Working in the Cloud

Web-based Version Control System for Task-oriented Group and Individual Projects

Sheng Yu

September 2011

Report on a project submitted in part fulfilment for the degree of

Master of Science in Information Technology

Department of Computer Science

The University of York

Supervisor: **Dr. Stefano Pirandola**

Number of words = 16,850, as counted by the Microsoft Word's word count command, excluding appendices and words on figures.

Abstract

Version control is one of the most common ways to manage computer based projects. However, due to the software based design and file-oriented mechanism, the existing version control systems are too large and too complicated for lightweight use, as well as being difficult to deploy in environments such as a user working on a computer without an administrator account or using a mobile device.

In this project, a web-based version control system for task-oriented group and individual projects has been developed, as a supplement to existing version control systems, by following the standard process of web application design and development: requirements analysis, methodologies, design, implementation and evaluation.

Using the concept of cloud computing, the version control system which was designed and developed in this research project is a fully web-based system. It does not need installation of any software on the client side and can be accessed anywhere, including on a mobile device or a computer without administrator privileges, only a web browser and network connection are needed.

To effectively assign work to group members and avoid hassle in editing overlapping files, the system has been designed using a task-oriented algorithm to divide the work traditionally done by files into tasks for assignment.

Statement of Ethics

Ethics are defined as moral judgement and moral decision of questions about human behaviour. In computer science, it is related to the responsibility and accountability of people design and implements information systems [1].

In this project, as a functions set in the title, private project function may involves personal sensitive data of users. According to UK Data Protection Act 1998, Schedule 1 – The Data Protection Principles [2] and ACM Code of Ethics and Professional Conduct, article 1.7 – Respect the privacy of others [3], all the personal data stored within the system should be carefully protected to avoid leak to unauthorised people. In order to protect user privacy, each operation regards personal data would be force requested sign in before continue, people who have not membership or not been authorised will be rejected. Private project data will never use for other purposes without consent of the users.

According to article 1.1 of ACM Code of Ethics and Professional Conduct [3], the working of information system should contribute to society and human well-being. As this project is focuses on aiming computer based group and individual project, it is actually improving efficiency of people's work. The increasing efficiency will gradually improve the quality of life. There are no immediately threats to health and safety.

The article 2.7 of ACM Code of Ethics and Professional Conduct [3] advocated that public understanding of computing and its consequences should be improved. This project is design for doing version control for different kind of computer-based projects, including document writing, digital art and audio processing, etc. After time of using the system, I believe it will gradually affect the understanding of computer in the public.

Contents

Index of Figures

Index of Tables

1. Introduction

Version control is a well-known way of controlling revisions of works, especially programming and documentation writing. It keeps all the historical change made to a piece of work and gives users the chance to roll back their work at any time. It also can be used as a synchronised platform, so that the document can be worked on in many places and with collaborations from other people. However, as observed above, the number of people using version control in managing their work is relatively few, even among computer science students. It is asserted that the reason leading to this may be that existing version control systems have too many limitations, in particular, they must be pre-deployed before use. The concepts and use of existing version control systems is also complicated. Some experienced users of version control system may think the system can track files by different tasks. So I wish to design and develop an easy to use web-based version control system with task-oriented features for computer-based group and individual projects, as a supplement to existing version control systems.

This report describes the system developed in the research project. The project report contains the following chapters, Literature Review, Requirements Analysis, Methodologies, Design and Implementation, Evaluation and Conclusion.

2. Literature Review

This chapter discusses the literature related to this project, which develops an application as a web-based system for doing task-oriented version control for group and individual computer-based projects. The literature discussion is divided into the following parts, version control, existing systems, web and cloud, programming languages and interaction design. These parts are all about the features of the system.

2.1 Version control

Version control, also called revision control or source control, is a method of managing files related to development of projects through the whole life cycle [4]. It is essential for multi-developer projects [5]. Lots of popular version control software, including CVS and Subversion, run as client-server models, so they support more than one user working on (and especially programming) the same project, because the multi-clients can be connected to a central version control server to be synchronised [6] [7]. The general features provided by version control include storing each commit/version of a file or directory, allowing rollback, showing modification history and assisting merge/integration [4]. Based on the features, many kinds or parts of a project can use version control to be effective managed, for example, software development projects and documentation writing. Each commit/ modification can be marked with the properties of which author did the commit and the time it was committed. The commits also comes with a unique version code to identify times of modifications. Users can also add comments to the version when they commit it; so it is very easy to identify the modifications made by each of the users in the committed version. When a version commit has been identified as worse than an older version, the modifications can easily be rolled back to a previous version at any time [4].

Even though the existing version control systems sound good enough, there is still an important point which should be noticed. That is the “concurrent access” problem which exists in almost all version control systems [8]. All project developers can access an entire project, and have privileges to change any part of the project. It is hard to assign responsibility of tasks to the developers, therefore, the developers may forget where their positions in developing the projects are. When a developer mixes up his/her area of development, and changes some files which another developer is working on, it may lead to serious problems, or wasting time in combining works manually. Even though the merge

algorithm can combine most of the work, the results cannot be guaranteed to be the most expected end product, due to the complicacy of different kinds of working [8] [9] [10]. Some version control systems, such as Subversion, have designed a "lock" feature to prevent this problem. However, even though a file has been locked by a developer, other developers can get a copy of it from their local update before the file was locked or from an archive of older versions. It is hard to restrict this problem under current version control systems [9].

To solve this problem, this project aimed to develop a version control system which has a "task-oriented" feature, which avoids developers forgetting their role in the development. It also prevents occurrences of concurrent editing of files. Task-oriented development supports more than one person working with one project on separate tasks, without interrupting or overlapping. Assignment of tasks for the developers can be confirmed by a group discussion of each developer's strong and weak points, and finish with the assigning of tasks by the group leader in the new system. To achieve task assignment, a schedulable feature may be useful. In the system in this project, tasks could be set up in relation to other tasks. In 1910, Gantt first published the concept of predecessors in his famous Gantt chart [11]. It resolved the organisation problem of group collaboration in scheduling very well. This project uses the concept from the Gantt chart to schedule tasks. A task can have a predecessor/father task, which means that in order to make sure tasks are worked on in sequence, a task can be started only when its predecessor task has already finished.

2.1.1 Existing version control systems

As coordination and work management are important features of the system, some designs can be learnt from the operating mechanisms of existing version control systems. In this section, the popular software-based client-server version control systems CVS [6] and Subversion [7] will be analysed by their important features, and associated with the idea inspired by them.

a. Unique version number

Every content modification of directory or file will generate a new version attached to the directory or file itself. Moving, renaming and deleting will also be considered as a change. Each historical version will be kept for in-case use [7].

b. Atomic commits

For coordination consideration, every submission of a modification set will generate a version of the modification history, even if only part of the set has been modified [7]. As the system which this project developed was task-based, it should be different from the existing SVN versioning mechanism (all files will be updated to the latest same version code after every submitting command). The task-based feature limits the modifications to each tasks; therefore, the versioning mechanism could be designed as, every time a file or directory is modified, it will increase the version number of the file and directory individually. The task version will be increased when a submit operation has been executed, no matter how many modifications of file and directory have been made, even only renaming a file. For example, when committing the modification of one file in a four file task, the version code of the one modified file will be increased by 1, the other unchanged files will still have the old version code. The version code of the entire task which contains this file will also be increased by 1, because the new mechanism is designed to able to track the modifications of each file associated with a whole task.

c. Locking

To avoid unsynchronised editing, lock-work-submit-unlock is a very good feature provided by SVN for conflict free editing [7]. In this project, the task as a minimum assignable unit, should be locked when a user starts doing a task. When a task is started by a user, a lock would be set to a task, none of the other users, apart from the user who set the lock can request any modification of the task, even if he/she was already assigned to the task.

2.2 Software-based, web-based and cloud

Lots of popular version control systems are traditional client-server model based [12]. Even though some version control systems were built using the distributed approach, like Git[1], it is very complicated for small group and individual projects to use, due to it being hard to understand and use for non-experienced users [13]. To set up a traditional version control system, requires setting up three parts, which will be configured to work together: a centralised server to run the server side software, a client(s) to run the client side software and reliable network connections between server and client(s). The server stores all

[1] Git is a distributed version control system developed by Torvalds, the father of Linux. It was used to manage the development of the Linux kernel originally, and is now used for some large scale projects [69] [70].

versions/commits of each file, the client(s) stores a copy of the latest version of project files and allows users to work on and change it for future commit [4] [14] [15]. Network connection is a very important component in a version control system, because it connects server and client(s) so that both of them can be synchronised to the latest status and keep files up-to-date.

In the normal way of doing a project in a version control system, using client side software users, as clients, usually sign in to the repository of the project on the version control server, download (update) the latest version of the copy of all files related to the project from the server to their own computer and then work on the files.

The users may have arranged in advance to avoid possible overlap work on the same files, at least in the same class of a programming project or the same paragraph of a documentation writing project, because the existing merge algorithm in most version control systems could not combine overlapped work in different commits without conflicts.

When a user completes a milestone in his/her files, he/she needs to re-log in via client software into the project repository in the server and commit all the work he/she did to the server. After committing, a new version code will be generated. When other users as clients try to download or update the project, if a newer version of files on the server has been detected compared to the local version, the files on the client side will be updated to the latest version [4] [16].

Such helpful features make version control systems work great for managing many kinds of project. However, they need to install software both server side and client side to perform the version control actions [4] [16]. For entry-level users, it may be hard to install and configure client side software to work with the server which is providing the version control service. Furthermore, users who often work in different kinds of environments, instead of at their own computer, such as working in computer labs or on mobile devices like iPad, may not have not privileges to install the client side software of the version control system, or the mobile device may not support version control software.

To allow the client side user to be able to work in most environments with version control support, it is important to find a way to deliver the service without the need of installing software. With the inspiration of more and more popular cloud computing concepts like Google's Cloud [17], this project considered developing a fully web-based version control system, which does not require any installation of client side software. Web-based design

can overcome some of the drawbacks of software-based design, such as being hard to use in certain places and being hard to configure by an entry-level user. It can be easily accessed with any computer, even a mobile phone. The user just needs a browser and network access to the version control server. When transferring from an existing version control system to a web-based system, the user can not only benefit from the "access everywhere" feature, but also gain from another important feature, "easy upgrade at cloud with less disruption" [17]. If the system needs to upgrade to the latest release, it does not need to ask the user to do an upgrade of the client side software like in the traditional way. To upgrade the web-based system, it just needs to change the server side software and all users can start using the new system as usual via their web browsers [17].

The concepts of "network", "cloud" and "web-based" indicate that the system is Internet-based. However, it can also run on a local area network as a "local cloud", because the Ethernet supports the same technology as the Internet, such as IP based TCP connections and HTTP protocols, which allow web-based systems to run on local area networks similar to running on the Internet [18]. To set up the web-based system in a company-wide network, it needs to allocate a computer as the server, configure its Apache, PHP and MySQL running environment and install the web-based version control systems. After installation, users can directly access the server's domain name[2] or IP address, even a private IP address[3], on any browsers on devices connected to the same local area network as the server. Due to the centralised structure and network connection dependency of existing version control systems, the cost of migration from existing systems to the new web-based system might be very low, as only server side configurations are necessary with the new web-based system. Users are asked to access the new system via their browsers, even mobile browsers and they can continue with their work.

2.3 Programming languages

In order to develop the system to be used on the web, there are several programming languages available, such as Java/JSP[4], ASP[5], ASP.net[6], CGI[7] and PHP. In these languages, Java

[2] Domain name, an easy to remember name which identifies a computer, which has a mapping relation (records) to IP addresses, allows users to access various services on a server by only using its domain name, such as http and ftp (A record), mail service (mx record), etc. [75] [76].
[3] Private IP address, is an IP address in a range of pre-reserved network address space. It is usually used in local area networks, which can only be accessed by a computer on the same network [71] [72].
[4] JSP, JavaServer Pages, a technology using Java which creates dynamic web content [68].

as a popular programming language in object-oriented software development can also provide web service by working with JSP on Servlet[8] [19]. However, even though it provides the most object-oriented structure for programming, it is hard to set up a server side environment and also hard to programme on the scale of this kind of project. Furthermore, its structure is too complicated for lightweight development, because of its full object-oriented design and the great number of components requirement for running [20]. ASP is Microsoft's outdated web application engine, which was very popular in the era of Microsoft Windows NT 4.0 and Windows 2000, with many downsides such as high cost, slow speed, lack of library support, low safety design, no debugging support and hard to do migration to other platforms [21]; however, ASP.net is the latest Microsoft web application engine, which overcomes many of the drawbacks which existed in ASP. Unlike the previous version, it is still not an open source platform, so it still hard to do migration in the future, and there is a high cost in setting up [21]. Finally, PHP, is an outstanding web programming language and platform with lots of impressive features, such as fully open-source, object-oriented support, abundant built-in library functions and rich high quality open source resources. It is totally free [22], but still has excellent features with a commercial programming language and platforms. The PHP running environment is also easy to set up as a number of pre-configured server kits are available to do one-click installation of the PHP running environment [23] [24] [25]. It can run with open source web server software such as Apache[9] and Nginx[10], also Microsoft's IIS[11] series. There is some opposition to PHP, because it is considered too simple and can only be used in developing lightweight applications; however, Facebook as one of the largest websites, uses PHP as its main programming language and it has also made great contributions to making PHP better [26]. Overall, PHP has been chosen as the programming language for developing this project.

---

[5] ASP, Active Server Pages, is a server-side script engine from Microsoft for dynamically generating web pages, using VBScript and JavaScript as server-side programming languages [80].
[6] ASP.net, is Microsoft's second generation server-side script engine. It uses Microsoft's .Net Framework as libraries and also uses object-oriented programming languages such as VB.net and C# [78] [79].
[7] CGI, Common Gateway Interface, is a platform-free interface for the client to execute applications on a web server [81].
[8] Servlet, is a server-side Java application which generates web pages as an interlayer between client request and server response, with platform-free and protocol-free features [19].
[9] Apache, the common name of Apache HTTP Server, is an open-source web server application which has been used widely in the world, and can be used on lots of operating systems [82].
[10] Nginx (Engine X) is an emerging high performance open source HTTP and proxy server [83].
[11] IIS, Internet Information Services, is Microsoft's Windows-based internet server application [84].

As a great partner of PHP in LAMP[12] group, MySQL[13] has been chosen as the database system for development of this project, because it provides a great number of features in a small size installation with easy configuration. MySQL is open-source and free. It also has lots of useful features, such as "view" and "lock", it is a good choice for different kinds of use and easy to customize [27].

2.4 Interaction design

Interaction design is "the practice of designing interactive digital products, environments, systems, and services." [28] It defines and designs the behaviours of human in perceiving and using digital objects [29]. All research in the area of interaction design focuses on its objective: to make a product easy to use and technology which people are happy to interact with. In order to achieve the objectives, there are several things which need to be finished in sequence: the expectation of different kinds of user should be classified for analysis; understanding what user behaviours would do or have done during their use of an interface; understanding the psychological characteristics of using the interface. The relationship between product and user is built by the interaction design of the interface, to help user to do what he/she needs via an effective and stress-free interface [29]. In measuring the usability of a product and its interaction design result, Nielsen introduces a framework of system acceptability [30]:

Learnability	How easy is it for users to accomplish basic tasks the first time they encounter the design?
Efficiency	Once users have learned the design, how quickly can they perform tasks?
Memorability	When users return to the design after a period of not using it, how easily can they are establish proficiency?
Errors	How many errors do users make, how severe are these errors, and how easily can they recover from the errors?
Satisfaction	How pleasant is it to use the design?

Table 1 - Nielsen's framework of system acceptability [30]

[12] LAMP, a powerful bundle of open-source software working together as a web server, includes Linux (operating system), Apache (HTTP server), MySQL (database) and PHP (script language) [85].
[13] MySQL, an open-source database system, developed by MySQL AB, now is a part of Oracle [86].

The five points from Nielsen describe the measuring standards of usability. Product designers and especially interface designers should make sure their work meets the five points, to enable a good user experience.

As the aim of the web application developed in this project is to provide an interface for the user to be able to easily find functions and feel relaxed when using it, the interaction design is an important part of this project. Additionally, it is important because users may have little experience in using the system. Users may not be clear how the logic works in the system, however, this project should make sure they feel that they can use the system easily with few barriers.

3. Requirements Analysis

The requirement analysis section investigates areas concerning the requirements to be considered in the design and implementation of this project.

