Mid-Project Deliverable

CSCI311 – Software Process Management

The Team

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| James Wilson |  |  | *Project Manager* |
| James Glennan |  |  | *Software Architect* |
| Joshua Brown |  |  | *System Data Specialist* |
| Kurt Robinson |  |  | *Tool Specialist and Change Control Manager* |
| Peter Brown |  |  | *Requirements Analyst and User Interface Engineer* |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| Feasibility Study |  |  | 1 |
| Risks |  |  | 8 |
| SRS |  |  | ? |
| Detailed Plan |  |  | ? |
| Milestones |  |  | ? |
| Objectives |  |  | ? |
| Visual Data Document |  |  | ? |
| Progress Report |  |  | ? |
| Code of Conduct |  |  | ? |
| Project diaries |  |  | ? |
| Contributions |  |  | ? |
|  |  |  |  |

Feasibility Study

Looking into Software Design and Collaboration Tools

# Executive Summary

This document outlines some of the considerations associated with developing a source code, bug tracking, and software design collaboration tool. Although individually these tools have existed independently of each other, it has been decided that I would be best to incorporate these tools into a ‘super collaboration’ tool, giving all the power of source control, mixed with bug tracking and reporting; while allowing communications between the team to be tracked during the software development process.  
One such software comes close in providing this service, but with the ready availability of multiple tools are open-source or freeware, so why not apply a mix readily existing tools, providing the development community at large, with the tools necessary to complete complex design tasks with minimal concern.

# Product Services

The problem with a product such as this is that, a single all in one solution isn’t readily available on the market, software such as ‘Bugzilla’, is able to accurately keep track of bugs, whereas software such as ‘Gource’ is excellent visual tool for seeing member contributions to software repositories. But none of these packages do everything, ‘Unfuddle’ probably being the closes

## Gource

One aspect of the software that’s being commissioned is being able to graphically visualise the changes being performed to the repository, by tracking the logs (through trunks, commits and checkouts) (acaudwell, 2013). Gource, another open source project provides an amazing display to visualise the data as it makes changes- of course this software is inherently limited since it provides no real other features related to this specification.

## Bugzilla

According to the Bugzilla website, this open source software tracks bugs and code changes, allows you to communicate with teammates and submit/review patches (Bugzilla.org, 2012). It markets itself as being able to “reduce downtime, increase productivity, raise customer satisfaction, and improve communication”.

## Fossil

Fossil (Fossil, 2014) is probably the closest to the sort of software that we’re after. It provides a web-based interface with graphical tools, bug tracking and support for creating your own wiki. The main issues with using this software is that email control, since the requirements specify importing project emails into the source control, although the use of the Wiki could work around this.

## Unfuddle

Unfuddle appears to fill most of the requirements, allowing bug tracking and management of source code, this software can also allow collaboration through a messaging service and wiki. The major downside of this software is that unlike all the previous software, its not freeware, and requires a minimum $15 a month (Unfuddle).

## Product/Service Marketplace

The only reputable competition to this software came from Unfuddle, and with their signup fees, would be a major upset, especially for the casual developer who prefers to use free and easy tools.  
All the other tools failed to capture all the necessity, and it should be noted that even Unfuddle still doesn’t quite do what’s required, its just happens to be the closest thing.  
This sort of software, due to its cost, would be targeted at developers and businesses that can both afford and are large enough to require using a comprehensive source control strategy. Smaller developers have to rely on existing open source and freeware tools, in order to collaborate and create their products, its here that exists a niche market for the serious but small developer.

# Marketing Strategy

Since this software is more likely to be used in-house, there doesn’t really require marketing a program like this. However, if this program is to be released externally, then it should be released as free, allowing for all users to popularise its use and become a staple of the software development community. In order to generate a profit (or at least break-even), advertising space can be bought to complement this software and pay for its development. As well as this, for large organisations a premium service can be offered that gives companies extra support and training for its staff, for a nominal fee, and as such generating a profit.

# Technology Considerations

## Development Environment

This service would potentially be used by a wide number of users each with many different Operating Systems and Configurations, as such, to combat this, the service needs to be platform independent, which can be achieved through use of a web service, requiring a web browser.

## Web Services

|  |  |  |
| --- | --- | --- |
| Service | Advantages | Disadvantages |
|  |  |  |
| Apache | Supports FastCGI  Has a module for PHP  Widely used and well documented  Good support  Runs on all Operating Systems (Best on Linux)  Open source (Free) | Can be very heavy as it supports many features |
|  |  |  |
| Nginx | Supports FastCGI  Runs on all Operating System (Best on Linux)  Extremely Lightweight  Growing and becoming more popular  Open source (Free)  Has easiest to read configuration | Does not support as many features as Apache  Relatively new to other web servers  Not as mature or tested |

## Language Implementation

|  |  |
| --- | --- |
| Language | Features |
|  |  |
| PHP | Object Oriented  Weakly Typed  Just in Time Compilation  Normally has a web server running in front of it (Nginx, Apache, ISS) |
|  |  |
| Java | Object Oriented  Strongly Typed  Compiled to byte code, ten interpreted by Java VM  Normally has a web server running in front of it (Apache Tomcat, Oracle) |
|  |  |
| C++ | Object Oriented  Strongly Typed  Compiled  Robust  Not easy to parse files with |
|  |  |
| Ruby on Rails | Object Oriented  Can easily parse files  Database management somewhat automatic  Easy to learn  High Overhead |

## Storage Solution

|  |  |
| --- | --- |
| RDBMS | Features |
|  |  |
| SQLITE | Creates files on disk containing data, not good for reading from or storing large amounts of data  Does not support all SQL statements  Does not support multiple users |
|  |  |
| MySQL | Widely supported and widely used on the web  Implements well with scripting language such as PHP or Ruby  Open source |

# Organisation and Staffing

### James Wilson – *Project Manager*

James is directing the group as well as coordinating back and front-end integration. James will monitor the individual module’s development direction and progress as well as ensuring that weekly reporting/testing suites are behind provided for the relevant sections.

### James Glennan – *Software Architect*

James is required to help implement software solutions and learn new languages that may be required. This job will require specific detail in ensuring that tests are designed for all of the system’s modules as well as ensuring that code is implemented as efficiently as possible.

### Joshua Brown – *System Data Specialist*

Josh has experience in database implementation and as such is fully capable of designing and implementing our data storage solution.

### Kurt Robinson – *Tool Specialist and Change Control Manager*

Kurt is hosting the software repository as well conducting research and learning about specific open-source components that can be implemented in to this project. He is tasked with third party implementation, as well as liaising with the relevant authorities to ensure that the product’s modules connect cohesively.

### Peter Brown – *Requirements Analyst and User Interface Engineer*

Peter is in charge of implementing the front-end of the software, specifically regarding how our user interacts with the product. Peter will also ensure that the interfaces are intuitive, and well documented such that anybody with limited knowledge of these systems will be able learn how to use the product.

# Schedule

For a more comprehensive schedule, please see the project plan.

1. Environment Set Up
2. Set up development environment
3. Set up MySQL database
4. Set up coding environment
5. Set up website tools
6. Milestone – Environment Set Up
7. Iteration 1.1 – Website Construction
8. Analysis
9. Design
10. Development
11. Testing
12. Review
13. Milestone – Website Constructed
14. Iteration 1.2 – User Management System
15. Analysis
16. Design
17. Development
18. Testing
19. Review
20. Milestone – User Management System Complete
21. Iteration 2.1 – Import Data
22. Analysis
23. Design
24. Code
25. Testing
26. Review
27. Milestone – Import Data Complete
28. Iteration 2.2 – Data Translation
29. Analysis
30. Design
31. Code
32. Testing
33. Review
34. Milestone – Data Translation Complete
35. Iteration 2.3 – Data Manipulation
36. Analysis
37. Design
38. Code
39. Testing
40. Review
41. Milestone – Data Manipulation Complete
42. Iteration 2.4 – Data Visualisation
43. Analysis
44. Design
45. Code
46. Testing
47. Review
48. Milestone – Data Visualisation Complete
49. Develop Final Test Script
50. Deployment
51. User Manual
52. Testing with Client
53. Approved By Client
54. Evaluation

## Financial Projections

At this stage there isn’t a real projection of financial income due to the nature of this software. If the above suggestions are taken on-board, that is, use the software to advertise and offer training programs, it can be assumed that money can be made. However this hinges on the fact that this software must become popular to start with, and that is why it will be released as freeware.

## Findings and Recommendations

It is our recommendation that software be immediately be commissioned to fill a need since collaboration isn’t cost effectively managed by any existing software. The conclusion being that a better and more tailored piece of software can be developed, that meets the requirements specified for this project.

This product will be designed around a three module system. Front-end that interfaces with the user, back-end that parses and stores the RAW data and the middle component that provides the bridge between both by making sense of the raw data and delivering it to the front to be displayed in a aesthetic and intuitive way. To do this, a web server utilising Nginx, will serve a web page that draws its data from a MYSQL database imported by our Ruby on Rails back-end reader. We believe this modular design will assist in ease of development, as well as allowing us to make modification as we go to ensure correctness and function of the system.

Risks

Risks that are taken throughout the course of this project

risks

Software Requirements Specification

Requirements provided that our software must meet

# SRS Contents

### Revision History

### Introduction

* 1. Purpose
  2. Scope
  3. Definitions, Acronyms and Abbreviations
  4. References
  5. Overview

### Overall Description

* 1. Product Perspective
  2. Product Functions\*
  3. User Characteristics
  4. Constraints
  5. Assumptions and Dependencies

### Specific Requirements

* 1. Requirements

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 25/03/2014 | 0.1 | First Draft | Peter Brown |
| 26/03/2014 | 0.2 | Continuation of Draft   * Added first Drafts of Sections 2 & 3 | Peter Brown |
| 27/03/2014 | 0.2.1 | Minor fixing of draft | Peter Brown |
| 03/04/2014 | 1.0 | First draft | Peter Brown |
| 04/04/2014 | 1.01 | Revision of User Management requirements | Peter Brown |
|  |  |  |  |
|  |  |  |  |

# 1. Introduction

## 1.1 Purpose

The Purpose of this document is to describe the specifications, requirements and risk evaluation for our Assignment in CSCI311. It contains non-functional requirements, design constraints and other factors necessary to provide a complete and comprehensive understanding of our project. The intended audience of this document includes the Client, Development team and the assignment supervisor/s.

