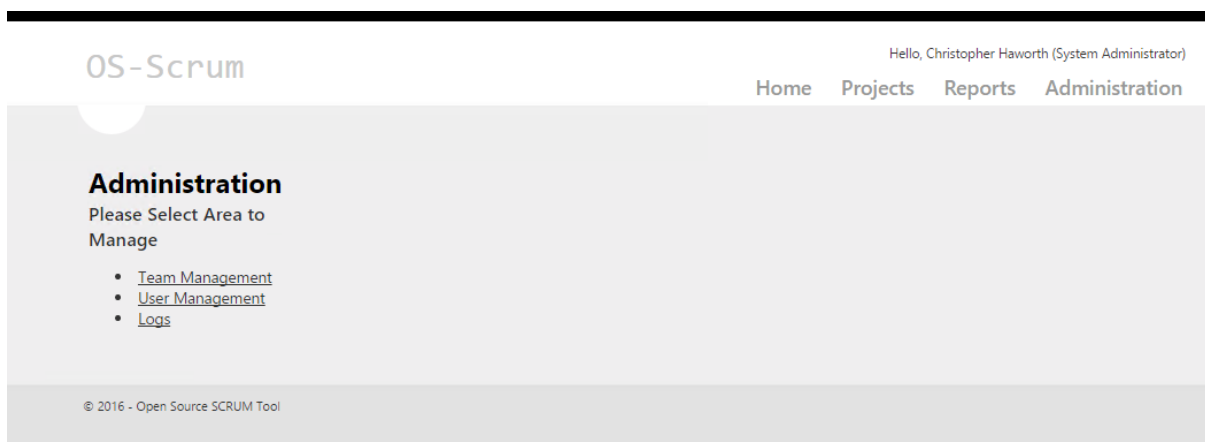


Figure 37. Administration Section

From here you can manage teams, view the current users and manage their explicit roles, and view the system logs.

6.1.1 Team Management

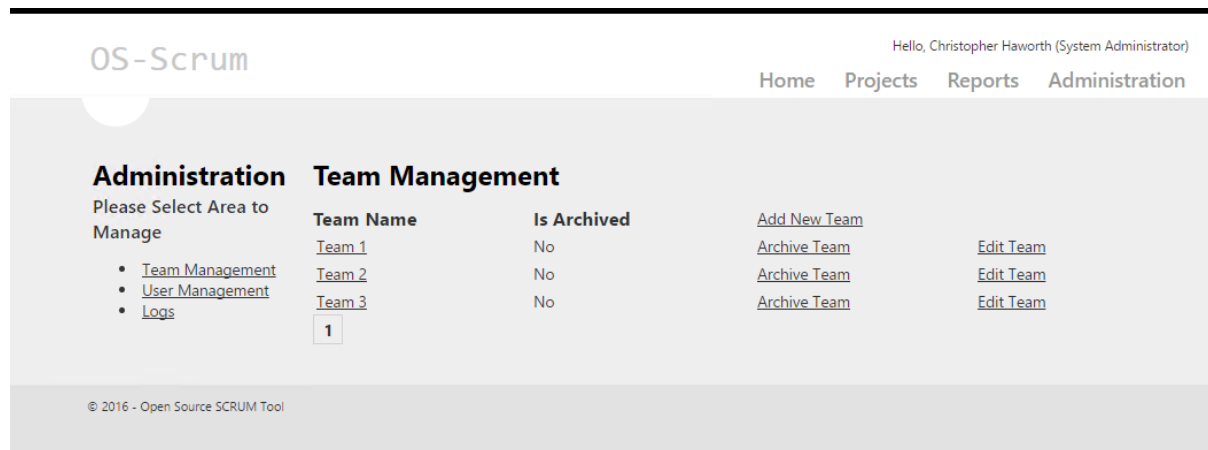


Figure 38. Team Management

When you enter the team management section, a paginated table of all the teams registered in the system is displayed. From here you can add new teams, archive/restore existing teams, and edit the team information. This is achieved through the use of jQuery dialogs. If you then proceed to click on a team name, information as to what projects the team is assigned to is then displayed, or a message displaying "Team not assigned to any Projects" will be displayed.

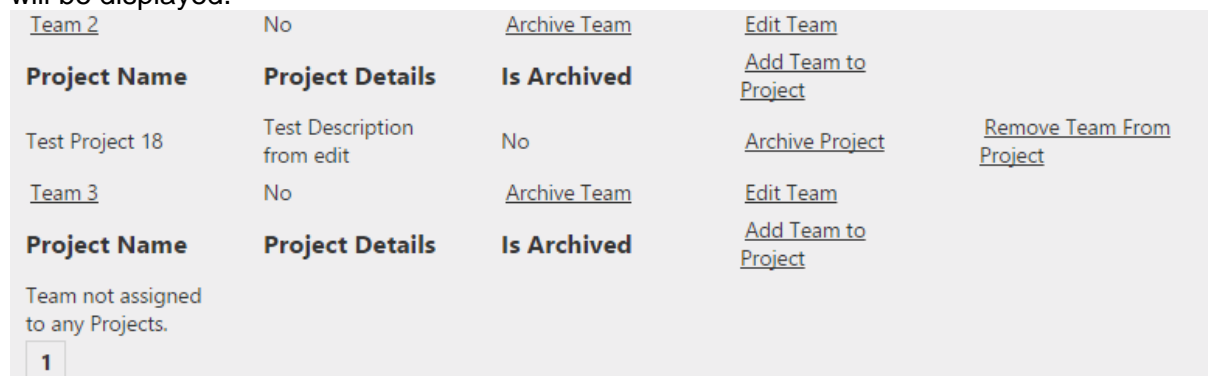


Figure 39. Showing Teams that are assigned and not assigned to projects.

At this point it is possible to add and remove teams from projects as well as archive/restore projects. Clicking on the team name again will hide the additional information.

6.1.2 User Management

Moving onto User Management, when you first enter the system, only your user information will be populated. It is not currently possible to add users from this page.

OS-Scrum

Hello, Christopher Haworth (System Administrator)

Home Projects Reports Administration

Administration

Please Select Area to Manage

- [Team Management](#)
- [User Management](#)
- [Logs](#)

User Management

First Name	Last Name	AD User Name	Email Address	
Christopher	Haworth	HAWORTH\Christopher	[Masked]	Edit Block User
FYP	ScrumMaster	HAWORTH\FYP_ScrumMaster	No Email Address	Edit Block User
FYP	TeamLeader	HAWORTH\Fyp_TeamLeader	No Email Address	Edit Block User

1

© 2016 - Open Source SCRUM Tool

Figure 40. User Management (Email Address Masked for Privacy)

From here you can edit the user information that has been populated by Active Directory, However the only updateable fields are the First Name, Last Name and Email Address data shown on the following page in the Edit User Information dialog box.

Edit User Information

Edit First Name
FYP

Edit Last Name
ScrumMaster

Edit Email Address

Update User Info Cancel

Figure 41. Edit User Information Dialog

From here you can also access both the User's associated teams and roles. To access the Teams, click on the User's First Name (A tooltip will appear when you hover over the link to inform you), and likewise to view the roles, click on the User Name.

inform you), and likewise to view the roles, click on the User Name.

FYP	ScrumMaster	HAWORTH\FYP_ScrumMaster	No Email Address	Edit	Block User
Role Name	Role Level	Mapped AD Group	Controlled by AD	Add Role	
SCRUM Master	Scrum Master	FYP_ScrumMaster_Group	Yes		
Team Name	Add User to Team				
User is not a member of any teams					

Figure 42. Team and Role Management from user control

The above figure shows both the role management and team management components from the user management section (to hide these, click on the respective links again). From here you can add or remove users from teams and define/remove explicitly granted roles for the selected user.