3.1 Project goal

As described above, the goal of this project is to design and develop a version control system with task-oriented feature for groups and individuals to use it in their computer-based projects via a web interface. Compared with existing version control systems, the most two important "different" features of the research project system are that it is web-based and task-oriented.

3.1.1 Web-based

The first feature, being web-based, is influenced by more and more popular cloud computing concepts. In this era which is becoming full of cloud computing, the computer is becoming a tool for using cloud services. The wide use of the internet provides many favourable factors for people's collaboration as a group [31]. For example, in the Google cloud computing design, the user only needs a device with a web browser to use all the Google services, no matter what device they have or where they are [17].

Google Apps is an interesting product which should be noted. It is a fully web-based enterprise office solution, which contains enough tools for companies to work with it. It is also more flexible than using local software. The user does not need to install any programmes or waste time in considering software compatibility between different hardware and software combinations. To run an online web-based programme, users just need a mainstream web browser, even on a tablet or mobile phone, without requirement of hardware or software configuration [17]. In Google Apps, users can work together without sending the work anywhere. They only need to create or upload previous work into Google Docs, and ask the group members to sign in with their Google accounts. Once this is done, all the group members can see the entire document and they can modify it, add new things or comment on paragraphs etc. When a user makes a change to the work, the old version is archived as "version control", in case there is a need to roll back to an old version, if the modification is not wanted. Users can work together at their own computers by signing in to

the same system without need of sending their work to each other, as well as preventing the confusion of different versions of a document [32].

Another product which should be noted is Google Chrome OS. It is the result of a pure web-based cloud computing concept. Chrome OS only has a web browser, and applications running in the OS are all web-based. A netbook loaded with Chrome OS does not need to have powerful processing capacity. The power for basic processing of web content is enough, because the core of the web-based cloud applications is running on the cloud side (or server side), the user side or client side computer in just in the role of helping the user to interact with the user interface of the application [33]. Based on the plentiful advantages of cloud computing, as a trend towards moving from traditional software-based applications to web-based applications, this project is aiming to design and develop an easy to use version control system, providing a fully web-based interface for users which can be used anywhere.

3.1.2 Task-oriented

In the existing version control systems, flies are base units to be version controlled by project. When commit modifications, all files modified in once would be counted as one version of a project [6]. In the common group working for a computer based project, they may have several members working together. In most cases of computer based project, members in a group are working their works paralleled, and combined/merged at the end when finished working. During the working, they are communicating with each other, and then continue doing their work in files. To improve efficiency of a computer based group work, a better way is to clearly define and design tasks within a project and assign them to group members by analyses points their good at or not, because distinct task assignment can maximum uses group psychosocial traits, as well as give pressure for member for motivation by compare progress and quality of tasks [34]. In the existing file based version control mechanism, group members working in a same project may usually confuse in identify duty of files [10]. To make the “task” can be a part of the version control system, in this project, the system was focus on build a task based mechanism in order to tracking two level of versions in a project: task level versions, directory and file level versions, for tracking directory and file within task assignment.

3.2 User needs

The user is one of the most important things on which this project should be focused. As defined in the project title, this system is for developing computer-based group and personal projects with version control. It is observed that people who do documentation writing or programming may focus on version control in aiming to keep work safe or make collaboration easier.

3.2.1 Deployment and Portability

Some users may be used to working in a computer lab or on public computers and therefore do not have administrator privileges, therefore, a big problem is that the user does not have any right to install software. If the user is relying on using a version control system to help with work, they may become frustrated at the lack of version control client software available for use.

In order to help this type of user be able to use a version control system, the deployment of the system especially on the client side should not require administrator privileges. At the same time, if the client side software is not supported to run on different operating systems, it may limit the usage of the system, so it should support different kinds of operating systems.

For people who like to use a version control system on a mobile device, such as a netbook, tablet or mobile phone, they may wish to get a similar experience or familiar interface as when they use it on a regular desktop PC. If the user can access the system on any platform with the same or similar “look and feel”, the system would score better on Nielsen’s framework of system acceptability [30]. This means the system would have better learnability for users when they first use the new platform and better efficiency once they start to perform tasks more quickly. Therefore, the system should have a universal interface design for different platforms.

Even though the system is install-free on the client side, it still needs to be set up before use on the server side. To make sure the system can be installed on different kinds of servers, the server needs to have a minimum of PHP and MySQL installed. The running parameters will not be integrated into the programme. A better solution may be to set up an external configuration file to place all parameters, such as database connection information, system

name, time-zone, administrator e-mail address, etc. Then, when the programme needs to know the value of the parameters, the configuration file can be loaded at the beginning to be initialized.

3.2.2 Accessibility

In order to serve as many people as possible, it is very important to also give accessibility to people who use screen reading software. The W3C has defined standards for better accessibility, the "Web Content Accessibility Guidelines" (WCAG). For example, as WCAG 1.0 Guideline 1 states, all the images on the web page should have an alternative text label for the screen reader to find and read; in Guideline 4, natural language usage should be clarified in the HTML head; in Guideline 5, the table header should be defined, in order to identify the property of the columns [35]. By following the guidance, blind users, amblyopia users and people in dark environments or any people for whom it is not convenient to understand written pages on screen could benefit from special consideration and be able to obtain what they need on the web pages much more easily [36]. As this project is aim to provide good accessibility for user, it is necessary to design by following the guidance of WCAG document carefully.

3.2.3 Easy to use

University of York computer science students were observed doing group or individual coursework. However, they were seldom interested in using a version control system, even some of the Software Engineering students. The most common reason for this was: "it is too complicated to configure the coursework to be version controlled" (by summarise results via chatting with master students about version control in labs during students' rest time in group work).

In setting up a traditional version control system to be worked on the client side, an account needs to be registered first. After registration, the user needs to apply for a repository/project to be opened on the client side of the version control system. Then the user downloads the client side software for the version control system, logs in with an administrator account and installs it. When the installation is finished, the user needs to configure a file folder as a project, with the username, password, server information and

repository address to "check out"[14] on the version control server. If all the above steps are completed without any errors, the user can do the first commit of the files and start using the version control system. If the user decides to move the work to another platform, they will need to redo everything, in order to re-configure the client side settings [4].

This project aims to design an easy to use version control system. To make the system easy to use in a small project like coursework given to students in the University of York Computer Science department, it should focus on core functions and make the process of using the core functions as simple as possible. As analysed above, a web-based interface would be a simple way to provide core functions with an easy to use interface. The user may use the system in the same way they use the webmail system; they can sign in and tick the "remember me" label to perform an automatic log in, in the future. They can then create a project with a project description and generate tasks to allow the work to be better scheduled. Finally, the user can easily upload files into tasks, as easily as adding an attachment when composing an e-mail. A new version of the files will then be created. This satisfies the easy to use requirement.

3.2.4 Privacy and safety of data

When working on a project on a version control system, the group members or the private project owners may not wish other people to see what they are doing. They may prefer not to use a version control system, due to fears about leaking their work to unauthorized people. Users usually have privacy considerations when placing sensitive data on a shared media, even though the media provider promises that the data will be well protected [37]. To reduce concerns, the system needs to be developed with a well-designed mechanism to ensure the safety of all data. None of the data should be obtained by people who do not have permission.

3.2.5 Response time

In the user experience evaluation, the page response time was an important factor in influencing service quality [38]. Regarding response times, Miller [39] defines one second as the maximum cap for a user in feeling that they are freely navigating between pages. If the

[14] Check out is a subversion (SVN) command, which makes the server side repository associated with a local file folder, as a version controlled project [4].

response time in switching pages is longer than one second, the user may feel that they are waiting and the user experience would be reduced.

To provide the best user experience with a fast response time, less time should be taken for database queries, as they often take much longer than PHP internal processing. Large content size, such as images within pages, should also be avoided as much as possible. The consideration of page response time should include any delays due to network connection speed.

3.2.6 Platform compatibility

Compatibility of a website is about the similarity in the expression of web pages when they are visited via different browsers and platforms [40]. In the past few years, during the development of the Internet, Microsoft Internet Explorer, as a built-in web browser in Microsoft Windows, has been the most widely used web browser [41]. In the early years of wide-spread web use (pre-2005) Internet Explorer occupied the largest market share of web browser use and web designers only needed to consider how pages would display on that browser, because users who used other browsers were in the minority, with only a fraction of the market share. However, after 2005, with the development of Mozilla Firefox, the market share of Internet Explorer browsers is reducing day-by-day [41]. More and more people started using operating systems other than Microsoft Windows, such as Linux, which Internet Explorer does not support. At the same time, Windows users also started using other web browsers, such as Mozilla Firefox, Google Chrome, Apple Safari and Opera because these browsers had better compatibility with W3C standards. In Browser Statistics [41], the market share of both Mozilla Firefox and Google Chrome since April 2011 is already beyond the share of Internet Explorer. A good way to provide a similar experience in various browsers is to make sure the website is W3C compliant.

Data Browser Display Statistics [42] shows there are still a number of users using a screen resolution of 1024x768 pixels. To avoid those users having to drag pages horizontally when browsing, the page must be no wider than their maximized browser windows. This would make the 1024x768 screen users feel much more at ease when using the system.

JavaScript is a widely supported scripting language, used in processing client side user behaviours locally. Web designers started relying on JavaScript in more and more operations, even though they should not have just used JavaScript for input validation in place of using

server side validations [43]. They may have considered that the use of JavaScript could provide a better user experience. However, if the user's web browser does not support JavaScript, it may cause harm to the system. To avoid harm from non-JavaScript-supported browsers, some key validations should be designed on the server side.

3.2.7 Reporting

When using the system, users may find a problem with the tasks, project or even the system itself. To make sure that any problems can be easily communicated to staff or other group members in the same project, an obvious indication of who to contact should be designed into the pages. It must be ensured that users can easily find how to communicate, and the way of communication needs to be two-ways.

If a user makes an illegal operation which the system cannot deal with, an error would be generated. When an error occurs, the system must not crash. The reason for the error should be displayed on the screen for the user to see. If the error is caused by the system itself, the system administrator should be sent an e-mail notifying them of the error.

4. Methodologies

Methodologies of this project were related to the models and methods used in design and development. As the project was based on the concept of cloud, software development and web design, it should be designed and implemented according to the methodologies of software as a service (SaaS) in cloud computing, waterfall model in software engineering, prototype design in interaction design and entity-relationship modelling in database design

4.1 Software as a service (SaaS)

Software as a service (SaaS), also known as on-demand software, is one of the three[15] well-known service models of cloud computing [44], which delivers software applications by user needs via Internet [45]. The growth and public acceptance increase of SaaS applications give great experience to the designer, and the most important characteristic made SaaS solutions successful should be the use of web-based interface [46].

In contrast to traditional software, the SaaS applications have less platform dependence and resource usage. For ordinary users, thin client with browser and Internet access is enough to support the running of cloud applications. At the same time, by the popularity of the high-speed Internet connection and the more standardized web technologies such as HTML, CSS, JavaScript and HTTP reduced the cost of developing SaaS solutions, more competitive applications were being developed and promoted.

Some people may have concern of the security of data, because of all the data was stored in the centralised cloud, however, the cloud service provider usually have more experience in protecting personal data than individuals [47].

SaaS solution advantages have been summarised as low cost, fast deployment, easy access, safe, etc. [48] As this project is aim to build a SaaS application, it should have features of [48]:

- Web-based interface;
- Well data protection;
- Less software dependence;

[15] The other two models are Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) [44].

- Anywhere access;
- Fast deployment;
- Low cost

4.2 Waterfall model

Waterfall model is a popular software development processes, which commonly have stages of conception, initiation, analysis, design, implementation, evaluation and maintenance, first formal described by Royce in 1970 [49]. The stages above made sure software can be designed and developed with good quality and well scheduled, however, it may not adapt to all software development projects. In reality, the stages can always hardly be linear executed. Some stages such as conception, initiation and analysis have been disputed as too cumbersome. To make sure the waterfall model efficient and effective, some stages have been combined. As shown in Figure 1, the most acceptable version of revised waterfall model now have five stages, including requirement analysis, design, implementation, evaluation and testing, maintenance [50]. It should be noted that the waterfall model is now a standard in software development [51].

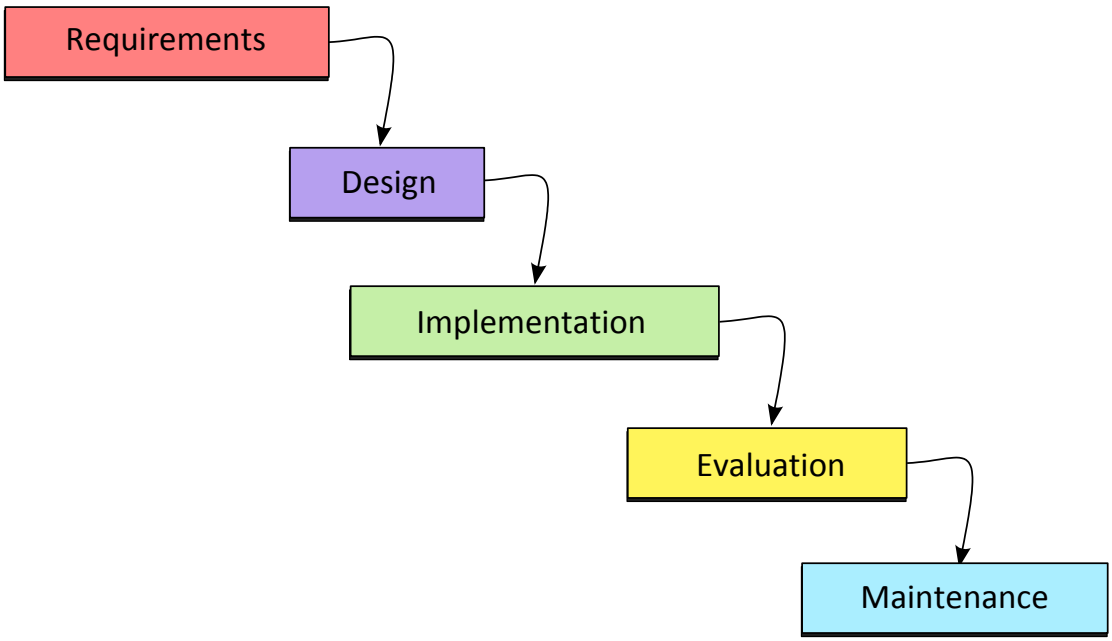

Figure 1 - Typical five stages revised waterfall model [52]

4.3 Prototyping

Prototyping is the process of creating early sample by an idea or a design [53]. In information system, prototype helps system designer build an intuitive model for doing analysis and basic evaluation. For design of information system, low fidelity[16] prototyping is great and simple step in displaying idea from designer. After first version of prototype finished, it can be used at simple evaluation for problem finding. Further version of prototype re-design may require for making the design better. During the progress of repeating prototyping and evaluation, more and more problems behind design would be continuous discovered and solved, especially user experience related problems.

4.4 Entity-relationship modelling

Entity-relationship modelling is method in describing data, structure and relationship in relational database [54]. The conceptual design of database is usually based on the entity-relationship modelling, and is it a very important step within the design stage of software development. A strict (good) E-R model is usually [55]:

- Has many entities (tables[17])
- In third normal form
- Connected by primary key and foreign key
- One to many relationships

4.4.1 Third normal form (3NF)

The third normal form, often abbreviated as 3NF, is a normal form in database normalization, which defined by Edgar F. Codd in 1971 [56] [57]. All database tables in 3NF should meet the conditions: table is already in 2NF[18] and its attributes except primary key (ID) should not dependent on other attribute [56]. In 3NF, database tables could minimal their redundancy

[16] Low fidelity prototype, usually be written as low-fi prototype and be called as paper-based prototype, is basic and the most low-cost type of prototype, often be used in initial stage of design. It can be performed by several software, or just using pen and paper [91].
[17] Entity is a more academic name, and it usually be called as “table” as an easy to understand name.
[18] 2NF, the second normal form, states all the records in database table should be unique identified [57]. The common way of making table meets requirement of 2NF is adding a unique ID to each record as its primary key [87].

and make the SQL statements could be connected together for multi-table operation much easier [58]. This means if a table in database meets requirement of 2NF, but not meets 3NF, it need to split into several tables. Redundancy attribute, which is the attributes dependent on other attribute, must to move out to a new table to keep avoid mistake/exception in operations (insert, update and delete) may lead by redundancy, also help tables can be isolated and uses SQL statements to be connected by their primary key and foreign key [59].