## 1.2 Scope

The aim of this software solution is to develop a tool that can be used as a dashboard for a given software project. It is intended to be able to intake raw data and allow visualization and manipulation of it. It is intended for a client that wants to be able to easily understand a variety of different information types in one easy software solution.

## 1.3 Definitions, Acronyms, and Abbreviations

**Apache Ant** – Apache Ant is a Java library and command-line tool whose mission is to drive processes described in build files as targets and extension points dependent upon each other. The main known usage of Ant is the build of Java applications. Ant supplies a number of built-in tasks allowing to compile, assemble, test and run Java applications. Ant can also be used effectively to build non Java applications, for instance C or C++ applications. More generally, Ant can be used to pilot any type of process which can be described in terms of targets and tasks.

**Python** - Python is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C. The language provides constructs intended to enable clear programs on both a small and large scale

## 1.4 References

## 1.5 Overview

The rest of this document is divided into two main sections:

* The Overall Description (section 2) describes the general factors that affect the system and its requirements.
* The Specific Requirements (section 3) contains all software requirements that the system must meet in order to satisfy customer’s needs.

# 2. Overall Description

## 2.1 Product perspective

Currently there is no existing dashboard system so we are working solely off the requirements and feedback from the user. As there is already a central data store which is provided by the SVN archives we have a localised store of raw information. This information in its raw form is hard to manage and relate unless you are heavily involved with the project and know the design of the data store or have a working knowledge of data repositories. This makes it hard for those not directly involved with the project to track the status of the project and managing the correspondence such as bug reports, update logs and emails sent to the developers.

From this we see the need to centralise the supporting data for the projects and our system aims at extracting the raw data into a dashboard which will be in a more easily interpretable format and represent the project’s information better.

### 2.1.1 System Interfaces

The Dashboard to be developed is a web-based application that can be integrated to an organization’s

Intranet or deployed on the Internet.

The *Clients* can simultaneously log into the system from any PC that supports Internet services and then view the displayed information in various different representations of their choice.

The *Project Manager* can add and edit the stored data through an administration menu which is available when they are logged into the system. They also can view the information like a client would be able to.

The *Web Server* allows client machines to connect to the Dashboard system and serves their needs via a set of web pages. It acts as a mechanism to accept and process client requests, then retrieve the database and finally return the results to these requests and display them in the correct format.

The *Database* is used to store the system’s user profiles, Dashboards resources and the stored archive of data from the project.

### 2.1.2 User Interfaces

The user interfaces provided to clients must be user friendly and accessible through the browser (IE, Mozilla, Firefox etc). The user interfaces are different between the Project Manager, Admin and a regular user. Project managers and Admins would be able to manipulate and add data to the system through a separate control panel. Admins would have a higher access which would also allow them to edit the structure of the site and its contents.

All accesses to the Database, which is powered by MySQL server, must be performed indirectly through

GUIs provided by the Dashboard system.

### 2.1.3 Hardware Interfaces

All components must be able to execute on a personal computer (PC or Mac).

### 2.1.4 Software Interfaces

#### 2.1.4.1 External System Interface:

* The system is able to import archived data from the SVN.

#### 2.1.4.2 User Interface:\*

* The User must be able to access the dashboard through a web browser
* The System must support the current web browsers of Chrome, IE and Firefox as of March 2014 (\*\*\*\*)

#### 2.1.5 Communication Interfaces\*

* The client machines must communicate with the Web Server over TCP/IP connection
* The Web Server and the Database Server are located on different servers (\*\*\*\*)

#### 2.1.6 Memory Constraints\*

* The client machine must be able to operate within 32MB minimum (including memory for browser)
* The Web Server and the Database Server must be able to operate within 128MB minimum

#### 2.1.7 Operations\*

The Dashboard system must be easy for all users to use, e.g. no specific information or skills (except knowledge on how to access the Internet via Web browser) must be required to use the tool.

The Charting and Data representation must be concise and easy to interpret for people with a decent knowledge of different charting styles.

The Web Server installation and maintenance should be simple enough for a network administrator to perform and should not require any special technical skills from the administrator.

The Database Server should be able to import data from the SVN. Backup and Recovery operations must be specified in case of network failure, database failure, out of power etc.(\*\*\*\*)

## 2.2 Product functions\*

The main functions of the Dashboard system are to retrieve the raw project data, store and display it in a more easily interpretable format such as charts and diagrams.

For Users the Dashboard system helps them to understand the project by (\*\*\*\*)

* Interpreting Raw project Data from the archive and giving it meaning
* Helping them to visualize the project data in a meaningful way which includes:
  + Bug Reports
  + Test Cases
  + Execution traces/logs,
  + Emails
  + Discussions
  + Real world user feedback
  + Maintenance evolution activities
  + Progress

Project Managers must also be able to (in addition to basic users):

* Add and Edit data directly into the system

Admins Must be able to:

* Access all levels of information
* Manage user profiles and accounts on the system

## 2.3 User characteristics

The users of the Dashboard system include System Admins, General Users and Project Managers.

* Administrators have strong knowledge on networks and web applications to be able to install and maintain the Dashboard system.
* Project managers have a good knowledge of the data being worked with and be able to update and change data as well as understand how it is represented.
* General users have a good understanding of the data being represented and its format.

## 2.4 Constraints

The system should strictly obey and satisfy the following constraints:

* Authentication security: the system should enforce user authentication security
* Access control: the system must provide appropriate access right and user interface to each type of user (for instance, administrators and librarians are allowed to access resources profiles and borrower profiles while borrowers are not allowed to)
* Backup and recovery: the backup and recovery of all the system’s database must be easy to perform to prevent databases from corruption and loss risks
* Integrity control: since the system consists of many databases that are correlated with each other, integrity among these databases must be strictly maintained.
* The system must be developed within 13 Weeks and must be released by the end of the session.

## 2.5 Assumptions and dependencies

The following assumptions and dependencies for the system are stated:

* All Users of the system will have an email address
* All Users of the System will understand basic console navigation and be able to read interpreted data

# 3. Specific Requirements

Each requirement (either functional or non-functional one) of the Dashboard system is ranked based on its level of importance.

1. Critical: highest importance level. Critical requirements are those that reflect core functionalities of Dashboard system and must be firstly implemented
2. Essential: second highest importance level. Essential requirements are those that reflect important functionalities of the Dashboard system and must be covered when Critical requirements have successfully implemented
3. Desirable: medium importance level. Desirable requirements are those that reflect necessary but not critical functionalities of the Dashboard system and should be implemented when all Critical and Essential requirements have been fulfilled
4. Optional: lowest importance level. Optional requirements are those that reflect enhanced functionalities of the Dashboard system and should be considered only when all Critical, Essential and Desirable requirements are completed

## 3.1 Requirements

### 3.1.1 Key requirements

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.1\_01 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | The dashboard should be able to import and parse existing raw data | | | | | | |
| **Rationale** | Importing and parsing existing raw data (e.g. bug/issue reports, test cases, execution traces/logs, emails, discussions, real-world user feedback, etc.) From a given large-scale open source project. | | | | | | |
| **Criteria** | Data is stored and translated into relevant data | | | | | | |
| **Source** | SVN Archive, All Users | | | | | | |
| **Created** | Created by Peter Brown 27/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.1\_02 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | Users Must be able to Manipulate Data through the Systems GUI | | | | | | |
| **Rationale** | The user (Project Manager) must be allowed to manipulate the data (e.g. Create new tasks, requests, assigning tasks/requests, generating reports etc.) | | | | | | |
| **Criteria** | Stored data can be manipulated | | | | | | |
| **Source** | Data Store, Project Manager, Admin | | | | | | |
| **Created** | Created by Peter Brown 27/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.1\_03 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | The dashboard must translate data into a visual representation | | | | | | |
| **Rationale** | Visualizing those data in a meaningful way to depict the current status of the project in various aspects (e.g. developers’ activities, expertise and networks, user feedback, maintenance and evolution activities and progress, etc.). | | | | | | |
| **Criteria** | Stored data is translated into visual data | | | | | | |
| **Source** | Data Store, Users | | | | | | |
| **Created** | Created by Peter Brown 27/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

### 3.1.2 User Management Module

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.2\_01 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | The dashboard must have a User Management System with differing levels of access | | | | | | |
| **Rationale** | Different Users have access to different levels of information (client) | | | | | | |
| **Criteria** | The dashboard has a user management system that stores a unique username and password for people to log in to the system | | | | | | |
| **Source** | Data Store, Dashboard, User, Admin | | | | | | |
| **Created** | Created by Peter Brown 03/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.2\_02 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | The Administrator must be able to add accounts to the User management system | | | | | | |
| **Rationale** | New users must be added to the system (Req# 3.1.2\_07) | | | | | | |
| **Criteria** | The admin is able to add an account to the user management system | | | | | | |
| **Source** | Data Store, Dashboard, Admin, User management System | | | | | | |
| **Created** | Created by Peter Brown 04/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

### 3.1.3 Data Management Module

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.2\_01 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | The data must be pulled from the SVN | | | | | | |
| **Rationale** | The dashboard should be able to import and parse existing raw data (Req# 3.1.1\_01) | | | | | | |
| **Criteria** | Data is retrieved from the SVN | | | | | | |
| **Source** | SVN Archive, Raw Data | | | | | | |
| **Created** | Created by Peter Brown 27/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.2\_02 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | The raw data must be translated into information | | | | | | |
| **Rationale** | The dashboard should be able to import and parse existing raw data (Req# 3.1.1\_01) | | | | | | |
| **Criteria** | Data is translated into meaningful information | | | | | | |
| **Source** | Raw Data, Data store, Users | | | | | | |
| **Created** | Created by Peter Brown 27/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