6.1.3 Log Viewer

The final section of the administration area of the system is the log viewer. This page allows the administrator to search through the log information that is stored in the database along with sorting by Event data or log level.

OS-Scrum Hello, Christopher Haworth (System Administrator)

Home Projects Reports Administration

Administration Log Viewer

Please Select Area to Manage

- Team Management
- User Management
- Logs

Search Log Messages:

Log Level	Log Message	User for Action	Event Date
Error	Invalid Access to the following url from website /Admin	HAWORTH\Christopher	3/22/2016 10:52:42 PM
Error	Error Unblocking User: System.InvalidOperationException: Collection was modified; enumeration operation may not execute. at System.Collections.Generic.HashSet`1.Enumerator.MoveNext() at OpenSourceScrumTool.DAL.DataAccessLayer.UnblockUser(Int32 id) in d:\08341 SVN\Ongoing Development\Software\SCRUMTool\OpenSourceScrumTool\OpenSourceScrumTool\DAL\DataAccessLayer.cs:line 380 at OpenSourceScrumTool.api.UserController.UnblockUser(Int32 id) in d:\08341 SVN\Ongoing Development\Software\SCRUMTool\OpenSourceScrumTool\OpenSourceScrumTool\api\UserController.cs:line 258	HAWORTH\Christopher	3/23/2016 10:45:25 AM
Error	Invalid Access to the following url from website /Admin	HAWORTH\Christopher	3/23/2016 8:33:38 PM
Error	No Iteration for project with ID: 2	HAWORTH\Christopher	4/14/2016 6:37:30 PM
Error	No Iteration for project with ID: 2	HAWORTH\Christopher	4/14/2016 6:37:47 PM

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

Figure 43. Log Viewer

This is the main way to view the log files for the application, however the information is also available from the API for 3rd party applications. (More information available in the Technical Documentation for this).

6.2 Project Management

6.2.1 Project List

When you click on the projects link, you are presented with a list of projects that are currently in the system as shown below:

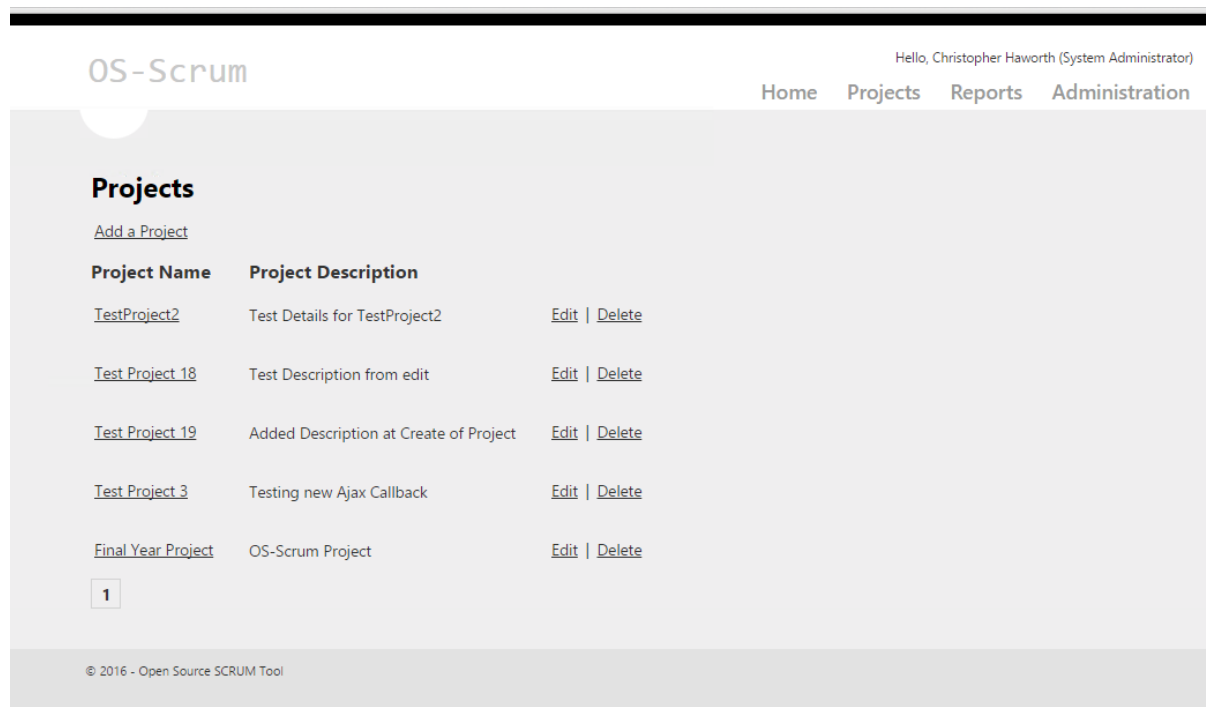


Figure 44. Project Index

If you are not in a Scrum Master or higher role, the Add Project, Edit and Delete items will not be visible to you. As an Administrator or Scrum Master, you can access any of the projects regardless of if you are a member of a team that is working on the project. However, if you are a team leader or less, you have to be a member of a team that is working on the project to get any further in the system. If you try to access a project that you do not have permission to access, an error message will be displayed (shown below)

You are not a team member working on this project or a Scrum Master or higher, therefore cannot access the requested project. If you believe this to be in error, please contact your administrator.

Figure 45. Error Message if accessing project that you are not a member of

As with the administration page, the add/edit links open dialogs to input the data. Clicking on a project name, so long as you have the appropriate access permissions will take you to the Project Details Page

6.2.2 Project Details

If you have the appropriate rights to access the project and have clicked on the project on the above screen, you will be presented with the Project Details page, shown below:

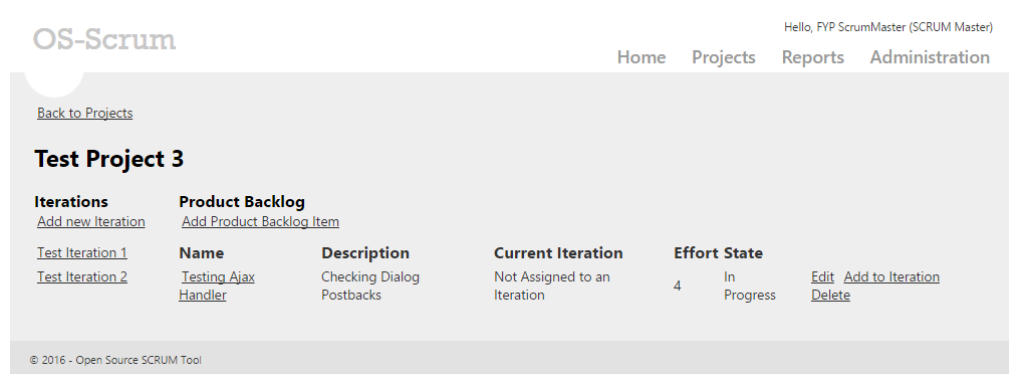


Figure 46. Project Details Page

From here you can manage iterations for the project as well as the Project Backlog. If you are a team leader or higher, the option to Add New Iterations will be displayed.

Currently this is where most time will be spent in the application. Clicking on the name of an iteration will open up additional details about it, and will also give a team leader or higher the option to edit or remove the iteration (Shown below).