4.4.2 Entity-relationship diagram

Entity-relationship diagram (ERD) is the chart shows entities of attributes in database and their relationships [60]. It looks like the class diagram of UML[19] in software engineering. In ERD, tables should be connected by primary keys and foreign keys referral. To identify the "one to one", "one to many", "many to many"[20] relationship between tables, some notations has been defined for distinguish them.

It should be noted that there are four syntax of data modelling notations [55]:

- Information Engineering
- Barker Notation
- IDEF1X
- UML

[19] UML, Unified Modelling Language, is the standardised modelling language in object-oriented software engineering for its structure, active, process, etc. It can be demonstrated by number kind of diagrams, such as use case diagram, class diagram, sequence diagram and active diagram [92].
[20] "Many to many" relationship is not yet supported in relational database [93].

5. Design and implementation

This section gives the details of the design and how it will be implemented into the programming.

5.1 Work flow

When using the system, all users including group leaders and group members, as well as private project holders, should follow a work flow to use the system in controlling versions of the works. As the system has a task-oriented design, working units in the system will be divided into project, task, directory and file. File belongs to directory; directory is under the control of the task; the task is managed by the group member within the project, the project is monitored by the group leader. Each person registered in the system as a user can work on both group projects and private projects. In a project group, the group leader is also a member of the group. If the system has several project groups, a user can work for more than one group at the same time.

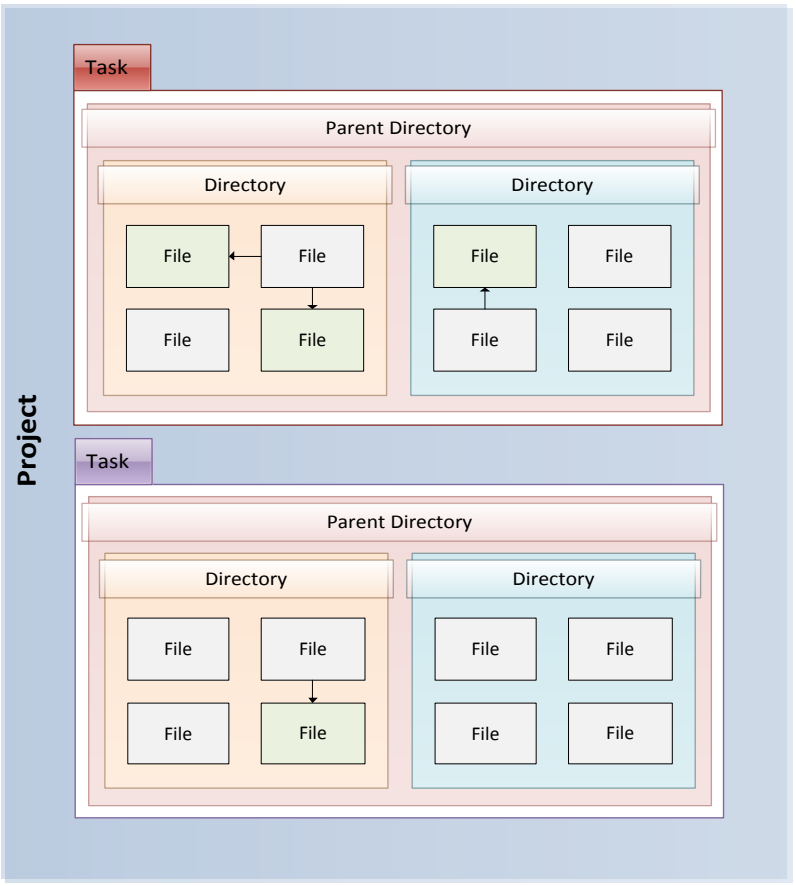

Figure 2 - System logic structure (project - task - directory - file)

The users in the system are being advised in a boundary-less organisations, which means the structure in organisations is neither flat nor tall, and can be grouped in any relationship when needed. For example, a group leader in project A can also be a group member in project B at the same time. This may break down barriers in collaboration [61]

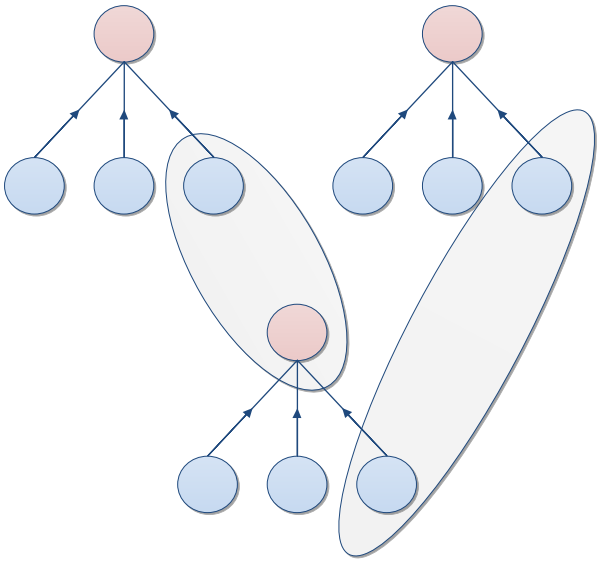

Figure 3 - Schematic diagram of boundary-less organisations

By analysing the relationships between file, directory, task and project, as well as the users, which are group leader, group member and private holder, a general group project work flow chart has been drawn in Figure 4.

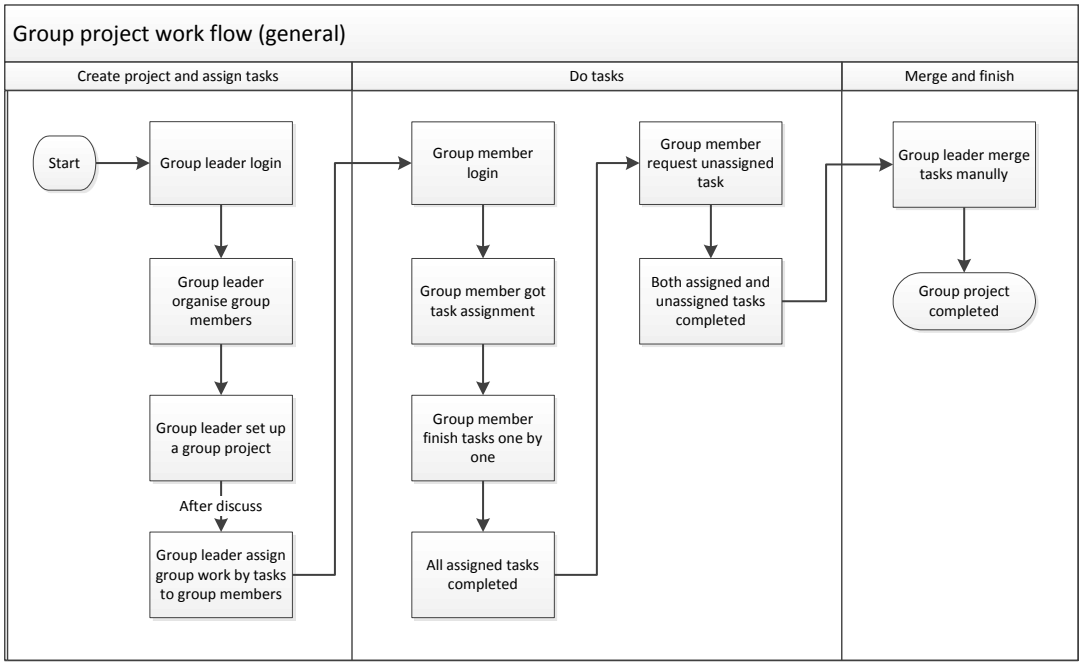

Figure 4 - Group project work flow (general)

Figure 4 shows the sequence and general steps in doing a group project. There are three main stage of working in a project.

The first stage is about creating the project and assigning tasks. In this stage, the project and its tasks should be created and assigned to group members by the group leader, after discussions.

The second stage is about the group members doing tasks. In this stage, after being given assignments in the previous stage, group members sign into the system using their own accounts, start doing the assigned tasks, finish the tasks and upload the document(s). If all a user's assigned tasks have been completed, the user can request to do any remaining unassigned tasks (if they exist). After every task is completed, the group leader merges all of them. After which the task will be finished.

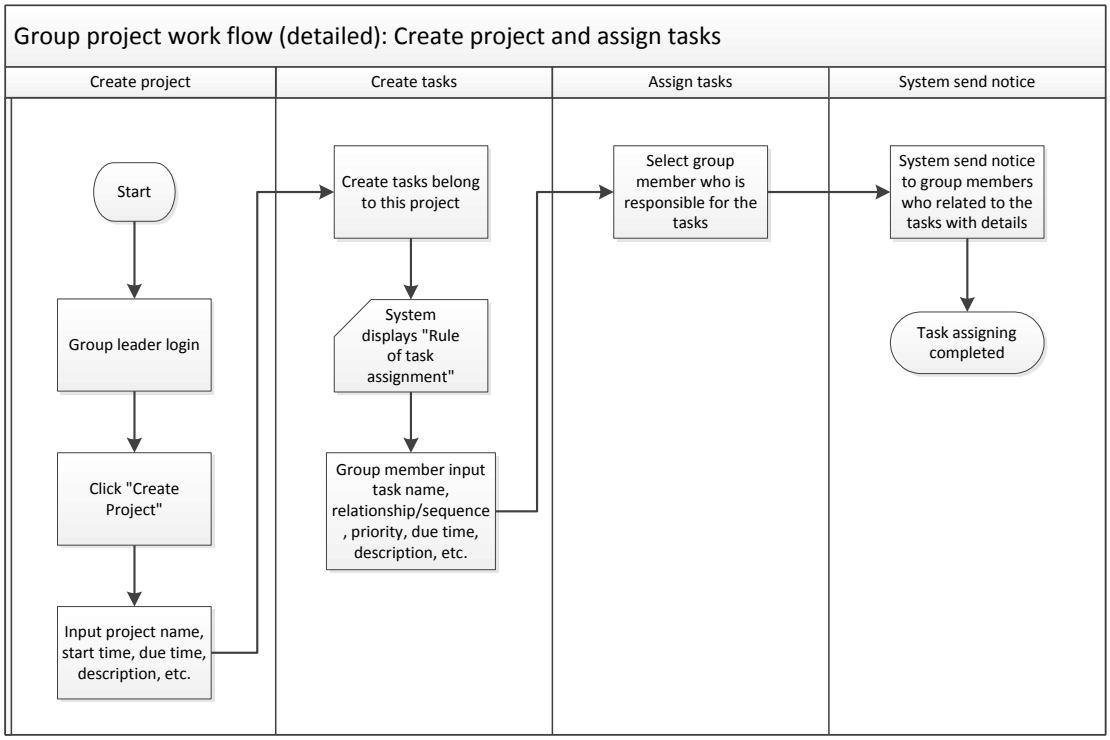

Figure 5 - Group project work flow: Create project and assign tasks

Figure 5 describes the details of the first stage of Figure 4. When starting a group project, everyone in the group should meet to discuss each member's different abilities in the project. The member who has the most obvious leadership qualities should act as the group leader.

After deciding on the group leader, the group leader should sign into the system, create a group project and input the details of the project, such as project name, start time, end time and description.

When the project creation is finished, the user who did this will be treat as group leader automatically; the group leader can then continue creating tasks belonging to the project. In creating tasks, the group leader should give the details of each task, like task name, start time, due time and description. A task also has the properties of any predecessor task, which can be set when creating the task.

After they have finished creating the task, the group leader can assign the task to a group member. After they have finished assigning tasks to all group members, any unassigned tasks can be left temporarily to await group members who finish their assigned task quickly and then request any unassigned tasks. If the task assignment request is confirmed, an e-mail with task information will sent to the group member.

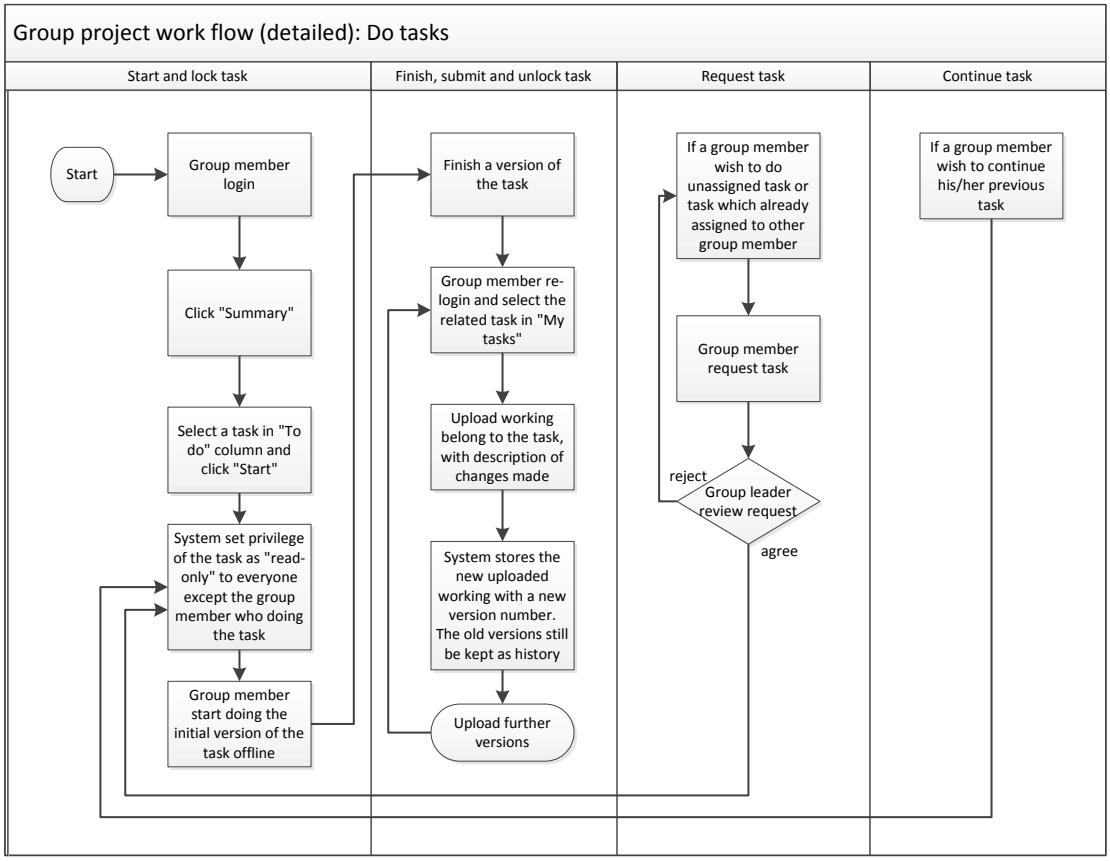

Figure 6 - Group project work flow: Do tasks

Figure 6 is related to the *Do Task* stage of Figure 4. In this stage, group members log in to the system and are redirected to the summary page. On the summary page, they will see

columns, including tasks which are being done, tasks to do, tasks waiting and tasks finished. If there is no task in a column, the column will be hidden to save space on the summary page.

When a user signs in, all the user's tasks which have not yet begun will be placed in the "tasks to do" column. When the user clicks on one of the tasks, a page for starting the task will be displayed. The user can create directories for the task as a structure of the file storage. The user can then start creating the initial version of the files within the task offline.

Once a file is finished, or once a milestone is reached, the user can upload files to the task at any time. When uploading a file, the user will be requested to input a description about the uploading/changing of the file. They also need to describe the task version commit, because each time the file changes will generate both a new version of the file and the task by the tracking consideration. After submitting the changes, the user can download his/her changes at any time for further work. If he/she re-uploads the file with further changes, again, both a new version of the file and its task will be generated. The older version will still be kept in the system as a historical version for "in case" use.

In a group project, there may be some unassigned tasks for group member who have already finished their work and still have time left to do more work. If a group member wishes to take on an unassigned task, they can click next to any unassigned tasks to send a request to the group leader. When the group leader receives the request, he/she can approve the request and assign the task to the user who wishes to do it.