### 3.1.4 Main Dashboard Module

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.2\_03 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | The data must be Accessed from the dashboard | | | | | | |
| **Rationale** | Visualizing those data in a meaningful way to depict the current status of the project in various aspects (e.g. developers’ activities, expertise and networks, user feedback, maintenance and evolution activities and progress, etc.). (Req# 3.1.1\_03) | | | | | | |
| **Criteria** | Data is able to be retrieved from the Data store by the dashboard | | | | | | |
| **Source** | Data Store, Dashboard | | | | | | |
| **Created** | Created by Peter Brown 03/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.2\_04 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | The data must be Represented Visually in the dashboard | | | | | | |
| **Rationale** | Visualizing those data in a meaningful way to depict the current status of the project in various aspects (e.g. developers’ activities, expertise and networks, user feedback, maintenance and evolution activities and progress, etc.). (Req# 3.1.1\_03) | | | | | | |
| **Criteria** | Data is able to be shown in a meaningful visual format | | | | | | |
| **Source** | Data Store, Dashboard | | | | | | |
| **Created** | Created by Peter Brown 03/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

### 3.1.5 Data Manipulation module

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.2\_05 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | New data must be able to be created form the dashboard | | | | | | |
| **Rationale** | The user (Project Manager) must be allowed to manipulate the data (e.g. Create new tasks, requests, assigning tasks/requests, generating reports etc.) (Req#3.1.1\_02) | | | | | | |
| **Criteria** | The user must be able to enter new data | | | | | | |
| **Source** | Data Store, Dashboard, User | | | | | | |
| **Created** | Created by Peter Brown 03/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.2\_06 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | Manipulated data must be saved into the data store | | | | | | |
| **Rationale** | The user (Project Manager) must be allowed to manipulate the data (e.g. Create new tasks, requests, assigning tasks/requests, generating reports etc.) (Req#3.1.1\_02) | | | | | | |
| **Criteria** | The manipulated data is added to the existing data | | | | | | |
| **Source** | Data Store, Dashboard, User | | | | | | |
| **Created** | Created by Peter Brown 03/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

### 3.1.6 Data Security Module

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement#** | 3.1.2\_09 | **Use Case** | 0 | **Requirement Type** |  | **Rank of Importance** | 1 |
| **Description** | The Data must be secure & backed up | | | | | | |
| **Rationale** | Data must be secure (client) | | | | | | |
| **Criteria** | The data is recoverable in the case of an accident | | | | | | |
| **Source** | Data Store, backup | | | | | | |
| **Created** | Created by Peter Brown 04/03/2014 | | | | | | |
| **Edited** | n/a | | | | | | |

Detailed Plan

An in depth view of our project plans

Dp lol

Milestones

Goals that we will work to achieve each week

# Mid-Project Deliverable

## Due April 8th

This milestone involves producing a report of what progress we have made in the planning stages of this project. I chose to produce a Gantt chart so that we can decide on our milestones and list each task that we have to undertake during development so that we can keep track during this project. All members are required to produce different sections of this report as displayed in the table below.

Tools that we require in order to reach this milestone include are Microsoft Word, as no development is required yet since it is all documentation and Microsoft Project, for creating the GANTT chart. In this stage it is very simple.

|  |  |  |
| --- | --- | --- |
| Resources | Tasks | Duration in Days |
|  |  |  |
| Peter | SRS Document | 8 |
|  | Risk Management | 2 |
|  |  |  |
| Jamie | Feasibility Study | 9 |
|  | Code of Conduct | 1 |
|  |  |  |
| Kurt | Progress Reports | 8 |
|  | Milestones Document | 2 |
|  |  |  |
| Josh | Archive Map  Visual Data Document | 1  1 |
|  |  |  |
| James | Detailed Project Plan  Objectives | 8  2 |

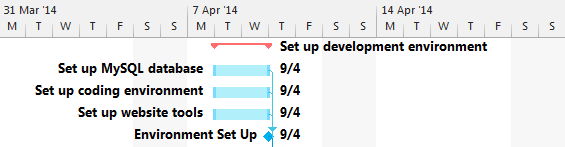
# 

# Environment set up

## Due April 9th

Setting up our development environment includes preparing our MySQL database with our tables prepared to store the imported data. Website hosting also must be set up with necessary tools ready such as user profile creation and security. Our coding environment must be set up identically across each group members computers so that there will be less risk of file incompatibility that may arise when using different versions of software.

|  |  |  |
| --- | --- | --- |
| Resources | Tasks | Duration in Days |
|  |  |  |
| Josh | Set up MySQL Database | 2 |
|  |  |  |
| James | Set up coding environment | 2 |
|  |  |  |
| Peter | Set up website tools | 2 |



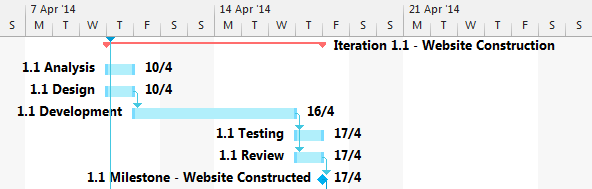
# Iteration 1.1 - Website Constructed

## Due April 17th

When development begins the iterations are split into two paths iterating at the same time (1.1, 1.2 and 2.1, 2.2) as they are not dependent of each other and have different resources assigned to those tasks.

The analysis and design tasks run simultaneously over a day. This is where research of website construction occurs and Peter begins to plan how development takes place. Once developed over multiple days Peter must then test all aspects of what he created and produce a test report then review the work he has done over the last week.

|  |  |  |
| --- | --- | --- |
| Resources | Tasks | Duration in Days |
|  |  |  |
| Peter | 1.1 Analysis | 1 |
|  | 1.1 Design | 1 |
|  | 1.1 Development | 4 |
|  | 1.1 Testing | 1 |
|  | 1.1 Review | 1 |



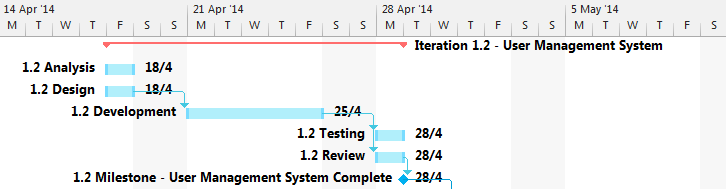
# Iteration 1.2 – User Management System

## Due April 23rd

This milestone is achieved through developing the user management system on the website. This involves enabling a log in service for those accessing the program and giving users different permissions based on their position (such as Administrator, Project Manager, General User).

As Peter handles front end development tasks he will take control of developing this module.

|  |  |  |
| --- | --- | --- |
| Resources | Tasks | Duration in Days |
|  |  |  |
| Peter | 1.2 Analysis | 1 |
|  | 1.2 Design | 1 |
|  | 1.2 Development | 5 |
|  | 1.2 Testing | 1 |
|  | 1.2 Review | 1 |

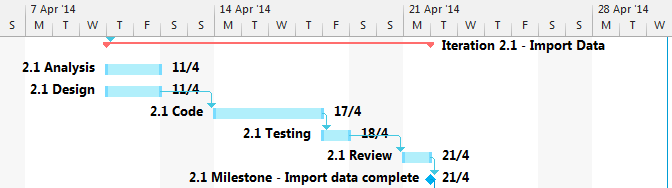


# Iteration 2.1 – Import Data

## Due April 21st

This milestone is achieved by successfully locating and importing the data from the Python SVN page, bugs page and archive files supplied to us into their respective modules in our database.

|  |  |  |
| --- | --- | --- |
| Resources | Tasks | Duration in Days |
|  |  |  |
| Josh, Jamie | 2.1 Analysis | 2 |
|  | 2.1 Design | 2 |
|  | 2.1 Development | 4 |
|  | 2.1 Testing | 1 |
|  | 2.1 Review | 1 |
|  |  |  |
|  |  |  |

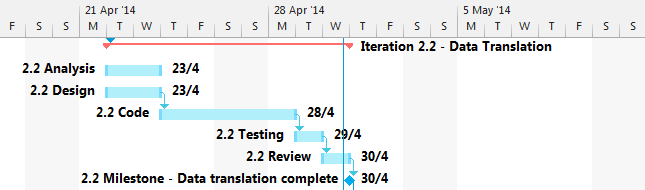


# Iteration 2.2 – Data Translation

## Due April 30th

This milestone is achieved through successfully translating the data imported in iteration 2.1 into data that can be stored within the MySQL database so that it can be accessed through the GUI to show visualisations and be able to be manipulated.

|  |  |  |
| --- | --- | --- |
| Resources | Tasks | Duration in Days |
|  |  |  |
| James, Kurt, Josh, Jamie | 2.2 Analysis | 2 |
|  | 2.2 Design | 2 |
|  | 2.2 Development | 3 |
|  | 2.2 Testing | 1 |
|  | 2.2 Review | 1 |

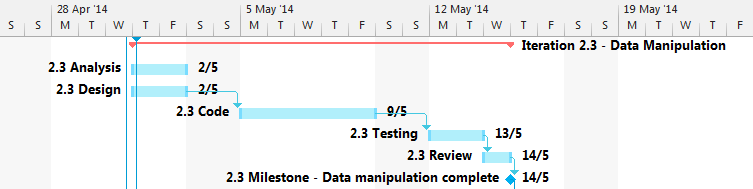


# Iteration 2.3 – Data Manipulation

## Due May 14th

Once we have our data translated and accessible from the database we can then begin to code the data manipulation module which should allow the user (who has the correct permission) to locate any data they wish from the database, edit its attributes, then save it.