[Test Iteration 1](#)

Start Date
3/20/2016

End Date
3/26/2016

[Edit](#)

[Delete](#)

Figure 47. Iteration Information

Clicking again on the name will hide this information, same as the administration page. Anyone who is a stakeholder or higher, has the option to add a product backlog item at this point. Clicking on the link will open up a new dialog to add the Backlog Item (Shown Below)

Add Product Backlog Item ✕

Name

Description

Effort

Backlog Item State
Not Started ▾

Add Product Backlog Item

Figure 48. Add Backlog Item Dialog

When the information is filled in, click on the Add Product Backlog Item button. A small loading screen will display, then the page will refresh when it has finished.

Test Item	testing	Not Assigned to an Iteration	2	Not Started	Edit Add to Iteration Delete
---------------------------	---------	------------------------------	---	-------------	--

Figure 49. New Backlog Item

To set the priority of a backlog item, simply drag its position in the list, for example moving test item above testing ajax handler

Test Project 3						
Iterations		Product Backlog				
Add new Iteration		Add Product Backlog Item				
Test Iteration 1	Test Item	testing	Not Assigned to an Iteration	2	Not Started	Edit Add to Iteration Delete
Test Iteration 2	Testing Ajax Handler	Checking Dialog Postbacks	Not Assigned to an Iteration	4	In Progress	Edit Add to Iteration Delete

Figure 50. Changing Priorities

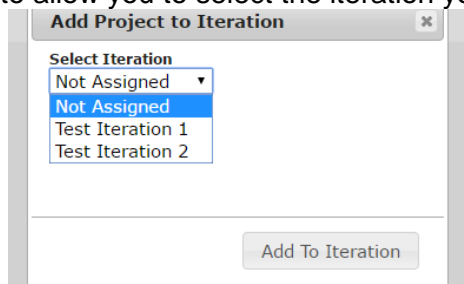
When you drop the item, a short loading screen will appear again, and the page will refresh with the new priority, shown below:

Product Backlog					
Add Product Backlog Item					
Name	Description	Current Iteration	Effort	State	
Test Item	testing	Not Assigned to an Iteration	2	Not Started	Edit Add to Iteration Delete
Testing Ajax Handler	Checking Dialog Postbacks	Not Assigned to an Iteration	4	In Progress	Edit Add to Iteration Delete

Figure 51. Updated Priorities

Modifying the backlog items requires a minimum role of a team member, if you are not a team member or higher, the three links on the right will not display.

To Add a backlog item to an iteration, click the add to iteration link. This will open up a dialog to allow you to select the iteration you want to add the item to:



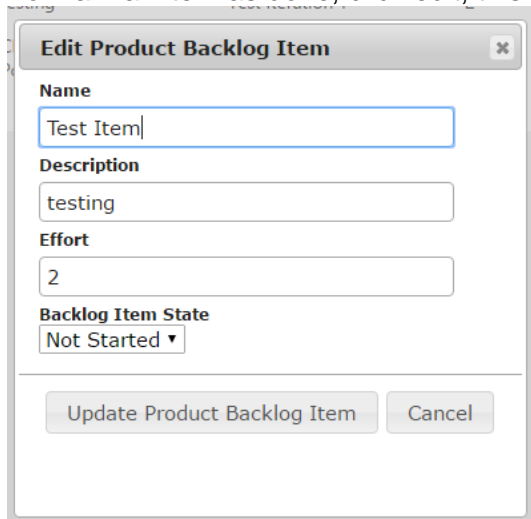
The dialog box titled "Add Project to Iteration" contains a "Select Iteration" dropdown menu. The menu is open, showing options: "Not Assigned" (selected), "Not Assigned", "Test Iteration 1", and "Test Iteration 2". At the bottom right of the dialog is an "Add To Iteration" button.

Figure 52. Add to iteration

When you select not assigned, it will remove any set value. Otherwise, selecting an iteration from this list, then clicking Add to Iteration, will trigger the loading screen and a page refresh. This action will be logged in the log viewer. Once the page refreshes, the iteration name will display in the table:

Name	Description	Current Iteration	Effort	State	
Test Item	testing	Test Iteration 1	2	Not Started	Edit Add to Iteration Delete

When you mark a backlog item as done, the time of this is recorded and used in reporting. To mark an item as done, click edit, this will open the following dialog:



The dialog box titled "Edit Product Backlog Item" contains several input fields: "Name" (with "Test Item" entered), "Description" (with "testing" entered), and "Effort" (with "2" entered). Below these is a "Backlog Item State" dropdown menu set to "Not Started". At the bottom are "Update Product Backlog Item" and "Cancel" buttons.

Figure 53. Edit PBI Dialog

This allows you to edit the item as well as change its state.

Finally, clicking on the name of a Product backlog item will take you to its details page which is where you can add tasks.

6.2.3 Product Backlog Item Details

[Back to Project Details](#)

Test Item

Item Details
[Edit Backlog Item](#)

Tasks
[Add Task](#)

Description
testing

Current Iteration
Test Iteration 1

Effort
2

State
Not Started

No Tasks available for this backlog item

Figure 54. Product Backlog Item Details Page

On the product backlog items page, if you are a team member or above, you have the option to both edit the backlog item as well as add and manage tasks associated with the backlog item. If the backlog item is in the current sprint however, this can also be achieved from the SCRUM board.

Editing the backlog item is the same as if you edit it on the other page, and adding a task is also similar. When you click on Add Task the following dialog is displayed:

Add Task [X]

Name

Description

Time Remaining

State
Not Started ▼

Assign User
Christopher Haworth ▼

Figure 55. Add Task Dialog

The Users list is populated by checking the teams that are on the project, and then displaying what users are in those teams. If not assigned is selected, then no user is assigned to the task.

When you have finished in this dialog, click add task. This again will display a short loading screen, then your task will be created, as shown below:

Test Item

Item Details
[Edit Backlog Item](#)

Tasks
[Add Task](#)

Description
testing

Name
Test Task

Description

Remaining Time

Current User
Christopher Haworth

State
Not Started

[Edit](#) [Delete](#)

Current Iteration
Test Iteration 1

Effort
2

State
Not Started

Figure 56. Added New Task

From here you can also remove tasks which will caused them to be archived, and require an administrator to restore them.

6.3 Reports

To access the reports section of the site, you need to be a minimum of a stakeholder in the system. Once you click on the link, you are presented with a list of projects that your user can access as shown below:

Reports

- [TestProject2](#)
- [Test Project 18](#)
- [Test Project 19](#)
- [Test Project 3](#)
- [Final Year Project](#)

Figure 57. Reports Page

When you click on one of these, this list will hide itself, then the report page will display for that project e.g:

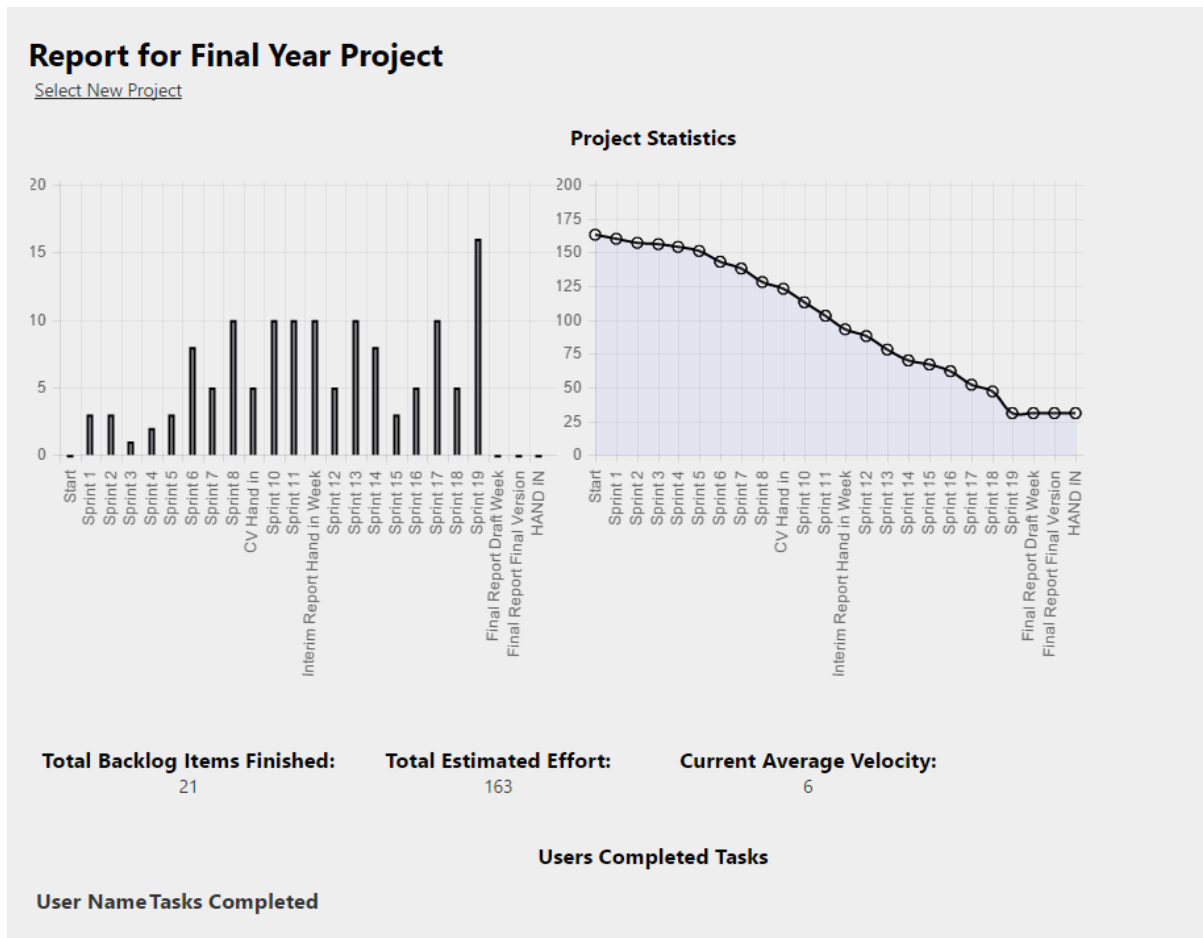


Figure 58. Report page

From here you can see the Velocity graph and project burndown, along with the total number of backlog items finished to date, the total estimated effort, and current average velocity. Also you will get a list of how many tasks each user has completed, this could be used to identify areas of improvement within the team if a user is slipping behind if the tasks taken were smaller tasks.

From this screen, clicking Select new project will return you back to the initial list shown on the previous page.

7. Frequently Asked Questions/Known Issues

7.1 Known Issues

- Iteration List on project details page randomly orders itself when items are added to iterations
- On some pages the edit dialogs do not correctly populate after already being open and closed without committing the changes
-

7.2 Frequently Asked Questions

1. System is displaying an unauthorized page?
 - a. Contact your system administrator to ensure you have got the required level of access for this page. Send a screenshot of the error message with it.
2. System is saying I am blocked; is there any way I can undo this?
 - a. No, either you have been added to the blocked list of users within Active Directory or your Administrator has expressly blocked you from the site. If you believe this to be in error, you will need to contact the administrator to ask why you have been blocked.

Appendix E: Technical Documentation

Again the contents page has been stripped in this to stop it from messing with the overall report contents page

The logo for OS-Scrum features the text "OS-Scrum" in a large, blue, serif font. To the left of the text is a vertical blue line. Below the main text, the words "API Technical Documentation Version 1" are written in a smaller, blue, sans-serif font.

OS-Scrum

API Technical Documentation Version 1

Christopher D. Haworth
4-18-2016

1. Introduction

This document is designed as an accompaniment to the source code and will go into technical detail regarding the API that the system uses to allow 3rd party systems to interact with the system.

2. API Overview

The API in this application is a stateless API that returns JSON objects as responses to the different queries given. It also requires that the user that is interacting with the API has the same level of permissions within the overall application to access the specific functions of the API for security purposes.

The Following class diagram is an overview of the available functions of this API:

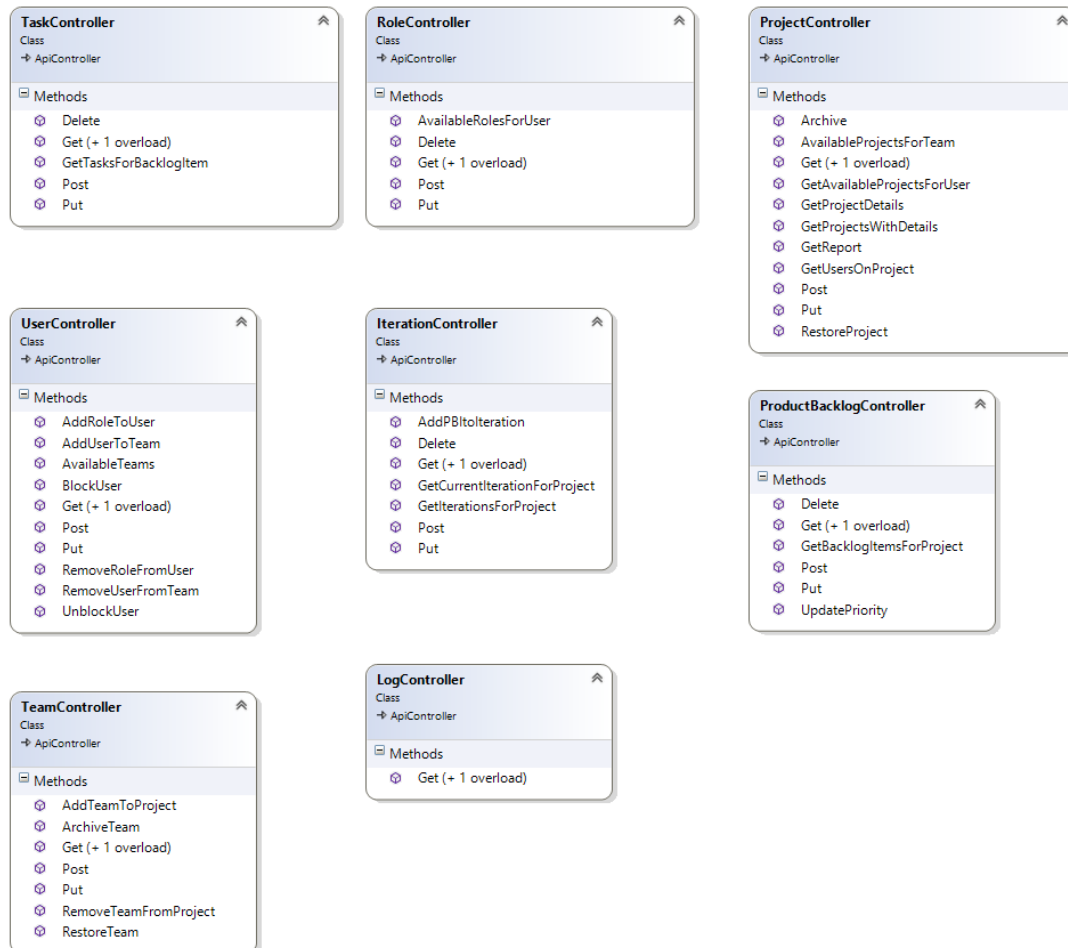


Figure 59. API Class Diagram

The API uses the same authentication system as the Web Interface to the application, therefore any requests will need to be authenticated in the same manner as the website.