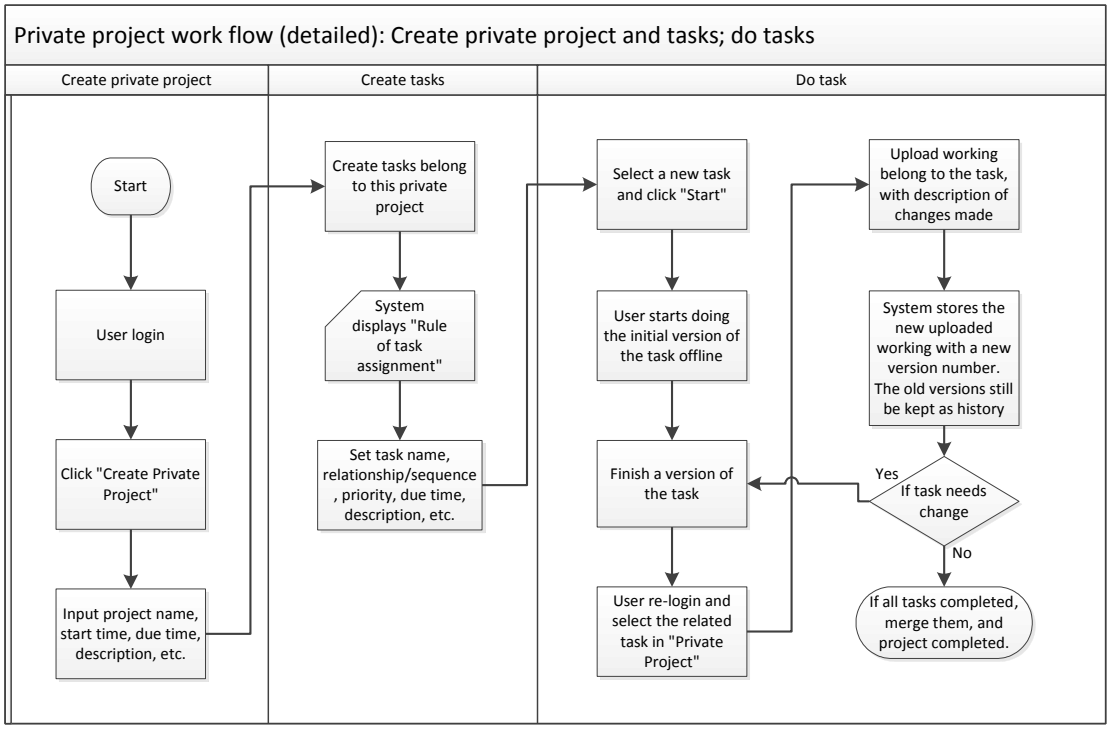

Figure 7 - Private project work flow: Create private project and tasks; do tasks.

Figure 7, shows that the system can also do version control for private projects (as defined in the project title). All members in the system can create a private project. The process of creating a private project is very similar to a group project. The system assumes that the group leader of the private project and the only member of the project is the private project owner. There is a tick box option for private project when creating a project. If the box has been ticked, the project will be set to private mode and no one other than the project holder has the rights to access anything related to the project.

5.2 Function and mechanism

In this section, the function and mechanism of the requirements of user needs will be designed and implemented, including file version control, task-oriented design, task relationship, directory version, directory relationship, file storage, error handling, log in, safety and performance optimisation.

5.2.1 File version control

Like existing version control systems, the new system should record a list of the history/previous versions of user files when a modification is submitted, including some special change such as add, rename or move a file. Each commit of a change will be stored in

the system as a new copy of the file. The older version of it will still be kept in the system as a historical version, without being changed, in case a change has been made by mistake. The user can then return to an older version at any time. For the user's personal information, he/she can use the system to make a backup of each step of their file work, in case of a mistake in editing, deleting or losing files. He/she also can track the modification history of a file in the system to do statistics after finishing the work.

For users who enjoy working on different working environments, the system can be used as a platform to transfer the latest version of a file to a computer anywhere. The user can commit a file modification for a new version, and after that, they can download the new version, no matter where they are.

To clearly identify each modification time, the user needs to input some descriptions to the file each time it is committed. At the same time, the file size and the commit time will be recorded in the file database. The type of operation will also be identified and stored as an operation code (int type) in the database table, in order to reduce database storage and avoid unsynchronised name being confused in the future.

Code	Operation
0	No change
1	Creating
2	Deleting
3	Content updating
4	Location moving

Table 2 - Operation code

Each time a file is committed, it will generate a new version number of the file. The first time the file is created, the version number will be "1", after any kind of modification, the version number will immediately be increased by 1, to equal to "2" etc.

In programming, several PHP functions from libraries defined by the researcher were used for doing version control (see Appendix D, especially "project.lib.php" for these functions). This included fetching the project information, task, directory and file from the database; receiving the uploaded file; comparing it to older versions; classifying and converting file information; updating related database records; and, storing the file on the server side physical hard drive .

5.2.2 Task-oriented design

The existing version control system treats files as base version control units; however, this project designed the system as a "task-oriented" version control system. The definition of "task" within the system is a set of directories and files in a project, which means a project

has several tasks, and a task contains several directories and files belonging to the directories. In group work, a task is the base unit of assignment to group members. If a group member has been assigned a task, files from the task can only be edited by this group member. Within the task, the member can create new files, rename files and change files and directory structure. For other members who have not been assigned to a task, this task is a read-only task, they can only download and view the files within the task, but cannot make any modifications to it.

Each task has a name which it is identified by within the project. A start time and due time can be set to help with the time management of scheduling tasks. The status of a task will be identified automatically. Statuses can be, “Waiting for predecessor task”, “Task expired”, “Not yet start” and “Finished xx%”.

To allow tasks to be sorted by importance level, tasks have a property recording priority, which can be set from 1 to 5. 1 means most important, and 5 means this task can be done after other tasks have finished. A user can change the priority of a task at any time, and it will be displayed in the table of tasks list, for sorting purpose.

The existing version control system treats files as base version control units; however, this project designed the system as a “task-oriented” version control system. The definition of “task” within the system is a set of directories and files in a project, which means a project has several tasks, and a task contains several directories and files belonging to the directories. In group work, a task is the base unit of assignment to group members. If a group member has been assigned a task, files from the task can only be edited by this group member. Within the task, the member can create new files, rename files and change files and directory structure. For other members who have not been assigned to a task, this task is a read-only task, they can only download and view the files within the task, but cannot make any modifications to it.

Each task has a name which it is identified by within the project. A start time and due time can be set to help with the time management of scheduling tasks. The status of a task will be identified automatically. Statuses can be, “Waiting for predecessor task”, “Task expired”, “Not yet start” and “Finished xx%”.

To allow tasks to be sorted by importance level, tasks have a property recording priority, which can be set from 1 to 5. 1 means most important, and 5 means this task can be done

after other tasks have finished. A user can change the priority of a task at any time, and it will be displayed in the table of tasks list, for sorting purpose.

5.2.3 Task relationship

Tasks can be set with a property of a "predecessor task", which means a task can have a parent task. If a task's predecessor task has been set, the task can only be started when the predecessor task has finished or expired. This is like the relationship between tasks in a Gantt chart, a predecessor task and its sub-task can be connected by end and head, to show the relationship. When setting a predecessor task for a sub-task, the system needs to judge using predefined conditions whether the relationship is valid. For example, is the due time of the predecessor task earlier than the sub-task's due time?

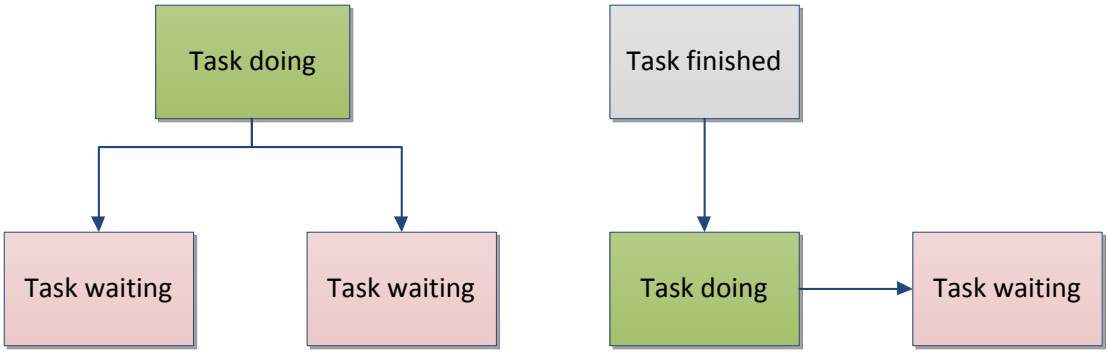

Figure 8 – Sample task relationship

In the implementation, the project.lib.php is also used for processing tasks, directories and files, as well as the relationships between them (see Appendix D). Functions in project.lib.php can deal with task predecessor set up and task status judgements using various conditions.

5.2.4 Directory version and relationship

In this project, directory will be treated as a unit in version controlling. Any operation related to a directory, including adding, renaming and moving, will trigger a version record. The modification version records of a directory are very similar to the version records of files, containing a description of the modification, type of operation and commit time.

A multi-level directory is not stored in a single database record. A series of single-level directory records compose a multi-level directory.

Each directory record in the database has an attribute which records the ID of its parent directory. If its parent directory attribute exists in the database, the directory will be connected to its parent directory as a two-level directory; if its parent directory also has a parent directory, they will be connected together as a three-level directory and so on. This continues until the parent level directory does not have a parent directory, and a full multi-level directory has been built. For example, the multi-level directory /A/B/C is composed of three single-level directories: A, B and C. A is the parent directory of B; B is the parent directory of C. A has no parent directory. Each of them is an independent version control unit. The parent directory attribute is used to join them together.

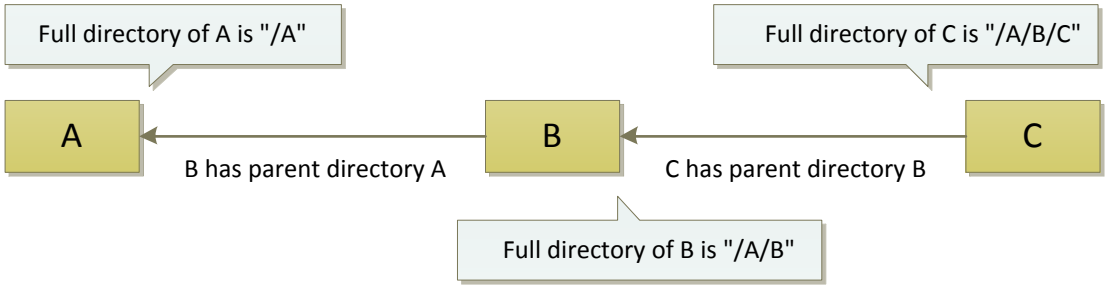

Figure 9 - Sample directory relationship

The file project.lib.php focuses on providing most functions related to files, directories, tasks and projects, therefore, the function which connects the parts of a multi-level directory is also placed in this file. Any PHP file which needs to call the functions in project.lib.php can simply “include” its file name with its location “style/”.

5.2.5 File storage

In this system, all file modifications will be issued with a unique “file change ID”. All the files uploaded into the system will be extracted by name and file size. The file name and file size will be stored in the database along with the file change record by its unique generated ID, then the content of file will be stored in the “files” directory of the system’s physical driver. In order to make files easy to store and load, the files stored in the physical storage on the server side will be named with the unique ID, without any extension. For example, if a file has been uploaded by a user via file uploading page, a unique ID (FCID, see Table 9) of the file version will be generated. If the FCID of this file change is 235, the file will be stored in the “files” folder as file name “235”.

In the implementation, the file_operation.lib.php in the libraries folder has some functions controlling file storage operations. The download.php in the root directory processes the download of files by request, the file name can be restored to its original name when the user uploads it, by combining the file contents on the physical disk and the real file name from the related database record.

5.2.6 Error handling

When an error happens, if the error is a user level error (maybe a mistake), the user will be provided with an alert message with the reason for the error, and the user can go back to the previous page to continue, after correcting it [GIVE AN EG]. If the error is at system level or caused by a bug, the user will be redirected to an error page with error information. At the same time, an e-mail including the error details will be sent to the e-mail address of the system administrator, which is pre-defined in the configuration file.

The error() function in file general.lib.php will deal with serious errors. It will be written in the place of code where may occur a serious error. [??] The error() function has a parameter for *error information*. If the function has been called with parameter information the page will be redirected by the function to an error page displaying the information, also an e-mail about the error will be sent to the administrator (in the background).

5.2.7 Login and Safety

To protect the user's work, the system is forced to require a user to sign in before use. A library file "identify.inc.php" is used for checking the login status of users. This file has been "included" in all PHP files, to be executed at their start, to make sure every time of operation is protected by the login identify mechanism of both Session and Cookie detection. Once an operation has been detected as not yet signed in, the page will be forced to redirect to the login page for the user to sign in before continuing the operation.

It is obviously a little tedious to be requested to login at each visit after closing and opening the browser. To make login easier for the "next time" use, a Cookie based automatic login mechanism has been designed and implemented, in the file "login_check.php". When a user ticks the "remember me" box on the login page, a Cookie containing login information will be stored in the client side browser cache area with an expire time pre-defined in the

configuration file. After the browser closes and the Session has been destroyed[21], if no Cookies with login information exist, the user will be automatic logged out; if the login Cookie exists, due to the "tick" on the previous sign in, the system will log in the user automatically, using the information in the Cookie, and the Session information will be restored to keep the user's status as logged in before closing the browser or manual sign out.

For operations related to private projects, the logged in user information will be carefully detected and compared to the owner information in the record of the related private project. If the user information matches, the operation can be executed as usual; if there is different user request operation from the private project, the operation will be rejected and an error message will be displayed. The functions which protect private projects are in project.lib.php.

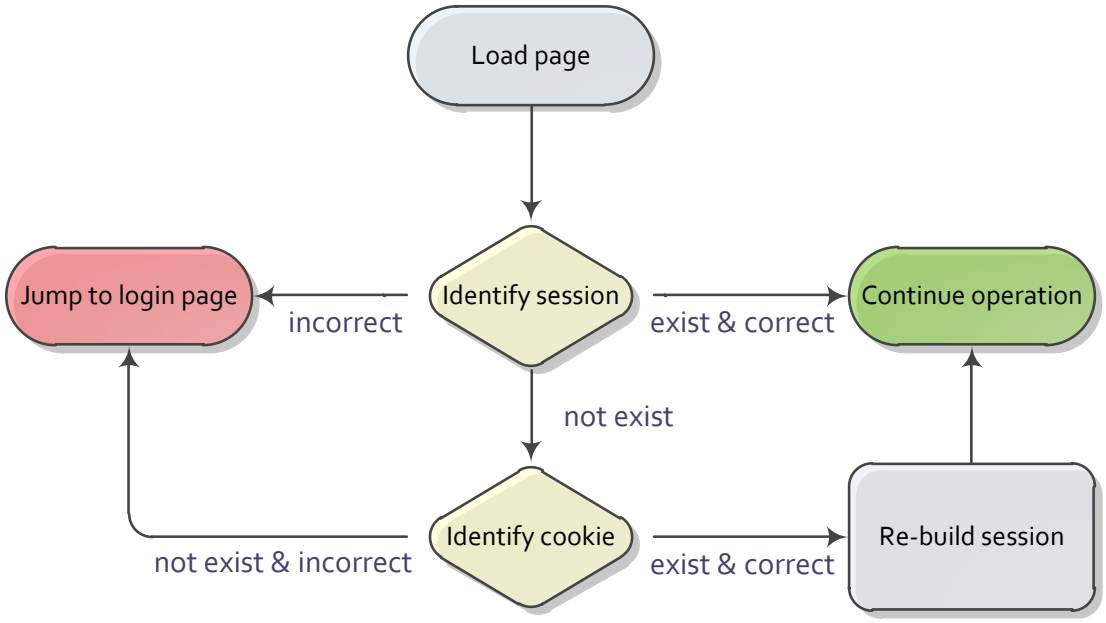

Figure 10 - Logical sequence of authentication in each page

To protect the safety of user's password, the database filed of password only stores MD5[22] value of password, which will be generated when login.

[21] Lifetime of Session is only until the browser closes. If the browser is closed, all Sessions will be destroyed automatically [89].
[22] MD5 is an irreversible encryption algorithm.