|  |  |  |
| --- | --- | --- |
| Resources | Tasks | Duration in Days |
|  |  |  |
| James, Josh, Jamie | 2.3 Analysis | 2 |
|  | 2.3 Design | 2 |
|  | 2.3 Development | 5 |
|  | 2.3 Testing | 2 |
|  | 2.3 Review | 1 |

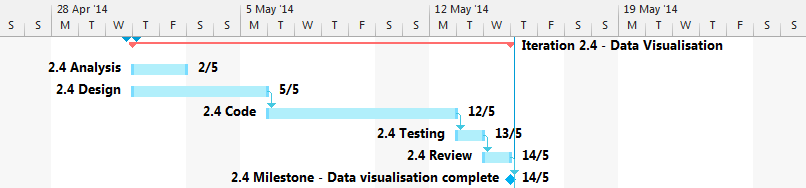


# Iteration 2.4 – Data Visualisation

## Due May 14th

This iteration runs concurrently with iteration 2.3 as they do not depend on each other. They both needed iteration 2.2 complete to access that data. This milestone is achieved by successfully using data stored in the database to convert into a visualisation such as a graph or other appropriate diagram.

|  |  |  |
| --- | --- | --- |
| Resources | Tasks | Duration in Days |
|  |  |  |
| Peter, Kurt | 2.4 Analysis | 2 |
|  | 2.4 Design | 3 |
|  | 2.4 Development | 5 |
|  | 2.4 Testing | 1 |
|  | 2.4 Review | 1 |

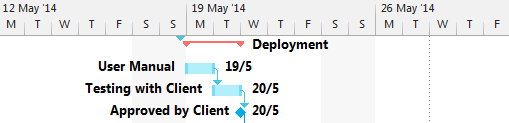


# Approval Testing with Client

## Due May 20th

This milestone is when we expect to have a fully functional program ready to use. As a part of deployment we plan to create a user manual to aid those who may need guidance in using this software. Once we sit down with our client and walk them through our product and receiving approval we will then be able to commence work on our final report.

|  |  |  |
| --- | --- | --- |
| Resources | Tasks | Duration in Days |
|  |  |  |
| Peter, Kurt | User Manual | 1 |
|  |  |  |
| James | Testing with Client | 1 |



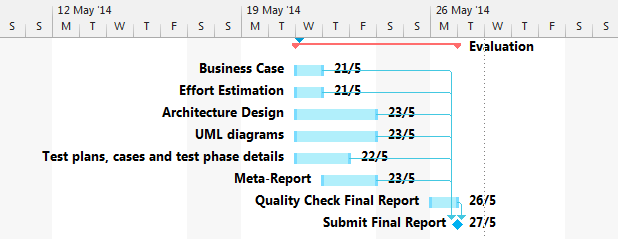
# Submit Final Report

## Due May 20th

Just with the first milestone of the mid-project delivery, this report requires Microsoft word in order to complete documentation. All members of the group are again taking part in this stage with tasks assigned to each.

Once we have completed each section of the final report we will go over it together to ensure that it is of substantial quality and is presented to the client in a well formatted manner.

|  |  |  |
| --- | --- | --- |
| Resources | Tasks | Duration in Days |
|  |  |  |
| Peter | Business Case  Meta-Report | 1  2 |
|  |  |  |
| Jamie | Test phase details, plans and cases | 2 |
| Kurt | UML Diagrams  Effort Estimation | 3  1 |
|  |  |  |
| Josh, James | Architecture Design | 3 |
|  |  |  |
| All Members | Quality Check Final Report | 1 |
|  |  |  |
|  |  |  |



Objectives

subtitle

a

Visual Data Document

Data map of the Python.org repositories

# Purpose

# This document is intended for the development team as a means of reviewing the repositories they are supporting with the dashboard system. It is also as a supplement to the design of the data storage system.

# Contents

## SVN Python Mercurial Repositories

* 1. Repository Map
  2. Repository Attributes
  3. Visualisation of Data

## Bug Tracker Python Repositories

* 1. Repository Map
  2. Repository Attributes
  3. Visualisation of Data

## Mail Python Repositories

* 1. . Repository Map
  2. Repository Attributes
  3. Visualisation of Data

# 1. SVN Python Mercurial Repositories

<http://hg.python.org> – Last Accessed April 7th 2014

## 1.1 Repository Map

### Benchmarks

### Performance benchmarks for Python implementations

### Buildbot

### Buildbot checkouts

* + Empty: The 'empty' env for cleanup of buildbot checkouts

### COC

Code of conduct

### Cpython

The development repository for cpython

### Cpython-fullhistory

The development repository for cpython with full svn history

### Devguide

The Cpython development guide and documentation

### Devinabox

The Python dev in a box development repository

### Distlib

Improved version of distutils

### Distutils2

Improved version distutils

### Extensions

Extensions for CPython

### Features

In development features for python

* + Cdecimal: A development repository for the feature cdecimal
  + Email6: A development repository for the feature email6 and its respectice patchs
  + Finalize: A development repository clone of the web ui
  + Packaging: A development repository for the packaging library
  + Pathlib: A development repository for the experimental pathlib library
  + Pep-3118: A development repository for the feature pep-3118 and its repective bugfixes
  + Pep-3151
  + Pep-3154
  + Pep-3154-alexandre
  + Pep-3155
  + Pep-382
  + Pep-382-2
  + Pep-420
  + Pep-433
  + Pep-443
  + Pep-446
  + Pep-451
  + Pep-456
  + Pep-463 – Secondary Pep-463 repo
  + Pep-463-except-expr: Pep 463 implementation
  + Pep397
  + Randomhash: test repository for the randomized hash function
  + Tracemalloc: A development repository for the new tracemalloc module.
  + Winapi: : A development repository for the rework of the win api

### Gietatu

* + Python: Python 3.4.0 alpha 4

### Hgsubversion

Sub versioning for the hg system

### Hooks

Hooks for the web mercurial service

### Jython

Jython development repository

### Jython-docs

Jython-docs development repository

* + 2.5.x: Clone repo for version 2.5.x
  + Devguide: The Jpython development guide and documentation
  + Website: Clone repo

### Jython-fullhistory

Full Jython development SVN repository

### Jython-releasing

Release branch for Jython

* + 2.5.3: release branch for 2.5.3

### Larry

Testing implantations

* + 340
  + Path\_error2
  + Path\_error2a
  + Path\_error2c

### Peps

Python enchantment proposals

### Platform

* + Plan9: Python 2.7.5 for the plan9 OS

### Python-wiki

Configuration for the python wiki

### Release

Python release script

### Releasing

Release repo clones for python

* + 2.7.6
  + 3.2.3
  + 3.2.4
  + 3.3.0
  + 3.3.1
  + 3.3.3
  + 3.3.4
  + 3.3.5
  + 3.4

### Sanbox

Support repository

* + Python\_shashank

### Sandbox

Experimental and side project repositories, either for features or for personal users

* + A
  + Akuchling
  + Antoine
  + Bcannon
  + Brian
  + Chelmes
  + Cjerdonek-cpython
  + Cococo
  + Cococox
  + Cpython
  + Cpython-issue18978
  + Cpython2
  + Email6-pypi
  + Empty282
  + Empty29
  + Follow\_symlinks
  + Gps
  + Gps-cpython
  + Guido
  + Jcea-cpython-2011
  + Jy3k
  + Jython-website
  + Jython27
  + Loewis
  + Merwok
  + Morph
  + Morph-devguide
  + Ncoghlan
  + Neologix
  + New\_news
  + Nvamda
  + Orsenthil
  + Prefast
  + Python3
  + Raymondh
  + Regex-integration
  + Registervm
  + Relpep
  + Sbt
  + Steve.downer
  + Tkdoc
  + Trent
  + Trent.oracle
  + Trent.peps
  + Vs2010port
  + Vsajip
  + Xxdistlib
  + Zware

### Sandbox1

* + Python3

### Sandbox2

* + Python3

### Stackless

Stackless Python development repository

### Stackless-fullhistory

The full Stackless Python development repository

### Test

Test repo for users

### Tracker

Bug tracker instance for systems

* + Django-gae2django
  + Jython
  + Meta
  + Python-dev
  + Rietveld
  + Roundup
  + Setuptools

### Unittest2

Improved version of unittes

### Users

Users work

* + Alexis: Alexis’ work
  + Barry: Barry Warsaw’s work

## 1.2 Repository Attributes

In the SVN repository the following attributes are relevant

* Author: The creator of the change
* Date: The time the change occurred
* Parent: The parent node of the current entry
* Child: The child node of the current entry
* Files: The applicable files in the system entry
* Files Changed: The amount of files changed
* Insertions: The amount of line insertions
* Deletions: The amount of line deletions
* Description: Description of the entry
* Node ID: A unique identifier for the instance

## 1.3 Visualization of data

In the dashboard system the following visualizations of data coming from the SVN system are recommended:

A Bar graph or histogram comparing subscribed repositories and their recent activity in the last 30 days.

A statistic for the user in the form of either a line graph. Showing total number of files edited, total additions, total deletions, total log entries. As well as an aggregate for all users of the type contributor.

A form based view providing the other statistics in a readable format to the user showing their activity (insertions, deletions, entries, number of files edited, how long they have being committing for).