3. API Method Index

Unless stated otherwise, if the method throws an error, null is returned and the error is logged into the database.

3.1 Project

HTTP Get Methods:

- Get()
 - URL: url/api/Project
 - Requires minimum role of Stakeholder
 - Returns an IEnumerable of ProjectDTO
- Get(int id)
 - URL: url/api/Project/{id}
 - Accessed by passing the id after Project
 - Requires minimum role of Stakeholder
 - Returns ProjectDTO for given ID or null
- GetProjectsWithDetails()
 - URL: url/api/Project/GetAllProjectDetails
 - Requires minimum role of Scrum Master
 - Returns IEnumerable of ProjectDetailsDTO
- GetProjectDetails(int ProjectID)
 - URL: url/api/Project/GetProjectDetails/{ProjectID}
 - Requires minimum role of Scrum Master
 - Returns ProjectDetailsDTO
- GetUsersOnProject(int ProjectID)
 - URL: url/api/Project/GetUsersOnProject/{ProjectID}
 - Requires minimum role of Team Member
 - Returns IEnumerable of UserDTO
- GetAvailableProjectsForUser()
 - URL: url/api/Project/GetAvailableProjectsForUser
 - Requires minimum role of Stakeholder
 - Returns IEnumerable of ProjectDTO
- AvailableProjectsForTeam(int teamID)
 - URL: url/api/Project/GetAvailableProjectsForTeam/{teamID}
 - Requires minimum role of Scrum Master
 - Returns IEnumerable of ProjectDTO
- GetReport(int projectID)
 - URL: url/api/Report/{projectID}
 - Requires minimum role of Stakeholder
 - Returns Report

HTTP Post/Put/Delete:

- Post([FromBody]ProjectDTO project)
 - POST URL: url/api/Project
 - Requires minimum role of Scrum Master
 - Returns 0 if error adding project, otherwise returns project id
 - Requires a serialized ProjectDTO passed in the body of the request
- Put(int id, [FromBody]ProjectDTO project)
 - PUT URL: url/api/Project/{id}
 - Requires minimum role of Scrum Master
 - Returns 1 if successful or 0 if not

- Requires an id in the url and a serialized ProjectDTO passed in the body of the request
- Archive(int id)
 - DELETE URL: url/api/Project/{id}
 - Requires minimum role of Scrum Master
 - Returns either “Complete” or “Error” depending on success or failure
- RestoreProject(int id)
 - POST URL: url/api/Project/Restore/{id}
 - Requires minimum role of Scrum Master
 - Returns either “Complete” or “Error” depending on success or failure

3.2 Product Backlog

HTTP GET Methods:

- Get()
 - URL: url/api/ProductBacklog
 - Requires minimum role of Scrum Master
 - Returns IEnumerable of ProductBacklogItemDTO
- Get(int id)
 - URL: url/api/ProductBacklog/{id}
 - Requires minimum role of Stakeholder
 - Returns ProductBacklogItemDTO
- GetBacklogItemsForProject(int ProjectID)
 - URL: url/api/ProductBacklog/Project/{projectID}
 - Requires minimum role of Stakeholder
 - Returns IEnumerable of Product BacklogItemDTO

HTTP POST/PUT/DELETE Methods:

All of the following methods require a minimum of role of Team Member apart from update priority

- Post([FromBody] ProductBacklogItemDTO pbi)
 - POST URL: url/api/ProductBacklog
 - Returns id of added PBI or 0 if error
 - Requires serialized ProductBacklogItemDTO in request body
- Put(int id, [FromBody] ProductBacklogItemDTO pbi)
 - PUT URL: url/api/ProductBacklog/{id}
 - Requires serialized ProductBacklogItemDTO in request body
 - Returns 1 for success or 0 for failure
- Delete(int id)
 - DELETE URL: url/api/ProductBacklog/{id}
 - Returns “Complete” or “Error”
- UpdatePriority([FromBody] ChangePBIPriority)
 - POST URL: url/api/ProductBacklog/UpdatePriority
 - Requires minimum role of Stakeholder
 - Requires a serialized ChangePBIPriority object in request body

3.3 Iterations

HTTP GET Methods:

- Get()
 - URL: url/api/Iterations
 - Requires minimum role of Scrum Master
 - Returns IEnumerable of IterationDTO

- Get(int id)
 - URL: url/api/Iterations/{id}
 - Requires minimum role of Stakeholder
 - Returns IterationDTO
- GetIterationsForProject(int projID)
 - URL: url/api/Iterations/GetProjectIteration/{projID}
 - Requires minimum role of Stakeholder
 - Returns IEnumerable of IterationDTO
- GetCurrentIterationForProject(int projID)
 - URL: url/api/Iterations/GetCurrentIterationForProject/{projID}
 - Requires minimum role of Team Member
 - Returns IterationDetailsDTO

HTTP POST/PUT/DELETE Methods:

- Post([FromBody]IterationDTO)
 - POST URL: url/api/Iterations
 - Returns id of added Iteration or 0 if error
 - Requires serialized IterationDTO in request body
 - Requires minimum role of Team Leader
- AddPBIToIteration([FromBody]SetIteration)
 - POST URL: url/api/Iterations/AddPBIToIteration
 - Requires minimum role of Team Member
 - Requires serialized SetIteration in request body
- Put(int id, [FromBody]IterationDTO)
 - PUT URL: url/api/Iterations/{id}
 - Returns id of added Iteration or 0 if error
 - Requires serialized IterationDTO in request body
 - Requires minimum role of Team Leader
- Delete(int ID)
 - DELETE URL: url/api/Iterations/{id}
 - Returns "Complete" or "Error"
 - Requires minimum role of Team Leader

3.4 Tasks

All Methods require a minimum of Team Member to use apart from the Generic Get which requires a minimum level of Scrum Master.

HTTP GET Methods:

- Get()
 - URL: url/api/BacklogTasks
 - Requires minimum role of Scrum Master
 - Returns IEnumerable of BacklogItemTaskDTO
- Get(int id)
 - URL: url/api/BacklogTasks/{id}
 - Returns BacklogItemTaskDTO
- GetTasksForBacklogItem(int backlogID)
 - URL: url/api/BacklogTasks/BacklogItem/{backlogitemID}
 - Returns IEnumerable of BacklogItemTaskDTO

HTTP POST/PUT/DELETE Methods:

- Post([FromBody]BacklogItemTaskDTO)
 - POST URL: url/api/BacklogTasks

- Requires serialized BacklogItemTaskDTO in request body
 - Returns task id or 0 if errored
- Put (int id, [FromBody]BacklogItemTaskDTO)
 - PUT URL: url/api/BacklogTasks/{id}
 - Requires serialized BacklogItemTaskDTO in request body
 - Returns 1 for success or 0 for error
- Delete(int id)
 - DELETE URL: url/api/BacklogTasks/{id}
 - Returns "Complete" or "Error"

3.5 Users

HTTP GET Methods:

- Get()
 - URL: url/api/Users
 - Requires minimum role of View Only
 - Returns IEnumerable of UserDTO
- Get(int ID)
 - URL: url/api/Users/{id}
 - Requires minimum role of View Only
 - Returns UserDTO or null
- AvailableTeams(int userID)
 - URL: url/api/Users/AvailableTeams/{userID}
 - Requires minimum role of View Only
 - Returns IEnumerable of TeamDTO
- UserDetails(int userID)
 - URL: url/api/Users/UserDetails/{userID}
 - Requires minimum role of View Only
 - Returns UserDetailsDTO

HTTP POST/PUT/DELETE Methods:

With the exception of block and unblock user, the following methods require a minimum role level of Scrum Master.