5.2.8 Performance optimisation

According to the requirements analysis related to response time, the performance of the system will directly affect the user experience. To give the user a better experience when carrying out operations within the system, the performance of the programme should be considered carefully. To enhance the performance of the system, in the programming part, the code related to the operations that use most time, such as database queries, has be designed to reduce the time taken. If a page needs to use information from a database table many times, it may require a very long time for processing. To reduce time wasted in this situation, the database tables will only be loaded once and then stored in a local data array. Repeated requests for information to the database table will be provided by the data array instead, and the time taken for requesting data will be much quicker than repeated database “selects”, because the information in the database is stored on the physical hard drive and the data in the data array is stored in RAM memory. The access speed of RAM memory is more than 500 times faster than accessing the hard drive [62]. So the performance will be greatly increased. The file related to performance improvement is “project.lib.php” with functions for selecting the database and putting it into a two-dimensional data array which means the database is only accessed once.

Even though the network bandwidth today is satisfactory, there are still some users who are using low speed internet access, especially mobile users and overseas users. To make sure that users who connected via low speed internet connections can still work on the project the page size which is loaded each time needs to be reduced to as small as possible. The use of CSS style sheet files will format the style of pages using the same file. So the CSS file can be cached for fast loading and applied to every page with the style. The system uses CSS for style design, so the formatting of each page only needs to call tags in the same CSS file, without separate code, which can be loaded much faster

As the design of the CSS followed the W3C standard guidance [63], it is very flexible. The traditional way of using images in formatting can be replaced in the latest standard of CSS, such as the background of table headers and links with “mouse over action” can be done by CSS description, instead of using images and JavaScript as before, therefore the page size at each load will be reduced.

5.2.9 Table sort

A lot of tables will be displayed for users, such as project list, task list, directory list and file list. Some of the tables contain sortable attributes like priority, start time and version code. If the contents in the table can be sorted, user may feel happier about organising and scheduling work. However, existing HTML does not provide features for sorting tables dynamically, even though the SQL database commands have parameters like "order by", it still needs to request a page refresh to make changes become effective, and the performance will be reduced by the number of database queries. An open-source jQuery[23] plugin called Tablesorter [64], solves this problem perfectly. By using the open-source JavaScript library, the tables displayed in the system can be sorted by any column of data. The user can sort by file name, version code, date or whatever he/she would like to sort, and the sorting results will be displayed in real-time, without a page refresh.

5.2.10 Migration and modification

This system has been defined as able to be used on any PHP and MySQL ready server. However, the running environment of the servers may vary. A small company may run their MySQL database service on the same server as the PHP running environment, but large companies may have separated PHP and MySQL servers. Some system administrators may call the system WVCS, but others may prefer to call it by the company name. Therefore, in order to solve the problem of running environment differences, a configuration file has been designed and implemented for storing all the configuration information, such as database connection details, system name, login basis, cookie validation time, system time zone and terms and conditions. This design allows system administrators to be able to configure the system to run on the local server much more easily. They only need to change the values by requirements in the configuration file, without modification of any programming code.

Property	Example value	Description
db_server	localhost	Database server name of IP address.
db_name	wvcs	Database name.
db_username_read	wvcs_read	User name of database read-only account.
db_password_read	york	Password of database read-only account.

[23] jQuery is a JavaScript library with good functions for dynamic effects on web pages [90].

db_username_write	wvcs_write	User name of database updating account.
db_password_write	york	Password of database updating account.
administrator_email	sy595@cs.york.ac.uk	E-mail address of system administrator.
system_name	Web-based Version Control System	Name of the version control system.
system_name_short	WVCS	Short name of the system.
system_version	1.0	System version number.
login_by	email	Login by verify which information of user.
after_login_redirect	summary.php	Jump to page after login.
cookie_valid	60	Cookie lifetime (days).
date_default_timezone_set	Europe/London	Time zone applied to the system.
terms_conditions	WVCS Terms and Conditions…	Content of terms and conditions (with HTML tags).

Table 3 - Configuration file information

To make the system easier to customise via secondary development, the programme has been designed and implemented with a library and style which have separate modes. The files of the libraries and styles have been placed into two folders called *libraries* and *styles*. If the customer only needs to change the styles and formatting of the system, only the files in the *styles* folder will need to be modified; if the customer needs to change features of the system, files in both the root directory and the *libraries* folder will need to change. The code for the files in both these folders has been well commented, so it will be much easier to find the part of the code which needs to be modified.

5.3 Database model

According to the requirements analysis for databases, the database tables will be designed in the third normal form (3NF) of database normalization. The properties of file and file changes will be recorded separately in the database, as will the directory and directory changes, task and task histories. This will ensure the tables are always in 3NF with no redundancy and the “update anomaly” will be avoided.

5.3.1 Entity-relationship modelling

There are nine tables in the database design: directory, directory_change, file, file_change, group_leader, project, task, task_history and user. Each of them has been linked by their "auto increase" primary keys and referenced foreigner keys.

The syntax of notations in the ERD is UML style notations.

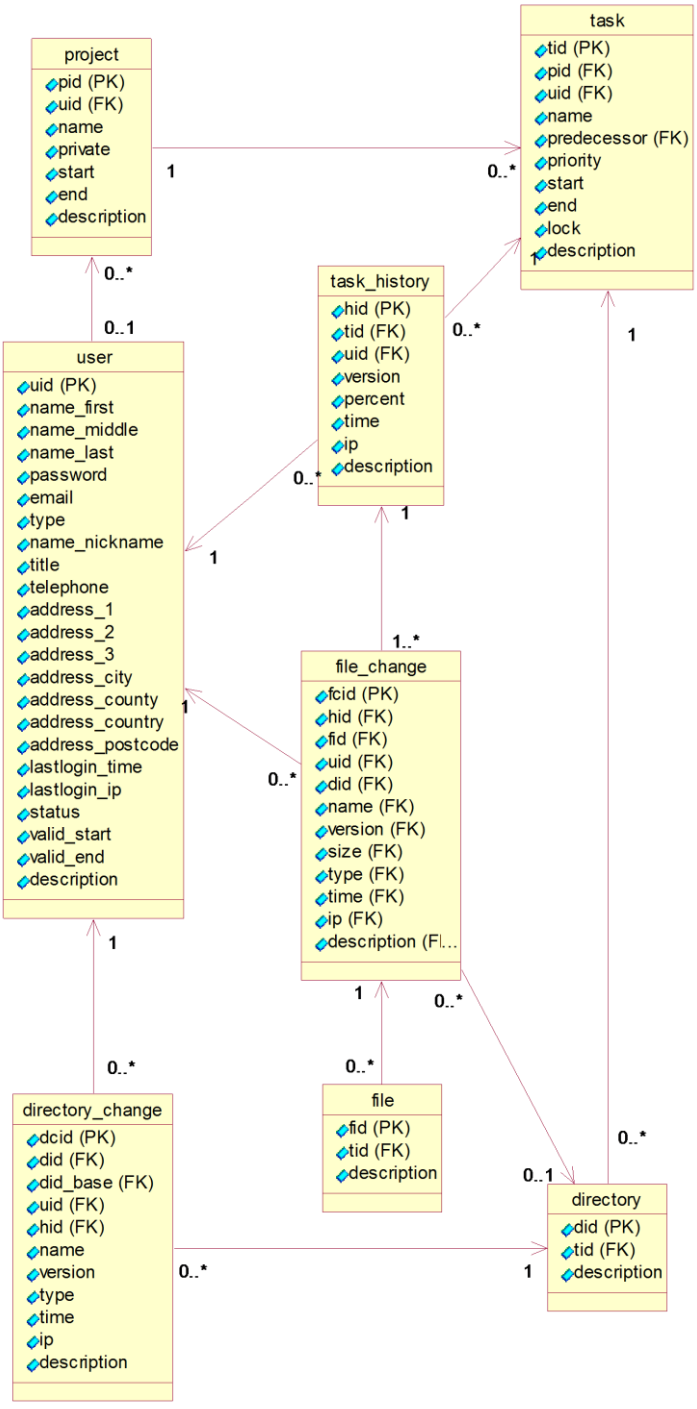

Figure 11 - Entity-relationship diagram

5.3.2 Attribute details of entities

In database table, all the fields[24] related to ID and numbers are all in integer type. Other text fields are in varchar type, with length of the maximum prediction the value would be. ID of each table is always in the first field. The details of the eight tables' attributes are listed below.

a. User

Field	Type	Null	Default	Comments
uid	int(8)	No		ID of user
name_first	varchar(100)	No		first name
name_middle	varchar(100)	Yes	NULL	middle name
name_last	varchar(100)	No		last name
password	varchar(50)	No		nick name
email	varchar(100)	No		email address
type	int(1)	No	1	type of user (0=admin, 1=regular)
name_nickname	varchar(50)	No		nick name
title	varchar(30)	Yes	NULL	title
telephone	varchar(30)	Yes	NULL	contact number
address_1	varchar(60)	Yes	NULL	address line 1
address_2	varchar(60)	Yes	NULL	address line 2
address_3	varchar(60)	Yes	NULL	address line 3
address_city	varchar(40)	Yes	NULL	city
address_county	varchar(40)	Yes	NULL	county
address_country	varchar(40)	Yes	NULL	country
address_postcode	varchar(15)	Yes	NULL	post code
lastlogin_time	datetime	Yes	NULL	last log-in time

[24] Field is another name of attribute in database entity.

Field	Type	Null	Default	Comments
lastlogin_ip	varchar(50)	Yes	NULL	last log-in IP address
status	int(1)	No	1	status (0=disabled, 1=active)
valid_start	datetime	No		register time or valid from time
valid_end	datetime	Yes	NULL	invalid time
description	varchar(8000)	Yes	NULL	description

Table 4 - Attribute details of user table

b. Project

Field	Type	Null	Default	Comments
pid	int(8)	No		ID of the project
uid	int(8)	Yes	NULL	ID of group leader
name	varchar(250)	No		project name
private	int(1)	No	0	identify private project or not
start	datetime	Yes	NULL	start time
end	datetime	Yes	NULL	due time
description	varchar(8000)	Yes	NULL	description

Table 5 - Attribute details of project table

c. Task

Field	Type	Null	Default	Comments
tid	int(8)	No		ID of the task
pid	int(8)	No		ID of project which the task belong to
uid	int(8)	No		ID of user which responsible to
name	varchar(500)	No		task name
predecessor	int(8)	Yes	NULL	ID of predecessor task
priority	int(1)	No	1	task priority
start	datetime	Yes	NULL	start time
end	datetime	Yes	NULL	due time

Field	Type	Null	Default	Comments
lock	int(1)	No	0	status of locked or not
description	varchar(8000)	Yes	NULL	description

Table 6 - Attribute details of task table

d. Task_history

Field	Type	Null	Default	Comments
hid	int(12)	No		ID of this task history
tid	int(10)	No		ID of task which the history related to
uid	int(8)	No		ID of user who updated the task history
version	int(5)	No		version number of the history
percent	int(3)	Yes	NULL	percentage finished predict
time	datetime	No		time of the history submitted
ip	varchar(30)	Yes	NULL	IP log
description	varchar(8000)	Yes	NULL	description

Table 7 - Attribute details of task_history table

e. Directory

Field	Type	Null	Default	Comments
did	int(11)	No		ID of the directory
tid	int(8)	No		ID of task which this directory belong to
description	varchar(8000)	Yes	NULL	description

Table 8 - Attribute details of directory table

f. Directory_change

Field	Type	Null	Default	Comments
dcid	int(12)	No		ID of the directory modification
did	int(11)	No		ID of which directory is related to
did_base	int(11)	Yes	NULL	Upper level directory ID
uid	int(8)	No		ID of user who did this modification

Field	Type	Null	Default	Comments
hid	int(12)	No		ID of task change history record which the directory related to
name	varchar(259)	No		directory name
version	int(5)	No		version of this modification
type	int(1)	No		type of operation of the modification
time	datetime	No		time of commit
ip	varchar(30)	Yes	NULL	IP record of the modification
description	varchar(8000)	Yes	NULL	description

Table 9 - Attribute details of directory_change table

g. File

Field	Type	Null	Default	Comments
fid	int(11)	No		ID of the file
tid	int(10)	No		ID of task which this file belong to
description	varchar(8000)	Yes	NULL	description

Table 10 - Attribute details of file table

h. File_change

Field	Type	Null	Default	Comments
fcid	int(12)	No		ID of the file modification
hid	int(12)	No		ID of task change history record which this file related to
fid	int(11)	No		ID of the file record
uid	int(8)	No		ID of user who did this modification
did	int(11)	Yes	NULL	ID of which directory the file modification belong to
name	varchar(259)	No		file name in this modification
version	int(5)	No		version number

Field	Type	Null	Default	Comments
size	int(25)	No		file size
type	int(1)	No	0	type of operation
time	datetime	No		time record of the modification
ip	varchar(30)	Yes	NULL	IP log
description	varchar(8000)	Yes	NULL	description

Table 11 - Attribute details of file_change table

5.4 Prototype design

In order to find out the problems within the design, some low-fi prototypes are being used during the design stage. In the low-fi prototype design, the interface of main pages in the system has been drawn by pen and paper.