# 2. Bug Tracker Python Repositories

<http://bugs.python.org> – Last Accessed April 7th 2014

## 2.1 Repository Map

The repository has three main dividing features

Type of bug:

* -no selection-
* Behaviour
* Crash
* Compile error
* Resource usage
* Security
* Performance
* Enhancement

Affected Version:

* 2.7
* 3.1
* 3.2
* 3.3
* 3.4
* 3.5

Component where bug occurred:

* 2to3 (2.x to 3.x conversion tool)
* Benchmarks
* Build
* Cross-Build
* ctypes
* Demos and Tools
* Devguide
* Distutils
* Documentation
* email
* Extension Modules
* IDLE
* Installation
* Interpreter Core
* IO
* Library (Lib)
* Macintosh
* Regular Expressions
* Tests
* Tkinter
* Unicode
* Windows
* XML

## 2.1 Repository Attributes

In the SVN repository the following attributes are relevant

### Assigned to

The user the bug has being assigned to

### Title

The title of the issue listing

### Type

The type of issue

### Version

The versions of python affected

### Components

The applicable components affected by the bug

### Status

Status of the issue (open/closed/pending)

### Stage

The stage of the resolution, (committed/rejected/patch review

### Resolution

Status of the resolution (i.e. committed/rejected)

### Dependencies

Any issues depend on this issue

### Superseder

Any superseding issues to this one

### Nosy List

Users alerted by notifications

### Priority

Priority of said issue (low/medium/high/…)

### Keywords

Tags for the searching for issue

### ID

Unique Identifier number for the issue

### File

Attached file

### File description

Description for the file

### Remote hg repo

Link to the patch from the mercurial repository

### Comment

Allows the user to comment on the issue

### Date

Date when the issue is lodged/last changed

### User

The user involved

### Messages

Message/comment count on the issue

## 2.3 Visualisation of Data

In the dashboard system the following visualizations of data coming from the SVN system are recommended:

A Bar graph or histogram comparing issues and their type. Also comparing versions with total issues by category.

A Bar graph displaying the open and closed issues for the user signed in, as well as for all users. An extra part in this graph displaying it by category will provide the user more information.

A pie graph comparing the components and their relevant percentage of issues.

A form based view providing some raw stats for the user. Total messages on issues which they created/assigned to.