- Post([FromBody]UserDTO)
 - POST URL: url/api/Users
 - Requires serialized UserDTO in body of request
 - Returns user id or 0 if error
- Put(int id, [FromBody]UserDTO)
 - PUT URL: url/api/Users/{userID}
 - Requires serialized UserDTO in body of request
 - Returns 1 if successful or 0 if error
- BlockUser(int id)
 - DELETE URL: url/api/Users/BlockUser/{userID}
 - Requires Administrator role
 - Returns "Complete" or "Error"
- UnblockUser(int id)
 - POST URL: url/api/Users/UnblockUser/{userID}
 - Requires Administrator role
 - Returns "Complete" or "Error"
- AddRoleToUser([FromBody]UserToRole)
 - POST URL: url/api/Users /AddRoleToUser
 - Requires serialized UserToRole object in body of request

- Returns 1 for success and 0 for error
- RemoveRoleFromUser([FromBody]UserToRole)
 - POST URL: url/api/Users/RemoveRoleFromUser
 - Requires serialized UserToRole object in body of request
 - Returns 1 for success and 0 for error
- AddUserToTeam([FromBody]UserToTeam)
 - POST URL: url/api/Users/AddUserToTeam
 - Requires serialized UserToTeam object in body of request
 - Returns 1 for success and 0 for error
- RemoveUserFromTeam([FromBody]UserToTeam)
 - POST URL: url/api/Users/RemoveUserFromTeam
 - Requires serialized UserToTeam object in body of request
 - Returns 1 for success and 0 for error

The API does allow for finer grain control of users than the Web UI currently offers. Including the ability of a scrum master to have a separate program that can interact with the dataset to manage user roles directly.

3.6 Roles

HTTP GET Methods:

- Get()
 - URL: url/api/Roles
 - Requires a minimal role of View Only
 - Returns IEnumerable of RoleDTO
- Get(int ID)
 - URL: url/api/Roles/{roleID}
 - Requires a minimal role of View Only
 - Returns RoleDTO
- AvailableRolesForUsers(int userID)
 - URL: url/api/Roles/GetRolesForUser/{userID}
 - Requires a minimal role of View only
 - Returns List of RoleDTO
- GetDetails(int id)
 - URL: url/api/Roles/Details/{roleID}
 - Requires a minimal role of view only
 - Returns RoleDetailsDTO

HTTP POST/PUT/DELETE Methods:

The following methods all require a role level of Administrator to function

- Post([FromBody]RoleDTO)
 - POST URL: url/api/Roles
 - Requires a serialized RoleDTO in the request body
 - Returns Role ID or 0 if error
- Put(int id, [FromBody]RoleDTO)
 - PUT URL: url/api/Roles/{roleID}
 - Requires a serialized RoleDTO in the request body
 - Returns 1 for success or 0 for failure
- Delete(int id)
 - DELETE URL: url/api/Roles/{roleID}
 - Returns "Complete" or "Error"

3.7 Teams

HTTP GET Methods:

The following methods require a minimal role of View Only

- Get()
 - URL: url/api/Teams
 - Returns IEnumerable of TeamDTO
- Get(int id)
 - URL: url/api/Teams/{id}
 - Returns TeamDTO
- GetDetails(int id)
 - URL: url/api/Teams/Details/{id}
 - Returns TeamDetailsDTO

HTTP POST/PUT/DELETE Methods:

The following methods require a minimal role of Scrum Master

- Post([FromBody]TeamDTO)
 - POST URL: url/api/Teams
 - Requires serialized TeamDTO in message body
 - Returns team id or 0 if error
- Put(int id, [FromBody]TeamDTO)
 - PUT URL: /url/api/Teams/{id}
 - Requires serialized TeamDTO in message body
 - Returns 1 for success and 0 for error
- ArchiveTeam(int id)
 - DELETE URL: url/api/Teams/ArchiveTeam/{id}
 - Returns "Complete" or "Error"
- RestoreTeam(int id)
 - POST URL: url/api/Teams/RestoreTeam/{id}
 - Returns "Complete" or "Error"
- AddTeamToProject([FromBody]TeamToProject)
 - POST URL: url/api/Teams/TeamToProject
 - Requires serialized TeamToProject object in request body
 - Returns "Ok" or "Error"
- RemoveTeamFromProject([FromBody]TeamToProject)
 - DELETE URL: url/api/teams/TeamToProject
 - Requires serialized TeamToProject object in request body
 - Returns "Ok" or "Error"

3.8 Logs

Logs require a minimum role of Scrum Master to view

HTTP GET Methods:

- Get()
 - URL: url/api/Logs
 - Returns IEnumerable of LogDetailsDTO
- Get(int id)
 - URL: url/api/Logs/{id}
 - Returns LogDetailsDTO

4. Object Types

The Application will generate a series of JSON Objects (as shown below) to match the different requests and responses as described in the previous section. Each of the following JSON Objects can also be returned in a list from the server.

4.1 ProjectDTO

```
{
  "$id": "1",
  "ProjectId": 2,
  "ProjectName": "TestProject2",
  "ProjectDetails": "Test Details for TestProject2",
  "Archived": false
}
```

4.2 ProjectDetailsDTO

```
{
  "$id" : "1",
  "Archived" : false,
  "ProductBacklogItems" : [
    {
      "$id" : "2",
      "CurrentIteration" : {
        "$id" : "3",
        "IterationID" : 2,
        "ProjectID" : 2,
        "SprintStartDate" : "2/22/2016",
        "SprintEndDate" : "2/28/2016",
        "SprintName" : "Second Sprint",
        "Archived" : false
      },
      "PBITasks" : [
        {
          "$id" : "4",
          "CurrentUserInfo" : {
            "$id" : "5",
            "UserID" : 2,
            "UserName" : "HAWORTH\\Christopher",
            "FirstName" : "Chris",
            "LastName" : "Haworth",
            "EmailAddress" : ""
          },
          "BacklogItemTaskID" : 3,
          "ProductBacklogID" : 28,
          "Name" : "Test Task",
          "Description" : "test",
          "RemainingTime" : 3,
          "CurrentUserID" : 2,
          "State" : 1,
          "Archived" : false
        }
      ],
      "ProductBacklogItemID" : 28,
      "ProjectID" : 2,
      "Name" : "Test PBI",