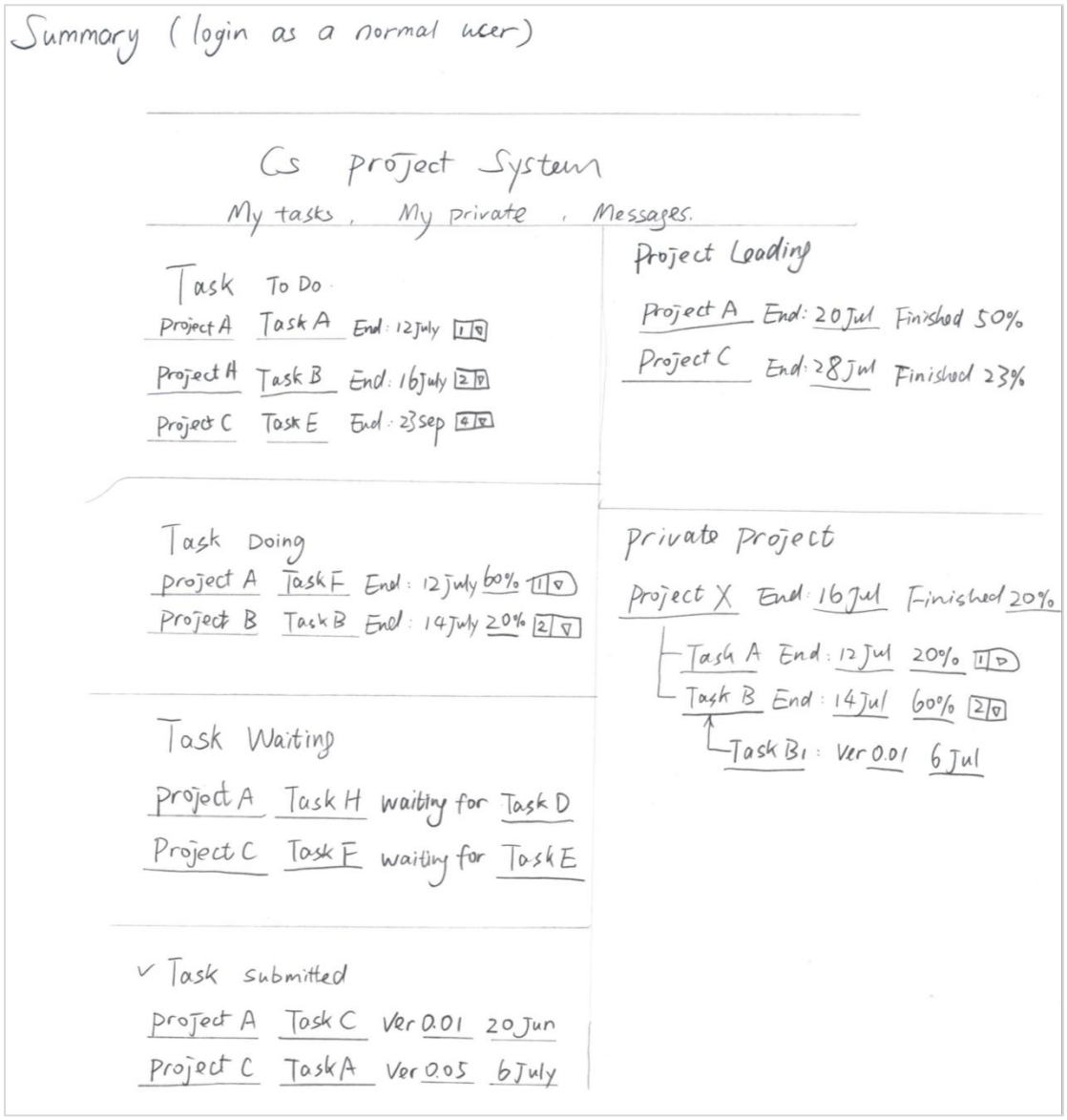

Figure 12 - Prototype: Summary page

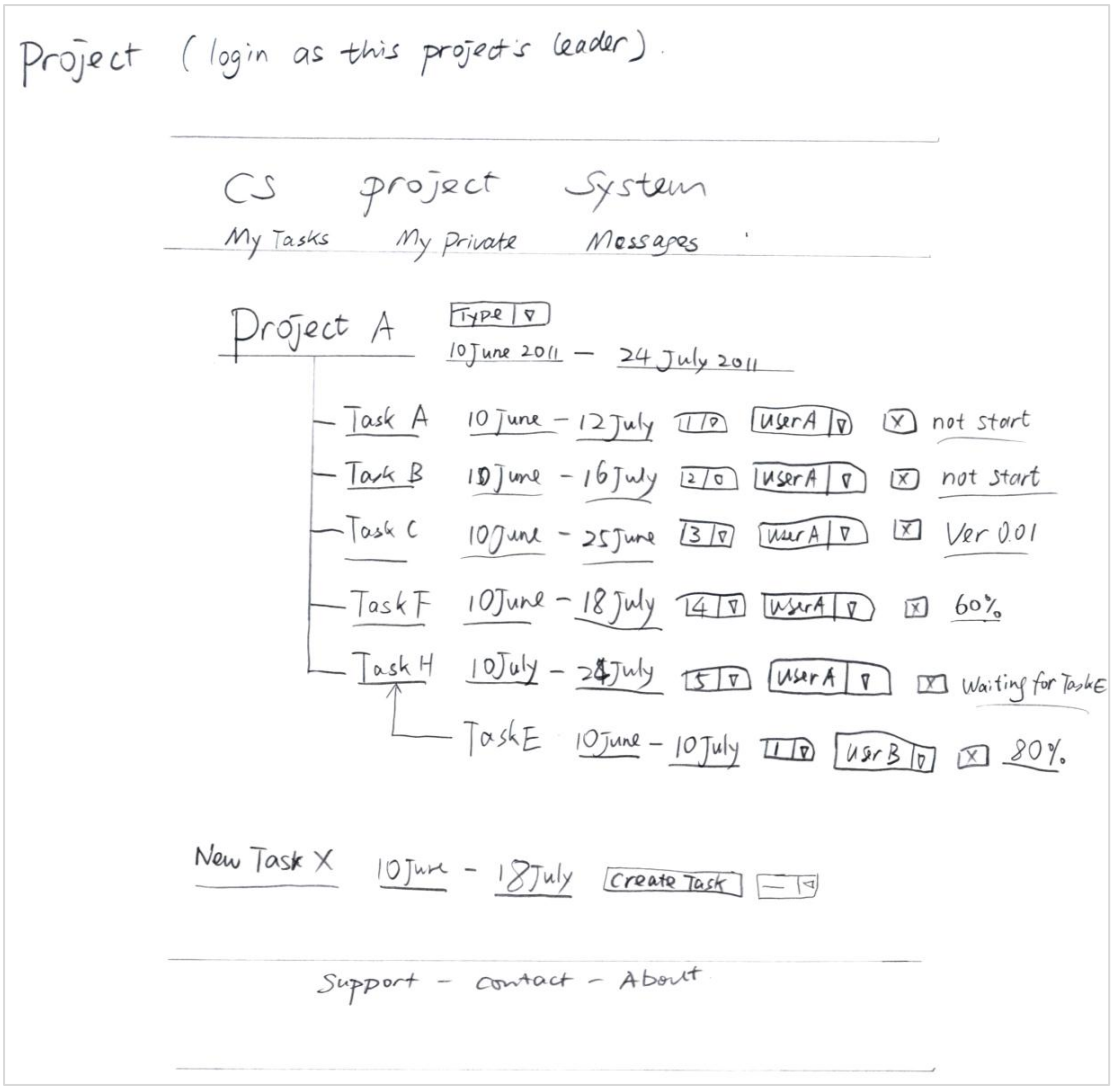

Figure 13 - Prototype: Group leader managing projects and tasks

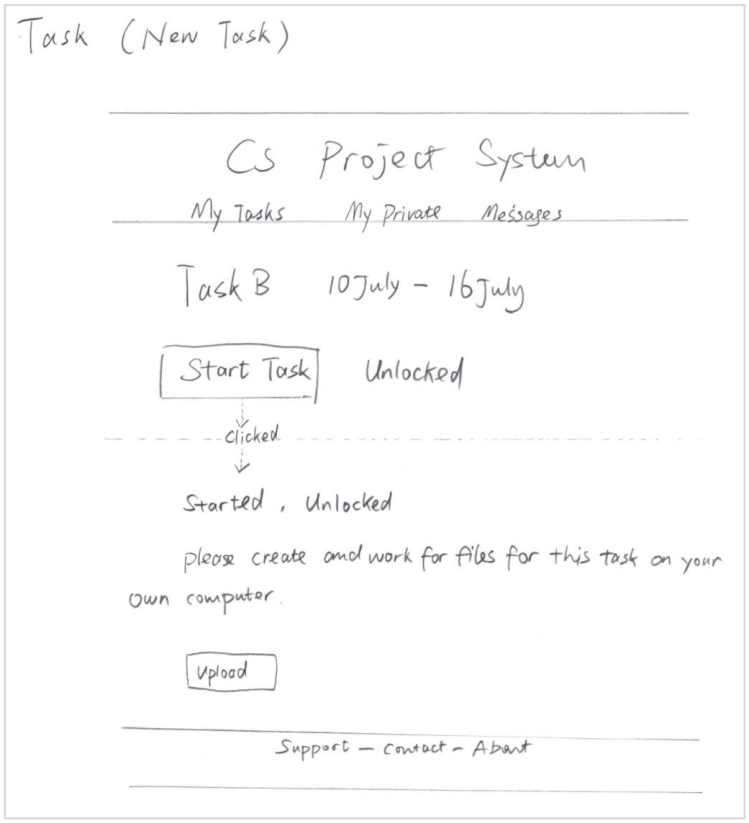

Figure 14 - Prototype: Starting task

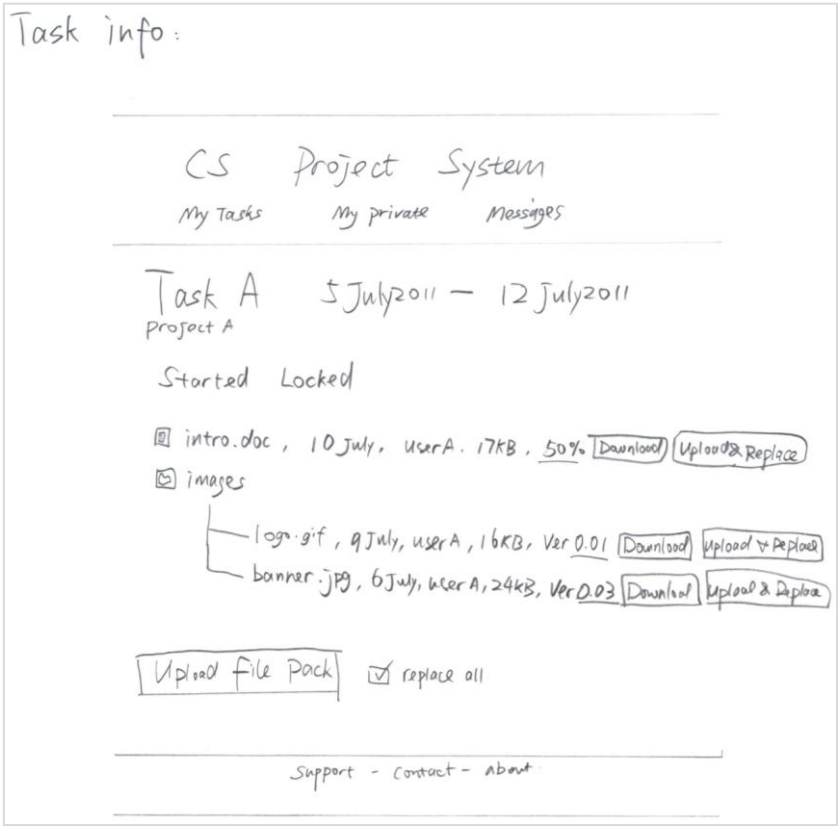

Figure 15 - Prototype: Task information and files related

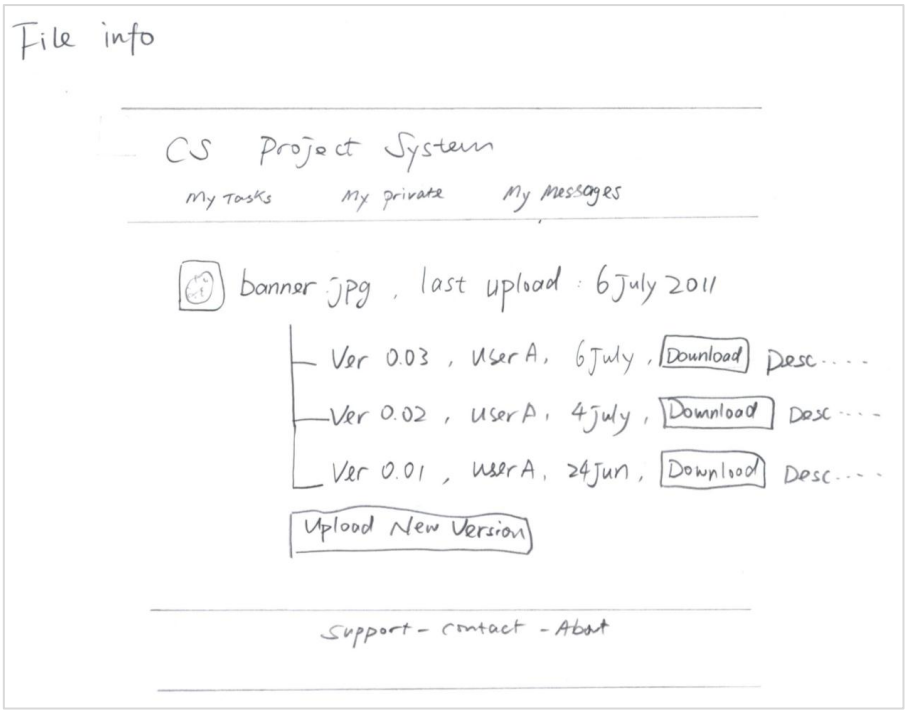

Figure 16 - Prototype: File information and versions related

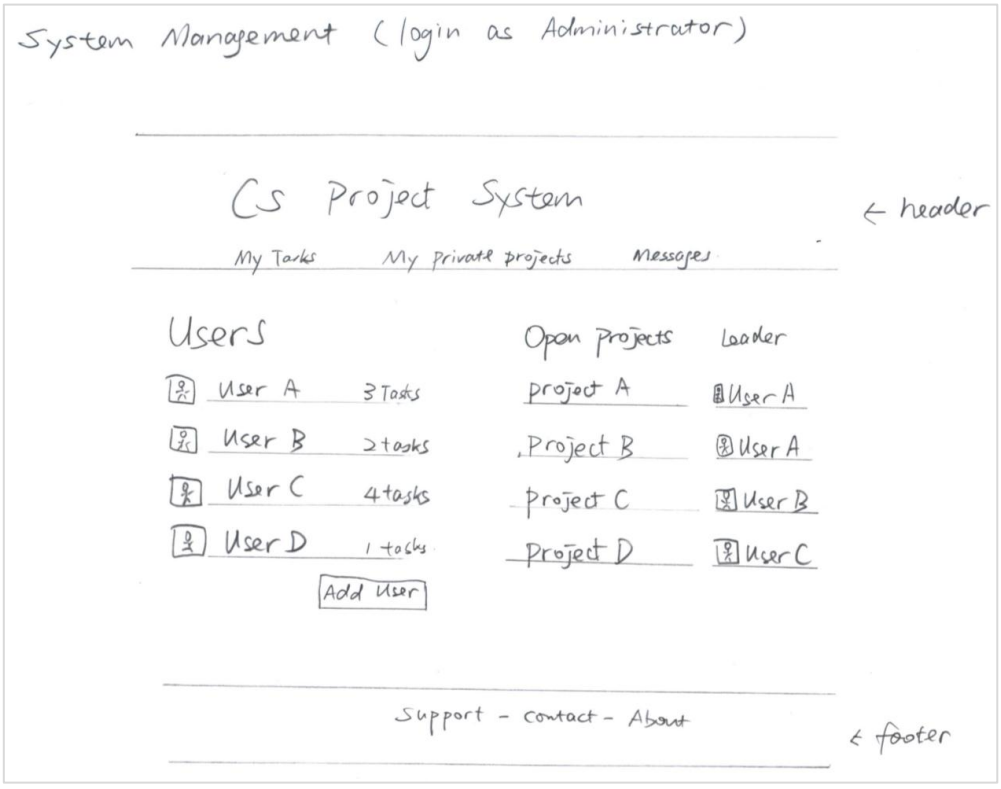

Figure 17 - Prototype: System management page

5.5 Prototype evaluation

In the prototypes, the header part in each page takes too much space and only provides little information. To give more space to main body than header, the height of header has been reduced.

In group leader's page of group project and task management, the delete button is too easy to be mistakenly hit. And the delete operation may use in very low frequency. Therefore, the delete button can move from tasks list to task information/details page.

The sign in and sign out button is missing in the prototype, even though close browser would automatically logged out, however, it may still leads user confused about their personal information. A sign in and sign out link is required.

There is no search bar provided on pages. Even though the system have dedicated task searching page, however, when user feels hard to find their tasks, they may wish to search the task name on any page directly.

5.6 Interface design

According to the requirement analysis of platform compatibility, there are lots of users still using screen with 1024 pixel wide. To make sure this kind of user can view pages without horizontal scrolling, after subtracting the width of browser border and vertical scroll bar on the right, the page width is sets at 960 pixels.

Generally, all the pages are divided into three parts: header, main body and footer. Header is responsible for the display of navigation bar (top part of page) and defining page properties; footer is used to presenting copyright information or other sentence need to be placed in the end of page.

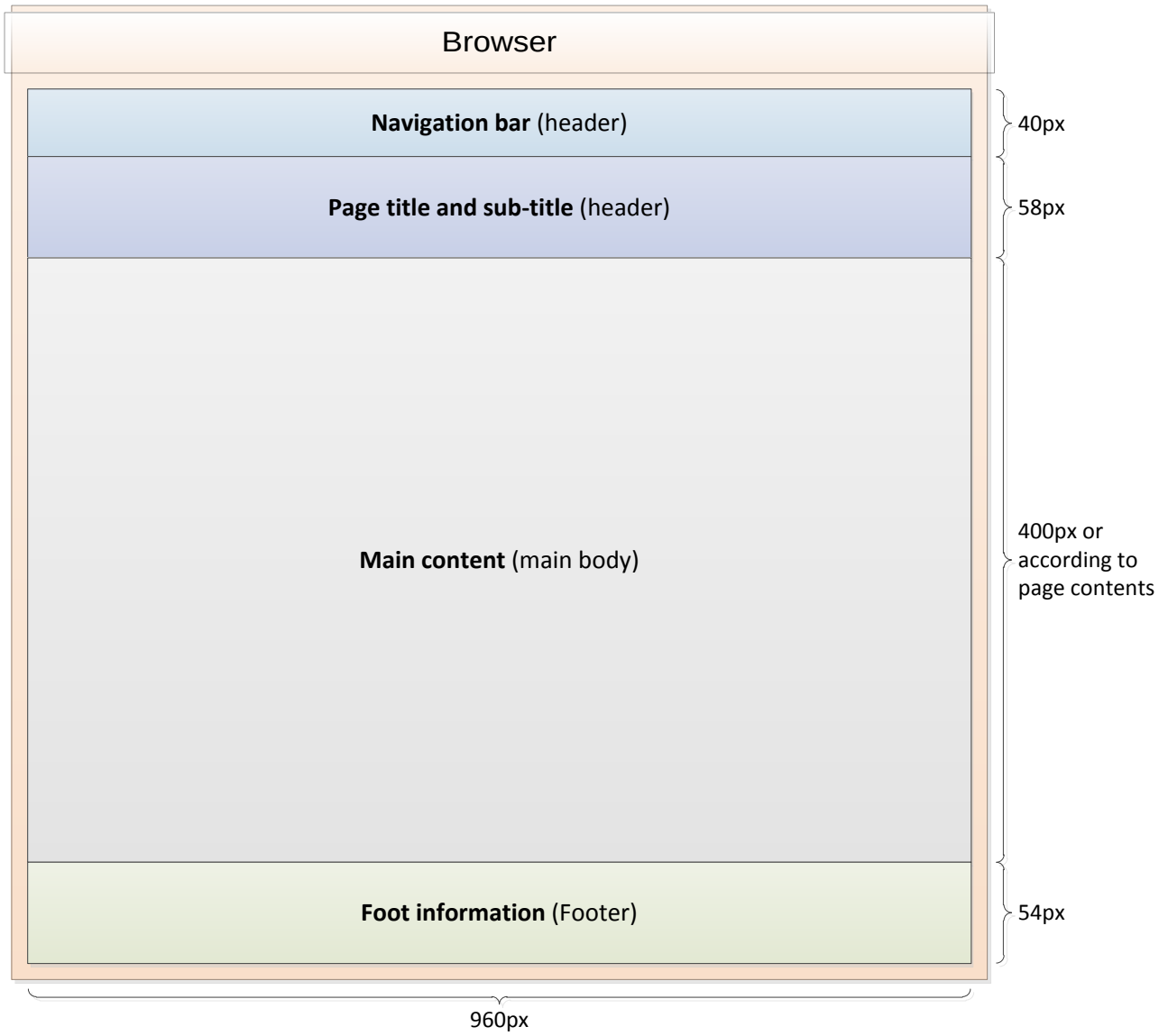

Table 12 - Page layout

The details style design is part from open-source CSS styles from Twitter's Bootstrap [65]. It is mature styling system with rich consideration of interaction design. The CSS from Bootstrap has been placed in the file style/bootstrap-1.2.0.css, and the self-defined supplements of Bootstrap are in the file style/common.css.