# Mail Python Repositories

<http://mail.python.org> – Last Accessed April 7th 2014

## 3.1 Repository Map

The repository has the following categories

|  |  |  |
| --- | --- | --- |
|  | List | Description |
| A | [Advocacy](https://mail.python.org/mailman/listinfo/advocacy) | Coordination of the Python Advocacy Effort |
|  | [Archiver-dev](https://mail.python.org/mailman/listinfo/archiver-dev) | Development of email archivers |
|  | [Arkansas](https://mail.python.org/mailman/listinfo/arkansas) | *Arkansas Python Group* |
| B | [BangPypers](https://mail.python.org/mailman/listinfo/bangpypers) | Bangalore Python Users Group - India |
|  | [Barcamp](https://mail.python.org/mailman/listinfo/barcamp) | - |
|  | [Baypiggies](https://mail.python.org/mailman/listinfo/baypiggies) | SF Bay Area Python group |
| C | [Canberra-PUG](https://mail.python.org/mailman/listinfo/canberra-pug) | Canberra Python Users Group |
|  | [capi-sig](https://mail.python.org/mailman/listinfo/capi-sig) | Mailing list for discussing the Python/C API |
|  | [CentralOH](https://mail.python.org/mailman/listinfo/centraloh) | Mailing list for Central Ohio Python User Group (COhPy) |
|  | [CentralVA](https://mail.python.org/mailman/listinfo/centralva) | Local user group for Central Virginia USA, based in Charlottesville. |
|  | [Chennaipy](https://mail.python.org/mailman/listinfo/chennaipy) | Chennai Python User Group Mailing List |
|  | [Chicago](https://mail.python.org/mailman/listinfo/chicago) | The Chicago Python Users Group |
|  | [ChiPy-announce](https://mail.python.org/mailman/listinfo/chipy-announce) | The Chicago Python Users Group -- Announce Only List |
|  | [code-quality](https://mail.python.org/mailman/listinfo/code-quality) | A mailing list for code-quality tools related to Python |
|  | [Compiler-sig](https://mail.python.org/mailman/listinfo/compiler-sig) | Discussion of Python compilation |
|  | [concurrency-sig](https://mail.python.org/mailman/listinfo/concurrency-sig) | py concurrency sig |
|  | [Conferences](https://mail.python.org/mailman/listinfo/conferences) | Open discussion of Python conferences, worldwide, existing and planned |
|  | [Core-mentorship](https://mail.python.org/mailman/listinfo/core-mentorship) | Python Core Development Mentorship |
|  | [Cplusplus-sig](https://mail.python.org/mailman/listinfo/cplusplus-sig) | Development of Python/C++ integration |
|  | [Cryptography-dev](https://mail.python.org/mailman/listinfo/cryptography-dev) | *Python dev relating to the topic of cryptography* |
|  | [Csv](https://mail.python.org/mailman/listinfo/csv) | Discussions about CSV support in Python |
|  | [cython-devel](https://mail.python.org/mailman/listinfo/cython-devel) | Core developer mailing list of the Cython compiler |
| D | [DB-SIG](https://mail.python.org/mailman/listinfo/db-sig) | SIG on Python Tabular Databases |
|  | [DCPIGgies](https://mail.python.org/mailman/listinfo/dcpiggies) | Washington DC Python Interest Group |
|  | [Dehradun-Python](https://mail.python.org/mailman/listinfo/dehradun-python) | Mailing List of Dehradun Python User Group |
|  | [Distutils-SIG](https://mail.python.org/mailman/listinfo/distutils-sig) | Python Distribution Utilities |
|  | [Diversity](https://mail.python.org/mailman/listinfo/diversity) | Promoting diversity in the Python community |
|  | [Diversity-SIG](https://mail.python.org/mailman/listinfo/diversity-sig) | Promoting diversity in the Python community |
|  | [Doc-SIG](https://mail.python.org/mailman/listinfo/doc-sig) | Python Documentation Special Interest Group |
|  | [docs](https://mail.python.org/mailman/listinfo/docs) | Mailing list for maintenance of the Python documentation |
| E | [Edinburgh](https://mail.python.org/mailman/listinfo/edinburgh) | Python Edinburgh |
|  | [Edu-sig](https://mail.python.org/mailman/listinfo/edu-sig) | Python in education |
|  | [Elections](https://mail.python.org/mailman/listinfo/elections) | Discussion of PSF elections |
|  | [Email-SIG](https://mail.python.org/mailman/listinfo/email-sig) | SIG for designing and implementing email 3.0 |
|  | [EuroPython](https://mail.python.org/mailman/listinfo/europython) | Planning the European Python community conference |
|  | [EuroPython-announce](https://mail.python.org/mailman/listinfo/europython-announce) | EuroPython announcements |
|  | [Europython-improve](https://mail.python.org/mailman/listinfo/europython-improve) | Discussion list for the EuroPython organisers & active volunteers |
|  | [europython-volunteer-announce](https://mail.python.org/mailman/listinfo/europython-volunteer-announce) | EuroPython volunteer announcement list |
|  | [EuroSciPy](https://mail.python.org/mailman/listinfo/euroscipy) | Planning the EuroSciPy conference |
|  | [euroscipy-org](https://mail.python.org/mailman/listinfo/euroscipy-org) | Organisation of EuroScipy |
|  | [execnet-commit](https://mail.python.org/mailman/listinfo/execnet-commit) | commits for the execnet project |
|  | [execnet-dev](https://mail.python.org/mailman/listinfo/execnet-dev) | Channel/Actor style communication between Python interpreters |
|  | [Expat-bugs](https://mail.python.org/mailman/listinfo/expat-bugs) | Issue tracker activity for Expat |
|  | [Expat-checkins](https://mail.python.org/mailman/listinfo/expat-checkins) | Revision-control actions for Expat |
|  | [Expat-discuss](https://mail.python.org/mailman/listinfo/expat-discuss) | Discussion of the Expat XML parser |
| F | [Fosdem](https://mail.python.org/mailman/listinfo/fosdem) | - |
|  | [FoxPUG](https://mail.python.org/mailman/listinfo/foxpug) | Fox Valley Python Users Group |
|  | [Freelancers](https://mail.python.org/mailman/listinfo/freelancers) | A list for freelance Python developers |
| G | [Glasgow](https://mail.python.org/mailman/listinfo/glasgow) | Python Glasgow |
|  | [Grants-discuss](https://mail.python.org/mailman/listinfo/grants-discuss) | Public discussion of PSF Grants applications |
|  | [Group-Organizers](https://mail.python.org/mailman/listinfo/group-organizers) | Support and Mentoring of Usergroup Organizers (Current or Wannabe) |
|  | [Gurkentruppe](https://mail.python.org/mailman/listinfo/gurkentruppe) | nobody should subcribe here |
| H | [Healthcare](https://mail.python.org/mailman/listinfo/healthcare) | Python community healthcare SIG |
| I | [I18n-sig](https://mail.python.org/mailman/listinfo/i18n-sig) | Python internationalization and localization |
|  | [IDLE-dev](https://mail.python.org/mailman/listinfo/idle-dev) | IDLE development |
|  | [Image-SIG](https://mail.python.org/mailman/listinfo/image-sig) | Image Processing with Python SIG |
|  | [Import-SIG](https://mail.python.org/mailman/listinfo/import-sig) | PEP 382 and other import machinery improvements |
|  | [Inpycon](https://mail.python.org/mailman/listinfo/inpycon) | Mailing list for the PyCon India conference |
|  | [Ipss](https://mail.python.org/mailman/listinfo/ipss) | Indian Python Software Society |
|  | [iQ-users](https://mail.python.org/mailman/listinfo/iq-users) | Support for Andor's iQ OpenSource package |
|  | [Ironpython-users](https://mail.python.org/mailman/listinfo/ironpython-users) | Discussion of IronPython |
| J | [Jython-checkins](https://mail.python.org/mailman/listinfo/jython-checkins) | Check-in messages from the Jython maintainers |
| K | [KolPy](https://mail.python.org/mailman/listinfo/kolpy) | - |
| M | [Mailman-announce](https://mail.python.org/mailman/listinfo/mailman-announce) | Announce-only list for Mailman releases and news |
|  | [Mailman-checkins](https://mail.python.org/mailman/listinfo/mailman-checkins) | Check-in messages from the Mailman maintainers |
|  | [Mailman-coders](https://mail.python.org/mailman/listinfo/mailman-coders) | Mailman list for bug and patch messages |
|  | [Mailman-Developers](https://mail.python.org/mailman/listinfo/mailman-developers) | GNU Mailman developers |
|  | [Mailman-docs](https://mail.python.org/mailman/listinfo/mailman-docs) | Coordinating the Mailman documentation project |
|  | [Mailman-i18n](https://mail.python.org/mailman/listinfo/mailman-i18n) | Mailman Internationalization Effort |
|  | [Mailman-Users](https://mail.python.org/mailman/listinfo/mailman-users) | Mailman mailing list management users |
|  | [MashhadPUG](https://mail.python.org/mailman/listinfo/mashhadpug) | Mashhad Python User Group |
|  | [Medusa-dev](https://mail.python.org/mailman/listinfo/medusa-dev) | Asynchronous programming framework |
|  | [melbourne-pug](https://mail.python.org/mailman/listinfo/melbourne-pug) | Melbourne Python Users Group |
|  | [Meta-sig](https://mail.python.org/mailman/listinfo/meta-sig) | SIG on python.org SIGs and mailing lists |
|  | [Multiprocessing-sig](https://mail.python.org/mailman/listinfo/multiprocessing-sig) | - |
| N | [Ncr-Python.in](https://mail.python.org/mailman/listinfo/ncr-python.in) | NCR Python Users Group India |
|  | [NeedForSpeed](https://mail.python.org/mailman/listinfo/needforspeed) | Reykjavik Sprint |
|  | [New-bugs-announce](https://mail.python.org/mailman/listinfo/new-bugs-announce) | List for new Python bug reports |
|  | [Newsletter](https://mail.python.org/mailman/listinfo/newsletter) | A mailing list for the Python Newsletter Editors... |
| O | [Omaha](https://mail.python.org/mailman/listinfo/omaha) | Omaha Python Users Group |
|  | [oscon](https://mail.python.org/mailman/listinfo/oscon) | Python/PSF planning for OSCON |
| P | [Pandas-dev](https://mail.python.org/mailman/listinfo/pandas-dev) | - |
|  | [Patches](https://mail.python.org/mailman/listinfo/patches) | Mailing list for submission of patches to Python |
|  | [PEPs](https://mail.python.org/mailman/listinfo/peps) | Python Enhancement Proposals (only PEP Editors can subscribe) |
|  | [Planet](https://mail.python.org/mailman/listinfo/planet) | Send requests for additions/changes to the Planet Python aggregator. No subscription requests, please (apart from maintainers). |
|  | [Playground](https://mail.python.org/mailman/listinfo/playground) | A playground for Mailman 2.1 testing |
|  | [PLPUG](https://mail.python.org/mailman/listinfo/plpug) | Polish PUG |
|  | [Portland](https://mail.python.org/mailman/listinfo/portland) | Python Users Group -- Portland, Oregon USA |
|  | [Psf-redesign](https://mail.python.org/mailman/listinfo/psf-redesign) | Python.org RFP Redesign |
|  | [PSF-sponsors](https://mail.python.org/mailman/listinfo/psf-sponsors) | Discussions among PSF sponsor members |
|  | [PSF-Volunteers](https://mail.python.org/mailman/listinfo/psf-volunteers) | Volunteer pool for the PSF |
|  | [PSSI](https://mail.python.org/mailman/listinfo/pssi) |  |
|  | [PyAR2](https://mail.python.org/mailman/listinfo/pyar2) | Arkansas Python User Group - Python ARtists of ARkansas |
|  | [Pycon-announce](https://mail.python.org/mailman/listinfo/pycon-announce) | *-* |
|  | [PyCon-AV](https://mail.python.org/mailman/listinfo/pycon-av) | *-* |
|  | [PyCon-de](https://mail.python.org/mailman/listinfo/pycon-de) | German Python Events and Conferences (in German) |
|  | [Pycon-feedback](https://mail.python.org/mailman/listinfo/pycon-feedback) | *-* |
|  | [Pycon-interest](https://mail.python.org/mailman/listinfo/pycon-interest) | *-* |
|  | [PyCon-organizers](https://mail.python.org/mailman/listinfo/pycon-organizers) | planning PyCon, the annual US Python conference |
|  | [Pycon-se](https://mail.python.org/mailman/listinfo/pycon-se) | *-* |
|  | [Pycon-staff](https://mail.python.org/mailman/listinfo/pycon-staff) | *-* |
|  | [PyCon-Tech](https://mail.python.org/mailman/listinfo/pycon-tech) | Development of PyCon conference software |
|  | [Pycon-volunteers](https://mail.python.org/mailman/listinfo/pycon-volunteers) | *-* |
|  | [PyCon-ZA](https://mail.python.org/mailman/listinfo/pycon-za) | *-* |
|  | [Pyconscot](https://mail.python.org/mailman/listinfo/pyconscot) | *-* |
|  | [pyconuk](https://mail.python.org/mailman/listinfo/pyconuk) | Mailing list for the PyCon UK conference |
|  | [pydotorg-www](https://mail.python.org/mailman/listinfo/pydotorg-www) | Public list for python.org maintenance |
|  | [Pygui](https://mail.python.org/mailman/listinfo/pygui) | PyGUI Developers and Users List |
|  | [Pyiran-organizers](https://mail.python.org/mailman/listinfo/pyiran-organizers) | *-* |
|  | [PyLadies-BLR](https://mail.python.org/mailman/listinfo/pyladies-blr) | PyLadies-BLR |
|  | [PyOhio](https://mail.python.org/mailman/listinfo/pyohio) | General discussion list for attendees of the PyOhio conference |
|  | [PyOhio-announce](https://mail.python.org/mailman/listinfo/pyohio-announce) | Announce-only list for prospective attendees of the PyOhio conference |
|  | [pyop](https://mail.python.org/mailman/listinfo/pyop) | Python On the Peninsula (SF Bay Peninsula) |
|  | [pyopenssl-users](https://mail.python.org/mailman/listinfo/pyopenssl-users) | pyOpenSSL Users List |
|  | [Pypi-checkins](https://mail.python.org/mailman/listinfo/pypi-checkins) | *-* |
|  | [pypy-br](https://mail.python.org/mailman/listinfo/pypy-br) | Desenvolvimento do PyPy em Português do Brasil |
|  | [pypy-commit](https://mail.python.org/mailman/listinfo/pypy-commit) | commits to pypy development |
|  | [pypy-dev](https://mail.python.org/mailman/listinfo/pypy-dev) | development of the PyPy Python implementation |
|  | [pypy-issue](https://mail.python.org/mailman/listinfo/pypy-issue) | *-* |
|  | [PySilesia](https://mail.python.org/mailman/listinfo/pysilesia) | PySilesia initiative group |
|  | [pytest-commit](https://mail.python.org/mailman/listinfo/pytest-commit) | commits for pytest and tox |
|  | [Pytest-dev](https://mail.python.org/mailman/listinfo/pytest-dev) | development of pytest testing tool |
|  | [Python-announce-list](https://mail.python.org/mailman/listinfo/python-announce-list) | Announcement-only list for the Python programming language |
|  | [Python-authors](https://mail.python.org/mailman/listinfo/python-authors) | For authors, editors, reviewers of writings about Python |
|  | [Python-az](https://mail.python.org/mailman/listinfo/python-az) | Azerbaijan Python Users |
|  | [Python-bugs-list](https://mail.python.org/mailman/listinfo/python-bugs-list) | List which receives bug reports on Python |
|  | [Python-checkins](https://mail.python.org/mailman/listinfo/python-checkins) | Check-in messages from the Python maintainers |
|  | [python-committers](https://mail.python.org/mailman/listinfo/python-committers) | Mailing list for Python committers |
|  | [python-crypto](https://mail.python.org/mailman/listinfo/python-crypto) | Cryptography with Python |
|  | [python-de](https://mail.python.org/mailman/listinfo/python-de) | Die Deutsche Python Mailingliste |
|  | [Python-Dev](https://mail.python.org/mailman/listinfo/python-dev) | Python core developers |
|  | [Python-es](https://mail.python.org/mailman/listinfo/python-es) | La lista de python en castellano |
|  | [python-events](https://mail.python.org/mailman/listinfo/python-events) | Python Events Calendar Team List |
|  | [Python-greece](https://mail.python.org/mailman/listinfo/python-greece) | Mailing list for Greek Python User Group. |
|  | [Python-Help](https://mail.python.org/mailman/listinfo/python-help) | Expert volunteers answer Python-related questions |
|  | [Python-HPC](https://mail.python.org/mailman/listinfo/python-hpc) | Python for High Performance and Scientific Computing |
|  | [Python-ideas](https://mail.python.org/mailman/listinfo/python-ideas) | Discussions of speculative Python language ideas |
|  | [Python-ir](https://mail.python.org/mailman/listinfo/python-ir) | Iranian Python Users |
|  | [python-ldap](https://mail.python.org/mailman/listinfo/python-ldap) | Discussion around development, usage etc. of module package ldap |
|  | [Python-legal-sig](https://mail.python.org/mailman/listinfo/python-legal-sig) | Discussion of Python Legal/Compliance issues |
|  | [Python-list](https://mail.python.org/mailman/listinfo/python-list) | General discussion list for the Python programming language |
|  | [Python-mode](https://mail.python.org/mailman/listinfo/python-mode) | For issues concerning python-mode for X/Emacs |
|  | [python-muc](https://mail.python.org/mailman/listinfo/python-muc) | Mailing list for µPy, the Munich Python User Group. |
|  | [python-nigeria](https://mail.python.org/mailman/listinfo/python-nigeria) | Nigerian Python Users Group |
|  | [Python-nl](https://mail.python.org/mailman/listinfo/python-nl) | Dutch Python developers and users |
|  | [Python-porting](https://mail.python.org/mailman/listinfo/python-porting) | Discussion about porting Python code (mainly 2.x -> 3.x) |
|  | [Python-se](https://mail.python.org/mailman/listinfo/python-se) | - |
|  | [Python-Togo](https://mail.python.org/mailman/listinfo/python-togo) | Togolese Python Users (Togo PyUG) |
|  | [python-uk](https://mail.python.org/mailman/listinfo/python-uk) | UK Python Users |
|  | [python-win32](https://mail.python.org/mailman/listinfo/python-win32) | Python on win32 |
|  | [python3-ldap](https://mail.python.org/mailman/listinfo/python3-ldap) | Pure Python 3 LDAP v3 client library |
|  | [PythonCAD](https://mail.python.org/mailman/listinfo/pythoncad) | Mailing list for PythonCAD development |
|  | [PythonCE](https://mail.python.org/mailman/listinfo/pythonce) | Python programming on Windows CE |
|  | [PythonDotNet](https://mail.python.org/mailman/listinfo/pythondotnet) | A list for users and developers of Python for .NET |
|  | [Pythonmac-SIG](https://mail.python.org/mailman/listinfo/pythonmac-sig) | SIG on Python for the Apple Macintosh |
|  | [PyVault-devel](https://mail.python.org/mailman/listinfo/pyvault-devel) | Package development issues |
|  | [PyVault-users](https://mail.python.org/mailman/listinfo/pyvault-users) | For users of the PyVault RPM Repository. |
| S | [sapug](https://mail.python.org/mailman/listinfo/sapug) | South Australian Python Users Group |
|  | [Shtoom](https://mail.python.org/mailman/listinfo/shtoom) | Discussion about Shtoom - a Python VoIP softphone |
|  | [Soap](https://mail.python.org/mailman/listinfo/soap) | SOAP libraries for Python developers |
|  | [SoC2010-General](https://mail.python.org/mailman/listinfo/soc2010-general) | A general list for discussion of Summer of Code 2010 |
|  | [SoC2011-mentors](https://mail.python.org/mailman/listinfo/soc2011-mentors) | A discussion for the mentors of Summer of Code 201` |
|  | [Soc2012-general](https://mail.python.org/mailman/listinfo/soc2012-general) | A general list for discussion of Summer of Code 2012 |
|  | [Soc2013-general](https://mail.python.org/mailman/listinfo/soc2013-general) | A general list for discussion of Summer of Code 2013 |
|  | [Soc2014-general](https://mail.python.org/mailman/listinfo/soc2014-general) | Summer of Code 2014 general discussion list (students and mentors) |
|  | [SpamBayes](https://mail.python.org/mailman/listinfo/spambayes) | Discussion list for Pythonic Bayesian classifier |
|  | [Spambayes-announce](https://mail.python.org/mailman/listinfo/spambayes-announce) | Announcements related to the SpamBayes project |
|  | [Spambayes-bugs](https://mail.python.org/mailman/listinfo/spambayes-bugs) | SF Bug tracker mailing list for SpamBayes project |
|  | [Spambayes-checkins](https://mail.python.org/mailman/listinfo/spambayes-checkins) | Check-in messages for the spambayes project |
|  | [spambayes-dev](https://mail.python.org/mailman/listinfo/spambayes-dev) | Development of the Pythonic Bayesian classifier |
|  | [Speed](https://mail.python.org/mailman/listinfo/speed) | *-* |
|  | [Sprints](https://mail.python.org/mailman/listinfo/sprints) | *-* |
|  | [stdlib-sig](https://mail.python.org/mailman/listinfo/stdlib-sig) | Python standard library development list |
|  | [Sunpiggies](https://mail.python.org/mailman/listinfo/sunpiggies) | Python Users Group -- Flagstaff/Phoenix/Tucson, Arizona USA |
| T | [TehPUG](https://mail.python.org/mailman/listinfo/tehpug) | Tehran Python User Group |
|  | [Texas](https://mail.python.org/mailman/listinfo/texas) | Discussion list for the Texas region Python community |
|  | [Tkinter-discuss](https://mail.python.org/mailman/listinfo/tkinter-discuss) | *-* |
|  | [tkRAD-discuss](https://mail.python.org/mailman/listinfo/tkrad-discuss) | tkRAD: Tkinter XML widget builder |
|  | [Tracker-discuss](https://mail.python.org/mailman/listinfo/tracker-discuss) | Discussion/coordination of the issue tracker used for Python development |
|  | [TriZPUG](https://mail.python.org/mailman/listinfo/trizpug) | Triangle (North Carolina) Zope and Python Users Group |
|  | [Tutor](https://mail.python.org/mailman/listinfo/tutor) | Discussion for learning programming with Python |
| U | [ULS-SIG](https://mail.python.org/mailman/listinfo/uls-sig) | Ultra Large Scale Systems with Python |
| V | [Vm-papers-discuss](https://mail.python.org/mailman/listinfo/vm-papers-discuss) | *-* |
| W | [Web-SIG](https://mail.python.org/mailman/listinfo/web-sig) | SIG for Python support for the Web |
|  | [Winnipeg](https://mail.python.org/mailman/listinfo/winnipeg) | Winnipeg Python Users Group |
| X | [xml-sig](https://mail.python.org/mailman/listinfo/xml-sig) | Python XML Special Interest Group |