```

```
"Data" : "AA==",
"Description" : "Testing Add Dialog Box",
"SprintID" : 2,
"EffortScore" : 1,
"Priority" : 1,
"State" : 2,
"Archived" : false,
"DateFinished" : null
}
],
"Teams" : [
{
  "$id" : "23",
  "Users" : [
    {
      "$id" : "24",
      "UserID" : 2,
      "UserName" : "HAWORTH\\Christopher",
      "FirstName" : "Chris",
      "LastName" : "Haworth",
      "EmailAddress" : ""
    },
    {
      "$id" : "25",
      "UserID" : 4,
      "UserName" : "HAWORTH\\Fyp_TeamLeader",
      "FirstName" : "FYP",
      "LastName" : "TeamLeader",
      "EmailAddress" : ""
    }
  ]
},
"AssignedProjects" : [
{
  "$id" : "26",
  "ProjectId" : 2,
  "ProjectName" : "TestProject2",
  "ProjectDetails" : "Test Details for TestProject2",
  "Archived" : false
},
{
  "$id" : "27",
  "ProjectId" : 30,
  "ProjectName" : "Test Project 3 ",
  "ProjectDetails" : "Testing new Ajax Callback",
  "Archived" : false
},
{
  "$id" : "28",
  "ProjectId" : 31,
  "ProjectName" : "Final Year Project",
  "ProjectDetails" : "OS-Scrum Project",
  "Archived" : false
}
],
"TeamID" : 1,
```

```

    "TeamName" : "Team 1",
    "Archived" : false
  }
],
"Iterations" : [
  {
    "$id" : "29",
    "productBacklogItems" : [
      {
        "$id" : "30",
        "CurrentIteration" : {
          "$id" : "31",
          "IterationID" : 2,
          "ProjectID" : 2,
          "SprintStartDate" : "2/22/2016",
          "SprintEndDate" : "2/28/2016",
          "SprintName" : "Second Sprint",
          "Archived" : false
        },
        "PBITasks" : [
          {
            "$id" : "32",
            "CurrentUserInfo" : {
              "$id" : "33",
              "UserID" : 2,
              "UserName" : "HAWORTH\\Christopher",
              "FirstName" : "Chris",
              "LastName" : "Haworth",
              "EmailAddress" : ""
            },
            "BacklogItemTaskID" : 3,
            "ProductBacklogID" : 28,
            "Name" : "Test Task",
            "Description" : "test",
            "RemainingTime" : 3,
            "CurrentUserID" : 2,
            "State" : 1,
            "Archived" : false
          }
        ],
        "ProductBacklogItemID" : 28,
        "ProjectID" : 2,
        "Name" : "Test PBI",
        "Data" : "AA==",
        "Description" : "Testing Add Dialog Box",
        "SprintID" : 2,
        "EffortScore" : 1,
        "Priority" : 1,
        "State" : 2,
        "Archived" : false,
        "DateFinished" : null
      }
    ],
    "IterationID" : 2,
    "ProjectID" : 2,

```

```

        "SprintStartDate" : "2/22/2016",
        "SprintEndDate" : "2/28/2016",
        "SprintName" : "Second Sprint",
        "Archived" : false
    }
],
    "ProjectId" : 2,
    "ProjectName" : "TestProject2",
    "ProjectDetails" : "Test Details for TestProject2"
}

```

4.3 ProductBacklogItemDTO

```

{
    "$id" : "1",
    "Archived" : false,
    "ProductBacklogItemID" : 28,
    "ProjectID" : 2,
    "Name" : "Test PBI",
    "Data" : "AA==",
    "Description" : "Testing Add Dialog Box",
    "SprintID" : 2,
    "EffortScore" : 1,
    "Priority" : 1,
    "State" : 2,
    "DateFinished" : null
}

```

4.4 ProductBacklogItemDetailsDTO

```

{
    "$id" : "47",
    "CurrentIteration" : {
        "$id" : "48",
        "IterationID" : 3,
        "ProjectID" : 2,
        "SprintStartDate" : "2/15/2016",
        "SprintEndDate" : "2/21/2016",
        "SprintName" : "Third Sprint",
        "Archived" : false
    },
    "PBITasks" : [
        {
            "$id" : "49",
            "CurrentUserInfo" : {
                "$id" : "50",
                "UserID" : 2,
                "UserName" : "HAWORTH\\Christopher",
                "FirstName" : "Chris",
                "LastName" : "Haworth",
                "EmailAddress" : ""
            },
            "BacklogItemTaskID" : 1,
            "ProductBacklogID" : 33,
            "Name" : "Sort out task testing",
            "Description" : "Testing Dialogs ",
            "RemainingTime" : 2,
            "CurrentUserID" : 2,

```

```

    "State" : 1,
    "Archived" : false
  },
  {
    "$id" : "51",
    "CurrentUserInfo" : {
      "$id" : "52",
      "UserID" : 2,
      "UserName" : "HAWORTH\\Christopher",
      "FirstName" : "Chris",
      "LastName" : "Haworth",
      "EmailAddress" : ""
    },
    "BacklogItemTaskID" : 2,
    "ProductBacklogID" : 33,
    "Name" : "Test Dialogs",
    "Description" : "Test Current Dialog Implementation",
    "RemainingTime" : 2,
    "CurrentUserID" : 2,
    "State" : 1,
    "Archived" : false
  }
],
"ProductBacklogItemID" : 33,
"ProjectID" : 2,
"Name" : "Dialog Test",
"Data" : "AA==",
"Description" : "Testing new Dialog Implementation :) Mega Yay",
"SprintID" : 3,
"EffortScore" : 1,
"Priority" : 2,
"State" : 2,
"Archived" : false,
"DateFinished" : "3/25/2016"
}

```

4.5 ChangePBIPriority (Taken From Fiddler)

pbilDs%5B%5D=30&pbilDs%5B%5D=33&pbilDs%5B%5D=28&pbilDs%5B%5D=29

4.6 IterationDTO

```

{
  "$id" : "1",
  "IterationID" : 11,
  "ProjectID" : 31,
  "SprintStartDate" : "10/21/2015",
  "SprintEndDate" : "10/23/2015",
  "SprintName" : "Sprint 3",
  "Archived" : false
}

```

4.7 IterationDetailsDTO

```

{
  "$id" : "1",
  "productBacklogItems" : [
    {

```

```

"$id" : "2",
"CurrentIteration" : {
  "$id" : "3",
  "IterationID" : 32,
  "ProjectID" : 31,
  "SprintStartDate" : "5/2/2016",
  "SprintEndDate" : "5/6/2016",
  "SprintName" : "HAND IN",
  "Archived" : false
},
"PBITasks" : [
  {
    "$id" : "4",
    "CurrentUserInfo" : {
      "$id" : "5",
      "UserID" : 2,
      "UserName" : "HAWORTH\\Christopher",
      "FirstName" : "Chris",
      "LastName" : "Haworth",
      "EmailAddress" : ""
    },
    "BacklogItemTaskID" : 9,
    "ProductBacklogID" : 55,
    "Name" : "Make Video",
    "Description" : "Create Video for Project",
    "RemainingTime" : 2,
    "CurrentUserID" : 2,
    "State" : 1,
    "Archived" : false
  }
],
"ProductBacklogItemID" : 55,
"ProjectID" : 31,
"Name" : "HAND IN ",
"Data" : "AA==",
"Description" : "All Finish",
"SprintID" : 32,
"EffortScore" : 1,
"Priority" : 25,
"State" : 0,
"Archived" : false,
"DateFinished" : null
}
],
"IterationID" : 32,
"ProjectID" : 31,
"SprintStartDate" : "5/2/2016",
"SprintEndDate" : "5/6/2016",
"SprintName" : "HAND IN",
"Archived" : false
}

```

4.9 SetIteration (Taken From Fiddler)

Pbild=30&IterationId=3

4.10 BacklogItemTaskDTO

```
{
  "$id" : "1",
  "BacklogItemTaskID" : 1,
  "ProductBacklogID" : 33,
  "Name" : "Sort out task testing",
  "Description" : "Testing Dialogs ",
  "RemainingTime" : 2,
  "CurrentUserID" : 2,
  "State" : 1,
  "Archived" : false
}
```