5.7 Accessibility

To give barrier free browsing to everyone, the page design considered some accessibility principles of web design in W3C WCAG 1.0 documentation [35]. The rules from the documentation of making website accessible have been performed into design and implementation.

a. Provide equivalent alternatives to auditory and visual content.

All the images have been placed along with “alt” tag for providing further information of the images. Screen reader could find the “alt” tag of images, do synthesized speech of them, and read out to the people with visual disabilities.

b. Don't rely on colour alone.

The colour use in page design of the system is only used in improving information identification. If colour has been removed, information can still be identified.

c. Use markup and style sheets and do so properly.

In organising page contents, CSS has been used for mark-up styles of content instead of traditional HTML control, such as controlling font and diversion properties. Nest OL, UL, and DL list is used in classify the level of information.

d. Clarify natural language usage

Language property is defined in HTML “lang” attribute to help screen reader identify the language using in the pages.

e. Create tables that transform gracefully.

The tables in the pages are always having logical level of row and column headers (thead).

f. Ensure that pages featuring new technologies transform gracefully.

To maximum the compatibility of all browsers, the JavaScript only be used seldom in page design. Once JavaScript has been used, and alternative way of it feature will also be designed in server side applications.

g. Ensure user control of time-sensitive content changes.

There are no automatic refreshing and unnecessary redirection in pages.

5.8 Two layer PHP architecture

In the design of this system, the PHP files were divided into two parts: function libraries and controllers. In the controller part, the block of PHP code is placed in the head of each PHP file, before the output of HTML code. Function libraries are focus on storing PHP functions for controller part files to call and use. To make the function libraries usable, the location and file name of library files should be “included” in the header of controller files.

This architecture will increase the readability of the code of the system, also make system administrator easy in maintenance. Programme efficiency would also increase, because PHP have cache mechanism in caching used files after compilation for next time use. If the library files have been cached after called by a controller file, when next time another controller file need to use the library file, it can be much faster.

5.9 Compatibility

As a web-based application, the compatibility is about the experience in different browsers. According to the requirement analysis of compatibility in chapter 0, lots kinds of web browsers are used by different people. It is not like years before that no one takes the most market share. The common point of the browsers is most of them supports the standard defined by W3C HTML and CSS. Therefore, the design and implementation was strictly followed the guide from W3C.

5.10 CSS classes multiple use

To avoid the CSS files becoming too large, so that they can be loaded quickly and without redundancy, in the system design, the use of CSS has been designed in fully multiple use mode. The CSS multiple use is a lesser known way to reduce CSS file size effectively, which means more than one CSS class can be referenced in one HTML element at the same time, by splitting the class names with a space [66].

6. Evaluation and testing

In this project, the evaluation and testing was performed by following the functional tests.

6.1 Testing of version control

The version control testing is a case based testing, which has been tested by the developer himself manually by following cases below:

a. User login

Login as a user, redirect to summary page. Passed.

b. Administrator login

Login as an administrator, redirect to project management panel. Passed.

c. Create project

Create both a group project a project start from 7th Sep 2011, due at 13th Sep 2011. Passed.

d. Create task

Create several tasks start from 7th Sep 2011, due at 13th Sep 2011 with different priorities. Passed.

e. Set predecessor task

Set up a predecessor task to a sub-task. Passed.

f. Change task priority

Change priority of a task. Passed.

g. Create and upload file

Create a new file. Upload an existing file. Passed.

h. Create directory

Create a directory and change location of a file to the directory. Passed.

i. Set parent directory

Change relationship with set up of a parent directory. Passed.

j. Success message testing

Get success message after successful operations. Passed.

k. Error message testing

Get error message after both a user error and a system error. Passed.

6.2 Testing of administration

Administration testing is also case based. The following cases have been tested in sequence.

a. Create user

Create a new user with name, address, telephone. Passed.

b. List projects

List all group projects within the system. Passed.

c. View group projects

View several group projects via link of group project list. Passed.

6.3 Compatibility testing

The testing of system compatibility is a subjective evaluation. The developer tested the interface of some typical pages includes task listing page, summary page and login page. There are five popular browsers (Firefox, Chrome, Safari, Opera, and IE) have been tested in five widely used operating systems including desktop and mobile platform (Windows, Linux, Mac OS, iOS, Android). As the system has been designed and implemented by following HTML and CSS standard from W3C strictly. Therefore, in the testing, details of the changes have been considered as differences.

Browsers	Overall similarity to expected design
Mozilla Firefox 3.5 (Linux)	85%
Mozilla Firefox 4.0 (Windows)	90%
Google Chrome 13 (Windows)	99%
Google Chrome (Android)	95%
Apple Safari 5.1 (Mac OS)	95%
Apple Safari (iOS)	92%
Opera 11.50 (Windows)	95%
Microsoft Internet Explorer 9 (Windows)	80%
Microsoft Internet Explorer 7 (Windows)	65%

Table 13 - Compatibility testing results

After the subjective evaluation, Google chrome has been identified as has the most similarity to W3C standardised design. In contrast, Microsoft Internet Explorer has the least similarity, even the latest and revolutionary version, still lower than the average. The compatibility testing on mobile devices receives a satisfactory result, which means the system could be used at mobile devices with good experience.

7. Conclusion

This project is an interesting design about tracking works in computer based group and individual projects. “Task” as a new tracking unit has been firstly proposed and implemented in this area. The new fully web based interface and lightweight functions provides user a relaxed way in doing version control without excessive worry.

I am happy that I have learnt a lot of new knowledge from the whole project process and progression. My personal ability has been improved by constantly trying to solve problems, doing the literature review, programming and writing project report.

After the evaluation, the system has been verified as running without obvious bugs. However, due to the tight time in testing, some bugs may not be discovered and corrected in time. As the author of this project, I will continue working for it, because the deep motive of I doing this project is because of my interest. Some work still needs to be done in the future, to ensure that the system is able to continue serving users. There are:

a. Easier local storage

An obvious drawback of web-based application is that the web pages cannot communicate to local file storage automatically. It may because of security consideration. However it brought a lot of trouble for programme designer and user. In this system, user needs to upload their files via web page manually, because it cannot be automatically detected and transferred. Good news is, after the widely spread use of HTML5, its mysterious features in local storage may be discovered and developed. After modification the system to use HTML5 local storage, the operation of users of doing upload would be much easily than before.

b. Multi task assignment

Due to the design of database and auto-lock mechanism, a task can be only assigned to one user. In further development, the system can be developed to support assign a task to more than one user.

c. Diff storage and analysis

In the very tight of doing this project, a very interesting feature in my original idea missed in the design and implementation. That is the difference comparison (diff) feature. The mechanism of diff is clear [67], however, due to it requests time to design algorithm, the feature finally be passed. If the system can do diff, each upload from user may show the

differences between versions, and user can understand what is he/she did and what are other people did.

d. Automatic merging

The system require group leader to do merge of works manually, it is also because there are no enough to develop merging algorithm. It can be imaged that if the system can merge tasks part or full automatically, the efficiency of group working may increase to a new higher level.

References

[1] G. Stamatellos, Computer Ethics: A Global Perspective, Jones and Bartlett, 2007.

[2] UK Parliament, *Data Protection Act 1998,* London, 1998.

[3] ACM Council, "ACM Code of Ethics and Professional Conduct," 16 Oct 1992. [Online]. Available: http://www.acm.org/about/code-of-ethics. [Accessed 2 Sep 2011].

[4] B. Collins-Sussman, F. W. Brian and C. M. Pilato, Version Control with Subversion, O'Reilly, 2004.

[5] B. Danella, "Rapid Subversion adoption validates enterprise readiness and challenges traditional software configuration management leaders," 15 May 2007. [Online]. Available: http://www.open.collab.net/news/press/2007/svn_momentum.html. [Accessed 10 July 2011].

[6] D. Price, "CVS v1.11.23 Manual," Ximbiot LLC, 8 May 2008. [Online]. Available: http://ximbiot.com/cvs/manual/cvs-1.11.23/cvs.html. [Accessed 1 July 2011].

[7] The Apache Software Foundation, "Apache Subversion Features," [Online]. Available: http://subversion.apache.org/features.html. [Accessed 1 July 2011].

[8] T. Oakden, "None Concurrent Access in Version Control," 12 Oct 2009. [Online]. Available: http://forum.unity3d.com/threads/36536-None-concurrent-access-in-version-control. [Accessed 22 July 2011].

[9] P. Roy, "Understanding Subversion's Problems," 9 Mar 2011. [Online]. Available: http://ventspace.wordpress.com/2011/03/09/understanding-subversions-problems/. [Accessed 22 July 2011].

[10] Jbcrouigneau, "Task Oriented Development and Validation Space," 24 Apr 2009. [Online]. Available: http://www.svnforum.org/threads/36840-Task-oriented-development-and-Validation-space?s=cf3b028492de3003320a35e609f4777b. [Accessed 22 July 2011].

[11] H. Gantt, Work, Wages and Profit, New York: The Engineering Magazine, 1910.

[12] Shawn, "Which is More Popular (Currently, by Recent Install Base) SVN or CVS?," 23 Apr 2009. [Online]. Available: http://stackoverflow.com/questions/782375/which-is-more-popular-currently-by-recent-install-base-svn-or-cvs. [Accessed 23 July 2011].

[13] C. Duan, "Understanding Git Conceptually," 17 Apr 2010. [Online]. Available: http://www.eecs.harvard.edu/~cduan/technical/git/. [Accessed 23 July 2011].

[14] T. Spencer, "Setup a Subversion Server in 4 Minutes," 2 Mar 2007. [Online]. Available: http://www.tonyspencer.com/2007/03/02/setup-a-subversion-server-in-4-minutes/. [Accessed 23 July 2011].

[15] "Setting Up Subversion," July 2006. [Online]. Available:

http://systhread.net/texts/200607subver.php. [Accessed 23 July 2011].

[16] D. Thomas and A. Hunt, Pragmatic Version Control Using CVS, Pragmatic Bookshelf, 2003.

[17] Google Inc., "Top ten advantages of Google's cloud," 2011. [Online]. Available: http://www.google.com/apps/intl/en/business/cloud.html. [Accessed 24 July 2011].

[18] L. L. Peterson and S. B. Davie, Computer Networks : A Systems Approach, Amsterdam; London: Morgan Kaufmann, 2007.

[19] Oracle, "Java Servlet Technology Overview," [Online]. Available: http://www.oracle.com/technetwork/java/overview-137084.html. [Accessed 10 July 2011].

[20] M. Hall, "Building Web Apps in Java: Beginning & Intermediate Servlet & JSP Tutorials," 2011. [Online]. Available: http://courses.coreservlets.com/Course-Materials/csajsp2.html. [Accessed 10 July 2011].

[21] TechyShell.com, "ASP – Its Advantages and Disadvantages," 27 May 2009. [Online]. Available: http://www.techyshell.com/internet/asp-its-advantages-and-disadvantages/. [Accessed 13 July 2011].

[22] The PHP Group, "The PHP License, version 3.01," 2010. [Online]. Available: http://www.php.net/license/3_01.txt. [Accessed 11 July 2011].

[23] Z:WAMP Group, "Z:WAMP Server Pack," 7 Nov 2010. [Online]. Available: http://zwamp.sourceforge.net/. [Accessed 21 July 2011].

[24] R. Bourdon, "WampServer," 24 Dec 2010. [Online]. Available: http://www.wampserver.com/en/. [Accessed 21 July 2011].

[25] E. Group, "EasyPHP," 2011. [Online]. Available: http://www.easyphp.org/introduction.php. [Accessed 12 July 2011].

[26] B. Shire, "PHP and Facebook," 3 May 2007. [Online]. Available: http://www.facebook.com/blog.php?post=2356432130. [Accessed 10 7 2011].

[27] Daniel, "Benefits Of MySQL," 20 Nov 2010. [Online]. Available: http://benefitof.net/benefits-of-mysql/. [Accessed 13 July 2011].

[28] A. Cooper, R. Reimann and D. Cronin, About Face 3: The Essentials of Interaction Design, Wiley, 2007.

[29] H. Sharp, Y. Rogers and J. Preece, Interaction Design : Beyond Human-computer Interaction, Chichester: Wiley, 2007.

[30] J. Nielsen, "Usability 101: Introduction to Usability," [Online]. Available: http://www.useit.com/alertbox/20030825.html. [Accessed 24 7 2011].

[31] M. Miller, Cloud computing: Web-based applications that change the way you work and collaborate online, Que, 2008.

[32] Google Inc., “Boost productivity with Google-powered collaboration apps,” 2011. [Online]. Available: http://www.google.com/apps/intl/en/business/collaboration.html. [Accessed 27 July 2011].

[33] S. Pichai, “Introducing the Google Chrome OS,” 7 July 2009. [Online]. Available: http://googleblog.blogspot.com/2009/07/introducing-google-chrome-os.html. [Accessed 27 July 2011].

[34] S. G. Cohen and D. E. Bailey, “What Makes Teams Work: Group Effectiveness Research from the Shop Floor to the Executive Suite,” *Journal of Management,* vol. 23, no. 3, pp. 239-290, 1977.

[35] W. Chisholm, G. Vanderheiden and I. Jacobs, “Web Content Accessibility Guidelines 1.0,” 5 May 1999. [Online]. Available: http://www.w3.org/TR/WCAG10/. [Accessed 27 July 2011].

[36] G. Adams-Spink, “New guidelines boost web access,” 22 Dec 2008. [Online]. Available: http://news.bbc.co.uk/1/hi/technology/7789622.stm. [Accessed 24 July 2011].

[37] S. Ahern, D. Eckles, N. Good, S. King, M. Naaman and R. Nair, “Over-Exposed? Privacy Patterns and Considerations in Online and Mobile Photo Sharing,” in *CHI '07: Proc. of the SIGCHI Conf., Human Factors in Computing Systems*, 2007.

[38] J. Nielsen, “Response Times: The 3 Important Limits,” 1993. [Online]. Available: http://www.useit.com/papers/responsetime.html. [Accessed 27 July 2011].

[39] R. B. Miller, “Response time in man-computer conversational transactions.,” in *AFIPS Fall Joint Computer Conference*, 1968.

[40] C. Heng, “Designing Your Website for Browser and Platform Compatibility,” 5 Sep 2008. [Online]. Available: http://www.thesitewizard.com/archive/compatibility.shtml. [Accessed 27 July 2011].

[41] W3schools.com, “Browser Statistics,” June 2011. [Online]. Available: http://www.w3schools.com/browsers/browsers_stats.asp. [Accessed 29 July 2011].