## 3.2 Repository Attributes

In the mail repository the following attributes are relevant

* Messages: The total number of messages sent
* Subject: The title of the email chain
* Date: The time the message was sent
* Mailing List: The category of the mail
* Author: The sender of the message
* Thread: Grouping emails by the original message

## 3.3 Visualisation of Data

In the dashboard system the following visualizations of data coming from the SVN system are recommended. It is important to note for this section a larger amount of the data will be aggregated data

A Line graph showing the total number of threads in a given month in a selected mailing list.

A form based display or a bar graph for displaying other stats for a given mailing list such as

* In a given time period (need to consult with the client). The dashboard should display.
* Total number of threads
* Total number of messages
* Total number of messages from the user
* Maximum number of messages on a thread
* Average number of messages on a thread

Progress Report

Weekly reports providing further detail into the current status of our project

# Progress Report for March 19th

## Week One

The project assigned requires us to develop a piece of software that serves as a dashboard capable of visualising data in a meaningful way and allowing the user to manipulate the given data. This progress report was written two days after our first meeting and its purpose is to provide an insight into how development of this project is coming along.

As we have only had one day since our first meeting we were still able to go over our project specification and identify who the stakeholders of the software would be, we also discussed some requirements that the software would have to fulfil as well as some resources that we would require in order to properly develop this software.

During our meeting we dedicated some time into opening the archive files given to us and conducting research to get an early idea of what type of data we would have to work with throughout this project.

## Decision Table

|  |  |  |
| --- | --- | --- |
| Description | Decision | Priority |
|  |  |  |
| Design Methodology | We have chosen to follow the agile development methodology.  It is preferable to each group member as we wish to complete tasks in bursts using an incremental and iterative method. | Neutral |
|  |  |  |
| Programming Language | Our choices were between Cocoa, C++, Java or Visual Basic.  Our final decision will be based on ease of use and its ability to support a GUI, for now it is undecided as it is not a high priority at this point in time. | Low |

## Issues Faces

A large issue we had was not being able to have a group formed until the third week of semester, leaving us only four weeks to complete all our tasks until the first deliverable rather than 6. This means that we will need to organise frequent and extended group meetings in order to produce a quality report in time.

## Action Items

### Completed

*No action items had been set the previous week*

### Assigned

Action items that were assigned at the end of the meeting that we planned to have completed by the next week included a design plan, business case, an initial SRS, measuring effectiveness, milestones, risk analysis and a git repository created for the project.

## Current Status

## 

Our Gantt chart does not show us much regarding how much has been completed since we have only had one day to put work into the task.

# Progress Report for March 26th

## Week Two

This week we had all members present during the Tuesday tutorial so we were able to set a weekly date that we can all get together and simultaneously work on this project. The date chosen that suited us best was on Wednesdays at 5:30pm.

We were able to meet with the client and we gathered some requirements including security details, database updating, and user details. These are taken into more detail within the SRS document.

Peter has chosen to handle development on the front end of the software. Meaning he will handle the website and data visualisation while the others focus on the technical back end tasks.

## Decision Table

|  |  |  |
| --- | --- | --- |
| Description | Decision | Priority |
|  |  |  |
| RDBMS\* | We have chosen to use MySQL as our RDBMS due to its capabilities of supporting concurrent updating across multiple clients since our software will need to support many users at once. | Neutral |
|  |  |  |
| Team Roles | We all discussed preferable parts of the project that we would like to be a part of and were able to determine team roles from this. From doing this we were able to assign tasks to people with the role that would best suit them. Those roles are shown in this report. | High |
|  |  |  |
| Programming Language | We revisited this decision as we have looked into developing the software to be based inside a browser. This gave us two options of languages being PHP, JavaScript and Ruby. JavaScript has a tool named D3js tool that can be used to show graphs and visualise data aesthetically. However most of us decided that Ruby would be the best language for us to develop in as it also has many tools available and it is a simple language for us to learn. | Neutral |

\*RDBMS: Relational Database Management System

## Issues

We had a small issue with communication where two members of our group had left early before our meeting with the client. However we were successfully able to complete our meeting with only three of us and gather more essential requirements.

## Action Items

### Completed

Through further discussion regarding development of the project we were able to come up with a design plan. A risk management report was also completed during the week and the milestones were decided on through creation of the Gantt chart. A git repository was also created with each team member added as collaborators.

### Assigned

Jamie and Kurt were assigned the feasibility study, Peter and Kurt were assigned the SRS, James and Josh were assigned the detailed plan. Everybody was advised to look into the Ruby language in their own time to become more familiar with what we will be coding with.

## Current Status

## 

Once we were given team roles we were than able to work on our given tasks. We are now currently working on the SRS, Progress reports (this document), feasibility study and a project plan. The milestones document still requires completion.