4.11 BacklogItemTaskDetailsDTO

```
{
  "$id" : "4",
  "CurrentUserInfo" : {
    "$id" : "5",
    "UserID" : 2,
    "UserName" : "HAWORTH\\Christopher",
    "FirstName" : "Chris",
    "LastName" : "Haworth",
    "EmailAddress" : ""
  },
  "BacklogItemTaskID" : 9,
  "ProductBacklogID" : 55,
  "Name" : "Make Video",
  "Description" : "Create Video for Project",
  "RemainingTime" : 2,
  "CurrentUserID" : 2,
  "State" : 1,
  "Archived" : false
}
```

4.12 UserDTO

```
{
  "$id" : "1",
  "UserID" : 2,
  "UserName" : "HAWORTH\\Christopher",
  "FirstName" : "Chris",
  "LastName" : "Haworth",
  "EmailAddress" : ""
}
```

4.13 UserDetailsDTO

```
{
  "$id" : "1",
  "UserInRoleMap" : [
    {
      "$id" : "2",
      "UserID" : 2,
      "RoleID" : 2,
      "isExplicit" : false
    }
  ],
  "CurrentRoles" : [
```

```

    {
      "$id" : "3",
      "RoleID" : 2,
      "RoleName" : "System Administrator",
      "AdGroupName" : "FYP_Admin_Group",
      "RoleType" : 0
    }
  ],
  "CurrentTeams" : [
    {
      "$id" : "4",
      "TeamID" : 1,
      "TeamName" : "Team 1",
      "Archived" : false
    }
  ],
  "UserID" : 2,
  "UserName" : "HAWORTH\\Christopher",
  "FirstName" : "Chris",
  "LastName" : "Haworth",
  "EmailAddress" : ""
}

```

4.14 UserToRole (Taken from Fiddler)

userID=4&roleID=5

4.15 UserToTeam (Taken from Fiddler)

userID=4&teamID=2

4.16 RoleDTO

```

{
  "$id" : "1",
  "RoleID" : 2,
  "RoleName" : "System Administrator",
  "AdGroupName" : "FYP_Admin_Group",
  "RoleType" : 0
}

```

4.17 RoleDetailsDTO

```

{
  "$id" : "1",
  "UsersInRole" : [
    {
      "$id" : "2",
      "UserID" : 2,
      "RoleID" : 2,
      "isExplicit" : false
    }
  ],
  "RoleID" : 2,
  "RoleName" : "System Administrator",
  "AdGroupName" : "FYP_Admin_Group",
  "RoleType" : 0
}

```

4.18 TeamDTO

```
{
  "$id" : "1",
  "TeamID" : 1,
  "TeamName" : "Team 1",
  "Archived" : false
}
```

4.19 TeamDetailsDTO

```
{
  "$id" : "1",
  "Users" : [
    {
      "$id" : "2",
      "UserID" : 2,
      "UserName" : "HAWORTH\\Christopher",
      "FirstName" : "Chris",
      "LastName" : "Haworth",
      "EmailAddress" : ""
    },
    {
      "$id" : "3",
      "UserID" : 4,
      "UserName" : "HAWORTH\\Fyp_TeamLeader",
      "FirstName" : "FYP",
      "LastName" : "TeamLeader",
      "EmailAddress" : ""
    }
  ],
  "AssignedProjects" : [
    {
      "$id" : "4",
      "ProjectId" : 2,
      "ProjectName" : "TestProject2",
      "ProjectDetails" : "Test Details for TestProject2",
      "Archived" : false
    },
    {
      "$id" : "5",
      "ProjectId" : 30,
      "ProjectName" : "Test Project 3 ",
      "ProjectDetails" : "Testing new Ajax Callback",
      "Archived" : false
    },
    {
      "$id" : "6",
      "ProjectId" : 31,
      "ProjectName" : "Final Year Project",
      "ProjectDetails" : "OS-Scrum Project",
      "Archived" : false
    }
  ],
  "TeamID" : 1,
  "TeamName" : "Team 1",
  "Archived" : false
}
```

}

4.20 TeamToProject (Taken from Fiddler)

teamID=4&ProjectID=2

References

- Anon., 2011. *Dynamic Systems Development Method*. [Online]
Available at: <http://dsdmofagilemethodology.wikidot.com/>
[Accessed 04 05 2016].
- Beck, K. et al., 2001. *Manifesto for Agile Software Development*. [Online]
Available at: <http://agilemanifesto.org/>
[Accessed 14 January 2016].
- Bloch, M., Blumberg, S. & Laartz, J., 2012. *Delivering large-scale IT projects on time, on budget, and on value*. [Online]
Available at: http://www.mckinsey.com/insights/business_technology/delivering_large-scale_it_projects_on_time_on_budget_and_on_value
[Accessed 14 October 2015].
- Boehm, B. W., 1988. A Spiral Model of Software Development and enhancement. *Computer*, 21(5), pp. 61-72.
- Firesmith, D., 2013. *Using V Models for Testing*. [Online]
Available at: https://insights.sei.cmu.edu/sei_blog/2013/11/using-v-models-for-testing.html
[Accessed 16:58 January 2016].
- Freedman, R., 2010. *Four variants of agile development methods*. [Online]
Available at: <http://www.techrepublic.com/blog/tech-decision-maker/four-variants-of-agile-development-methods/>
[Accessed 14 January 2016].
- International SCRUM institute, n.d. *Sprint Burndown Reports/Charts*. [Online]
Available at: http://www.scrum-institute.org/Sprint_Burndown_Reports.php
[Accessed 09 October 2015].
- James, M., n.d. *Scrum Reference Card*. [Online]
Available at: <http://scrumreferencecard.com/ScrumReferenceCard.pdf>
[Accessed 08 October 2015].
- Mahalakshmi, M. & Sundararajan, D. M., 2013. Traditional SDLC Vs Scrum Methodology - A Comparative Study. *International Journal of Emerging Technology and Advanced Engineering*, 3(6), pp. 192-196.
- Microsoft, n.d. *TFS requirements and compatibility*. [Online]
Available at: <https://msdn.microsoft.com/en-us/library/vs/alm/tfs/administer/requirements>
[Accessed 16 04 2016].
- Microsoft, n.d. *Velocity & forecasting*. [Online]
Available at: <https://msdn.microsoft.com/en-us/Library/vs/alm/Work/scrump/velocity-and-forecasting>
[Accessed 09 October 2015].
- Palmquist, M. S. et al., 2013. *Parallel Worlds: Agile and Waterfall Differences and Similarities*. [Online]
Available at: <http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA610501>
[Accessed 12 October 2015].

Poppendieck.LLC, 2015. *The Lean Mindset*. [Online]
Available at: <http://www.poppendieck.com/>
[Accessed 04 05 2016].

Schwaber, K. & Sutherland, J., 2013. *The Scrum Guide*. [Online]
Available at: <http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf#zoom=100>
[Accessed 08 October 2015].

Sparrow, P., 2011. *Spiral Model: Advantages and Disadvantages*. [Online]
Available at: <http://www.ianswer4u.com/2011/12/spiral-model-advantages-and.html#axzz3xJB0Djwm>
[Accessed 15 01 2016].

Trello Inc., n.d. *About Trello*. [Online]
Available at: <https://trello.com/about>
[Accessed 18 January 2016].

Trello Inc., n.d. *Trello*. [Online]
Available at: <https://trello.com/home>
[Accessed 18 January 2016].

Wells, D., 2009. *Extreme Programming: A gentle Introduction*. [Online]
Available at: <http://www.extremeprogramming.org/>
[Accessed 04 05 2016].