# Progress Report for April 2nd

## Week Three

This week we spent time looking through the Python SVN, bugs website and archive files for bug reports, test cases, execution traces/logs, emails, discussions and emails. We were only able to find the bug reports and emails for now. We are searching for this data so that when we get to our import data stage we would immediately know where the data would come from, thus saving time for ourselves in the future. Peter and Josh also conducted a meeting with our client and received more requirements for the SRS.

The Wednesday meeting consisted of checking up with each other’s progress and helping each other out so we could gain a mutual understanding of different parts of the project.

## Decision Table

|  |  |  |
| --- | --- | --- |
| Description | Decision | Priority |
|  |  |  |
| How to import data | We decided to use Ruby’s integrated unarchiver to access the data supplied to us in the Python archive files. Importing data is an integral part of our project which was why it was classed as a high priority task. | High |
|  |  |  |
|  |  |  |

## Issues

We had an issue with locating the data in SVN and the Python Archives. As we still had current tasks to complete before the deliverable we decided to put this issue on hold for a week while we focused on our report.

## Action Items

### Completed

Action items from last week required more than a week to complete, hence why no items have been completed. Members also studied Ruby during the week.

### Assigned

A code of conduct is to be completed by Jamie, Objectives were assigned to James, completion of an archive map was assigned to Josh, Peter and Josh were also assigned to have a visual data document completed.

## Current Status

## 

We have one week until our deliverable is due and each team member is still working on the same task as the previous week. The milestones document has also been completed.

# Progress Report for April 7th

## Week Four

This is the last week before our deliverable is due, therefore all tasks must be completed before the 8th of April (tomorrow).

All sections of the mid-project deliverable have been complete and we just have to format the document and prepare it for submission.

We must now progress to the iterative execution stage where our Front-end team and our Back-End team will branch off into completing different iterations.

## Decision Table

|  |  |  |
| --- | --- | --- |
| Description | Decision | Priority |
|  |  |  |
| Design Methodology | We originally decided on agile development however we have specifically chosen feature-driven development. We feel that this would suit our project as we have design our development phases into iterations. (See Milestones Document) | Low |
|  |  |  |

## Issues

Because of the first couple of weeks being behind on meetings and progress, we decided that we would host a secondary meeting on the Friday to ensure that all deliverables would be completed.

## Action Items

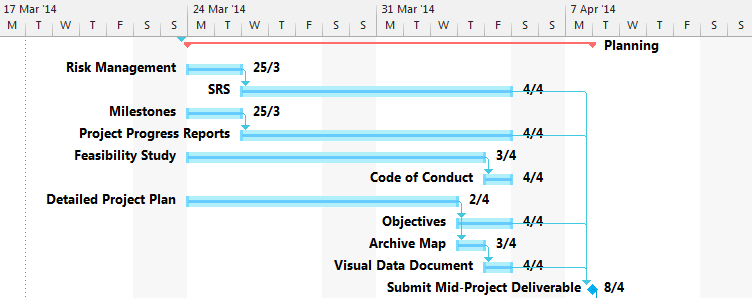
### Completed

All action items from last week have been completed in time for the deliverable.

### Assigned

Josh has been assigned to set up our MySQL database, James has been assigned to set up our code development environment, Jamie has been assigned to locate the required data from SVN and the Python archives while Peter has been assigned to set up tools required to host a website and begin development on that.

## Current Status



All tasks within planning phase of the project is complete and we are prepared to submit our report for the next milestone on the 8th of April. We are now ready to prepare for our execution stage and setting up the development environment.

Code of Conduct

A guide we agree to follow when working together

1. Respect and Courtesy  
   We as a team will endeavor to treat each other, as they would want to be treated. Understand that disagreement is a natural part of teamwork, but that steps should be undertaken to minimize the harm from disagreement including rational (not heated) debate and by democratic vote.
2. Accountability  
   Every member must ensure that they are accountable for the action they take, this includes not only the mistakes, but also the good work that they perform. Each member must also ensure that they perform and deliver the workload that they have committed to, not only as a project, but in minor tasks that they have been asked to perform.
3. Quality  
   It is of the utmost importance that all products delivered are of the utmost quality and that all members do not supply untidy or ill-prepared components; this includes code, as well as reports.
4. Collaborate  
   Never be afraid to ask for, or offer help. Everyone has their own experience, forte’s and weakness. It is of the utmost importance that we collaborate and be supportive to one and another.
5. Responsibility

Each member has responsibilities that they have been allocated, as noted in the Project Plan and Feasibility Study:

### Peter Brown-Requirements Analyst and User Interface Engineer

Peter is in charge of implementing the front-end of the software, specifically regarding how our user interacts with the product. Peter will also ensure that the interfaces are intuitive, and well documented such that anybody with limited knowledge of these systems will be able learn how to use the product.

### James Glennan- Software Architect

James is required to help implement software solutions and learn new languages that may be required. This job will require specific detail in ensuring that tests are designed for all of the system’s modules as well as ensuring that code is implemented as efficiently as possible.

### Kurt Robinson- Tool Analyst and Change Control Manager

Kurt is hosting the software repository as well conducting research and learning about specific open-source components that can be implemented in to this project. He is tasked with third party implementation, as well as liaising with the relevant authorities to ensure that the product’s modules connect cohesively.

### James Wilson- Project Manager

James is directing the group as well as coordinating back and front-end integration. James will monitor the individual module’s development direction and progress as well as ensuring that weekly reporting/testing suites are being provided for the relevant sections.

Joshua Brown- System Data Specialist

Josh has experience in database implementation and as such is fully capable of designing and implementing our data storage solution.

If a problem with an element of this project is found, it is the responsibility of that person to contact the relevant authority within the group.

Project Diaries

An in depth view of our project plans

# James Wilson

# James Glennan

## Week 1 - March 10th

Still currently searching for a new member for the group since, Matt decided to drop this subject, which has left us in a bit of a lurch.  
I’ve started writing what deliverables are required for week 6 as well as the overall specification for the project. As it stands, we have to design a piece of software that is able to handle SVN data as well as other archived data (which I’m not all that sure about yet).

We’re all still trying to work out subjects and work schedules and as such have not come to an agreement about when we can meet (since having another project is also interfering with our ability to figure this out).  
This week I will be attempting to contact anybody that wants to be in a group.

## Week 2 - March 17th

Gathered some details from the client, as well as being able to figure out a meeting time of Wednesday afternoons. We also will meet and discuss in the tutorial (as expected) meaning that we are maximising our contact time. We’ve discussed team roles and as it stands it appears I will be helping manage the database(s) and helping to parse and import the data. I’m still struggling to get anything down from the repository.

I’ve been assigned the feasibility study, to help ascertain other technologies for use, and even what market standards are at. We’ve also decided to look into ruby, and start practicing in development.

## Week 3 - March 24th

Still currently attempting to learn Ruby, as well as completing the Feasibility Document.  
I’ve offloaded the responsibility of looking for the information on the repository to Josh, since he will liaise with the client about display of data.  
  
At this stage we’re meeting 3 times a week in order to complete the necessary documentation, especially since I am now responsible for the Code of Conduct.

# Joshua Brown

# Kurt Robinson

## Week One - March 11th

Me, Jamie Glennan, Peter Brown and Matthew Boroczky initially formed a group of four, however shortly afterwards Matthew dropped the subject and three people wasn't enough so we needed to spend the next week finding people to expand our group.

Even though we did not have a full group we read through the assignment specification anyways to gain an early understanding of what we would be required to develop.

No more progress towards the project was made this week.

## Week Two - March 18th

During the week we were able to find Josh Brown and James Wilson to join our group to give us a total of 5 people. Jamie, Josh and myself were present in our first meeting during the 311 lab on Tuesday.

Action items from this meeting consisted of constructing a design plan and business case, an initial SRS document, listing our objectives, measuring effectiveness, milestones, making a git repository and completing a risk analysis.

## Week Three - March 26th

Since last week a git repository has been set up by myself using Github and the 4 other members have been added to it. We also decided to meet on Wednesday nights to have extended meetings and work together as a group with easier communication between us.

Action items from this meeting to be completed for next week included learning how to code in ruby, completing a GANTT chart showing all the tasks and milestones we could think of (for now) and completing the progress reports for each week. For the reports I will use the meeting minutes and expand what we have done as a group and challenges/decisions we have faced along the way.

## Week Four - April 2nd

We went over our deliverables one more time and made sure that each member were assigned different parts of the report. During the past week I had done some tutorials on Ruby programming to prepare myself for this project. I found it to be a simple language suitable to be used with a web application and database interaction. During this phase of the project where our current milestone to achieve is having documentation prepared for the mid-project there is not much to recount on in these project diaries.

# Peter Brown

Contributions

Agreement of each team members level of contribution to the project

|  |  |  |
| --- | --- | --- |
| James Wilson – jw192 |  |  |
| Contributed |  |  |
|  |  |  |
| Jamie Glennan – jg631 |  |  |
| Contributed |  |  |
|  |  |  |
| Josh Brown – jb740 |  |  |
| Contributed |  |  |
|  |  |  |
| Kurt Robinson – kr185 |  |  |
| Contributed |  |  |
|  |  |  |
| Peter Brown - pb864 |  |  |
| Contributed |  |  |

# Feasibility Report Works Cited

Unfuddle. (n.d.). *Unfuddle Signup.* Retrieved 03 26, 2014, from https://unfuddle.com/signup

acaudwell. (2013, 04 26). *Gource-Software Version Control Visualisation.* Retrieved 03 26, 2014, from https://code.google.com/p/gource/

Bugzilla.org. (2012, 11 19). Retrieved 03 26, 2014, from http://www.bugzilla.org/about/

Fossil. (2014, 03 15). *Fossil.* Retrieved 03 26, 2014, from https://www.fossil-scm.org/index.html/doc/tip/www/index.wiki