[42] W3school.com, “Browser Display Statistics,” Jan 2011. [Online]. Available: http://www.w3schools.com/browsers/browsers_display.asp. [Accessed 29 July 2011].

[43] R. Adams, “JavaScript is Good, But Should Not be Relied Upon,” 19 July 2009. [Online]. Available: http://wblinks.com/notes/javascript-is-good-but-should-not-be-relied-upon. [Accessed 29 July 2011].

[44] CloudTweaks, “Cloud Computing – Demystifying SaaS, PaaS and IaaS,” 3 May 2010. [Online]. Available: http://www.cloudtweaks.com/2010/05/cloud-computing-demystifying-saas-paas-and-iaas/. [Accessed 20 June 2011].

[45] OpenCrowd, “Software as a Service (SaaS),” 2010. [Online]. Available: http://cloudtaxonomy.opencrowd.com/taxonomy/software-as-a-service/. [Accessed 20 June 2011].

[46] S. Schwartz, "Just what is 'SaaS' and are some vendors abusing the term?," 21 Mar 2011. [Online]. Available: http://blog.connectedplanetonline.com/unfiltered/2011/03/21/just-what-is-saas-and-are-some-vendors-abusing-the-term/. [Accessed 30 June 2011].

[47] J. Brodkin, "Gartner: Seven cloud-computing security risks," 2 July 2008. [Online]. Available: http://www.infoworld.com/d/security-central/gartner-seven-cloud-computing-security-risks-853?page=0,0. [Accessed 30 June 2011].

[48] GPS Systems, "10 Reasons Why You Need SaaS Web-based Telematics," 12 Dec 2009. [Online]. Available: http://gpssystems.net/10-reasons-saas-webbased-telematics/. [Accessed 30 June 2011].

[49] W. Royce, "Managing the Development of Large Software Systems," in *IEEE WESCON 26*, 1970.

[50] S. McConnell, Rapid Development: Taming Wild Software Schedules, Redmond: Microsoft Press, 1996.

[51] B. W. Boehm, "A Spiral Model of Software Development and Enhancement," *Computer,* pp. 61-72, May 1988.

[52] P. Smith, "File:Waterfall model (1).svg," 8 Mar 2009. [Online]. Available: http://en.wikipedia.org/wiki/File:Waterfall_model_(1).svg. [Accessed 30 June 2011].

[53] C. M. Mcclendon, L. Regot and G. Akers, "What is Prototyping?," 26 May 1999. [Online]. Available: http://www.umsl.edu/~sauterv/analysis/prototyping/proto.html. [Accessed 30 June 2011].

[54] P. P.-s. Chen, "The Entity-Relationship Model: Toward a Unified View of Data," *ACM Transactions on Database Systems,* pp. 9-36, March 1976.

[55] S. W. Ambler, "Data Modeling 101," [Online]. Available: http://www.agiledata.org/essays/dataModeling101.html. [Accessed 30 June 2011].

[56] E. F. Codd, "Further Normalization of the Data Base Relational Model," IBM Research Report, New York City, 1971.

[57] R. J. Rustin, Data Base Systems: Courant Computer Science Symposia Series 6, New York City: Prentice-Hall, 1972.

[58] E. F. Codd, The Relational Model for Database Management, Addison-Wesley, 1990.

[59] C. Zaniolo, "A New Normal Form for the Design of Relational Database Schemata," *ACM Transactions on Database Systems,* vol. 3, no. 7, 1982.

[60] R. Elmasri, Fundamentals of Database Systems, London: Addison-Wesley, 2007.

[61] S. P. Robbins and M. Coulter, Management, Prentice Hall, 2007.

[62] E. Larsen, "Ram disk 500 times faster than hard drive," 22 Mar 2007. [Online]. Available: http://www.computeractive.co.uk/pcw/news/1922962/ram-disk-500-times-faster-hard-

drive. [Accessed 1 Aug 2011].

[63] W3Schools.com, “CSS Tutorial,” 2011. [Online]. Available: http://www.w3schools.com/css/. [Accessed 1 Aug 2011].

[64] C. Bach, “Tablesorter: Flexible client-side table sorting,” 17 Mar 2008. [Online]. Available: http://tablesorter.com/docs/. [Accessed 1 Aug 2011].

[65] M. Otto and J. Thornton, “Bootstrap, from Twitter,” 2011. [Online]. Available: https://github.com/twitter/bootstrap. [Accessed 30 July 2011].

[66] J. Kyrnin, “Use Multiple CSS Classes on a Single Element,” 2011. [Online]. Available: http://webdesign.about.com/od/css/qt/tipcssmulticlas.htm. [Accessed 26 July 2011].

[67] K. Azad, “A Visual Guide to Version Control,” 27 Sep 2007. [Online]. Available: http://betterexplained.com/articles/a-visual-guide-to-version-control/. [Accessed 30 8 2011].

[68] Oracle, “JavaServer Pages Technology,” [Online]. Available: http://www.oracle.com/technetwork/java/javaee/jsp/index.html. [Accessed 2 July 2011].

[69] S. Chacon, “Git - The Fast Version Control System,” 2011. [Online]. Available: http://git-scm.com/. [Accessed 23 July 2011].

[70] J. Loeliger, Version Control with Git, O'Reilly, 2009.

[71] R. M. Hinden and B. Haberman, “RFC 4193 : Unique Local IPv6 Unicast Addresses,” Feb 2005. [Online]. Available: http://tools.ietf.org/html/rfc4193. [Accessed 24 July 2011].

[72] Y. Rekhter, R. G. Moskowitz, D. Karrenberg, G. J. d. Groot and E. Lear, “RFC 1918 : Address Allocation for Private Internets,” Feb 1996. [Online]. Available: http://tools.ietf.org/html/rfc1918. [Accessed 24 July 2011].

[73] P. Mockapetris, “The Domain Name System,” in *Proceedings of the IFIP 6.5 Working Conference on Computer Message Services*, Nottingham, 1984.

[74] P. Mockapetris, J. Postel and P. Kirton, “Name Server Design for Distributed Systems,” in *Proceedings of the Seventh International Conference on Computer Communication*, Sidney, 1984.

[75] P. Mockapetris, “RFC 882 : Domain Names - Concepts and Facilities,” Nov 1983. [Online]. Available: http://tools.ietf.org/html/rfc882. [Accessed 24 July 2011].

[76] P. Mockapetris, “RFC 883 : Domain Names - Implementation and Specification,” Nov 1983. [Online]. Available: http://tools.ietf.org/html/rfc883. [Accessed 24 July 2011].

[77] D. Parrack, “New Twitter.com UI is faster, better,” 15 Sept 2010. [Online]. Available: http://tech.blorge.com/Structure:%20/2010/09/15/new-twitter-com-ui-is-faster-better/. [Accessed 25 July 2011].

[78] Microsoft Corporation, “ASP.NET Web Pages,” 2011. [Online]. Available:

http://www.asp.net/web-pages. [Accessed 2 July 2011].

[79] A. Sami, "What is New in ASP.NET 4.0, Visual Studio 2010 IDE," 13 Jan 2010. [Online]. Available: http://www.codeproject.com/KB/aspnet/Whatis_New_ASP_Net_4.aspx. [Accessed 2 July 2011].

[80] C. D. Knuckles and D. S. Yuen, Web Applications: Concepts & Real World Design, Hoboken, N.J.: John Wiley & Sons, Inc., 2005.

[81] D. Robinson and A. L. K. Coar, "RFC 3875: The Common Gateway Interface (CGI) Version 1.1," Oct 2004. [Online]. Available: http://tools.ietf.org/html/rfc3875. [Accessed 10 7 2011].

[82] The Apache Software Foundation, "Apache HTTP Server Project," 2011. [Online]. Available: http://httpd.apache.org/. [Accessed 13 July 2011].

[83] Igor Sysoev, "Nginx," [Online]. Available: http://nginx.org/en/. [Accessed 13 July 2011].

[84] Microsoft Corporation, "IIS: Overview," 2011. [Online]. Available: http://www.iis.net/overview. [Accessed 13 July 2011].

[85] J. Lee and B. Ware, Open source Web development with LAMP using Linux, Apache, MySQL, Perl, and PHP, Boston: Addison-Wesley, 2003.

[86] Oracle Corporation, "MySQL Standard Edition," 2010. [Online]. Available: http://www.mysql.com/products/standard/. [Accessed 13 July 2011].

[87] TutorialsPoint.COM, "Database - Second Normal Form (2NF)," 2011. [Online]. Available: http://www.tutorialspoint.com/sql/second-normal-form.htm. [Accessed 22 July 2011].

[88] F. Spillers, "How Usable is Jakob Nielsen?," 7 Apr 2004. [Online]. Available: http://experiencedynamics.blogs.com/site_search_usability/2004/04/how_usable_is_j.html. [Accessed 21 July 2011].

[89] The PHP Group, "Session Handling," 20 July 2011. [Online]. Available: http://www.php.net/manual/en/book.session.php. [Accessed 31 July 2011].

[90] "Documentation of jQuery," 2010. [Online]. Available: http://docs.jquery.com/Main_Page. [Accessed 1 Aug 2011].

[91] T. Chahine, "Low-fidelity prototyping – an overview of tools," 8 May 2011. [Online]. Available: http://aixd2011.com/2011/05/08/low-fidelity-prototyping-an-overview-of-tools/. [Accessed 30 June 2011].

[92] G. Booch, J. Rumbaugh and I. Jacobson, The Unified Modeling Language User Guide, London: Addison-Wesley, 2005.

[93] T. M. Connolly, Database systems : a practical approach to design, implementation, and management, Harlow: Addison-Wesley, 2002.

Appendices

A. Key source code

The entire code is available at project library in Department of Computer Science of the University of York, also available at http://s-yu.org/project/ . A demo user account was offered for evaluation at the online site:

User e-mail address (log in): demo@cs.york.ac.uk

Password: demo

D.1 Libraries

e. general.lib.php

```
<?php
//function of general library

//connect to database as read-only account
function connectdb_read()
{
	global $db_server;
	global $db_name;
	global $db_username_read;
	global $db_password_read;
	global $db_username_write;
	global $db_password_write;

	// Connect to the database for read-only access
	if (@mysql_connect("$db_server", "$db_username_read",
		"$db_password_read"))
		if ($result=@mysql_select_db("$db_name"))
			return TRUE;

	return FALSE;
}

//connect to database as write account
function connectdb_rw()
{
	global $db_server;
	global $db_name;
	global $db_username_read;
	global $db_password_read;
	global $db_username_write;
	global $db_password_write;

	// Connect to the database for read and write access
	if (@mysql_connect("$db_server", "$db_username_write",
		"$db_password_write"))
		if (@mysql_select_db("$db_name"))
```

```
                    return TRUE;

        return FALSE;
}

//format inputed data for SQL command use(), if $lower_case==1 convert to
lower case
function format_sql($data, $lower_case=0){
        if (!get_magic_quotes_gpc()){
                $data=addslashes($data);
        }
        $data=trim($data);
        if ($lower_case==1){
                $data=strtolower($data);
        }
        return $data;
}

//format data for HTML display
function format_html($data){
        return htmlspecialchars($data);
}

//count array size. If=0, return false, if>1, return true
function array_isset($array){
        if (count($array)>=1){
                return true;
        }
        else {
                return false;
        }
}

//send email
function send_email($to, $subject, $message) {
        //*******************2 STEP VALIDATION
        global $administrator_email;
        mail($to, $subject, $message, "From: ".$administrator_email);
}

//deal with system errors
function error($message) {
        global $error_message;
        $error_message = $message;
        global $administrator_email;
        send_email($administrator_email, 'System Error!', $message);
        goto_url("error.php?e=$error_message");
}

//deal with successful information
function success($message) {
        goto_url("success.php?s=$message");
}

//page redirector
function goto_url($url) {
        header("Location: $url");
}
```

```
//connect and query database, return 2D array for SELECT query
function db_query($query) {

	//use read account for SELECT query and return data set
	if (substr($query, 0, 6) == 'SELECT') {
		if (connectdb_read()) {
			$db_result = mysql_query($query);
		    $i=0;
			$data_array = array();
			while($data_array_line = mysql_fetch_array($db_result))
{
				$data_array[$i] = $data_array_line;
				$i++;
		}
			return $data_array;
		}
		else {
		      error('database connection error (r) ' . mysql_error());
		}
	}

	// use read and write account for other query without return
	else {
		if (connectdb_rw()) {
			mysql_query($query);
		}
		else {
			error('database connection error (rw) ' .
mysql_error());
		}
	}

}

//calcuate days difference between today and a future date
function days_diff($date) {
	$time = strtotime($date);
	if ($time == FALSE) {
		error('incorrect date format');
		return FALSE;
	}
	else {
		$days_diff = ($time - time() + 86400) / 86400;
		$days_diff_out = floor($days_diff);
		return $days_diff_out;
	}
}

//convert text month to 2 digits number month
function month_dig($month_text) {
	$month_number = 0;
	switch ($month_text) {
		case "Jan":
			$month_number = 1;
			break;
		case "Feb":
			$month_number = 2;
			break;
		case "Mar":
```

```
                $month_number = 3;
                break;
        case "Apr":
                $month_number = 4;
                break;
        case "May":
                $month_number = 5;
                break;
        case "Jun":
                $month_number = 6;
                break;
        case "June":
                $month_number = 6;
                break;
        case "Jul":
                $month_number = 6;
                break;
        case "July":
                $month_number = 7;
                break;
        case "Aug":
                $month_number = 8;
                break;
        case "Sep":
                $month_number = 9;
                break;
        case "Sept":
                $month_number = 9;
                break;
        case "Oct":
                $month_number = 10;
                break;
        case "Nov":
                $month_number = 11;
                break;
        case "Dec":
                $month_number = 12;
                break;
    }
    return $month_number;
}

// validate name text
function validate_name ($text) {
    $result = preg_replace ('/[a-zA-Z\s\'\.]*/isU', '', $text);
    if (strlen($result) > 0) {
        return FALSE;
    }
    elseif (strlen($result) == 0) {
        return TRUE;
    }
}

// convert DB style time to UK style time
function time_uk($time_db) {
    $time_d = date('j', strtotime($time_db));
    $time_m = date('M', strtotime($time_db));
    $time_y = date('Y', strtotime($time_db));
    $time_h = date('G', strtotime($time_db));
```

```
	$time_min = date('i', strtotime($time_db));
	$time_s = date('s', strtotime($time_db));
	if ($time_m == 'Jun') {
		$time_m= 'June';
	}
	if ($time_m == 'Jul') {
		$time_m= 'July';
	}
	$time_uk=$time_h.':'.$time_min.', '.$time_d.' '.$time_m.' '.$time_y;
	return $time_uk;
}

// convert DB style time to UK style date
function date_uk($time_db) {
	$time_d = date('j', strtotime($time_db));
	$time_m = date('M', strtotime($time_db));
	$time_y = date('Y', strtotime($time_db));
	$time_h = date('G', strtotime($time_db));
	$time_min = date('i', strtotime($time_db));
	$time_s = date('s', strtotime($time_db));
	if ($time_m == 'Jun') {
		$time_m= 'June';
	}
	if ($time_m == 'Jul') {
		$time_m= 'July';
	}
	$time_uk=$time_d.' '.$time_m.' '.$time_y;
	return $time_uk;
}

// show this time
function time_this() {
	$time_d = date('j', time());
	$time_m = date('M', time());
	$time_y = date('Y', time());
	$time_h = date('G', time());
	$time_min = date('i', time());
	$time_s = date('s', time());
	if ($time_m == 'Jun') {
		$time_m= 'June';
	}
	if ($time_m == 'Jul') {
		$time_m= 'July';
	}
	$time_this=$time_d.' '.$time_m.' '.$time_y.'
'.$time_h.':'.$time_min.':'.$time_s;
	return $time_this;
}

// log out
function logout() {
	setcookie("id", "", 0, '/');
	setcookie("password", "", 0, '/');
	session_start();
	unset($_SESSION ["user"]);
	unset($_SESSION);
	session_unset();
	session_destroy();
}
```

```
// file name of visiting
function file_name(){
	$url = $_SERVER['PHP_SELF'];
	$filename= substr( $url , strrpos($url , '/')+1 );
	return $filename;
}

// convert UK style time to DB style time
function time_db($time_uk) {
	$time_db = date('Y-m-d H:i:s', strtotime($time_uk));
	return $time_db;
}

// convert file size unit from byte to B, KB, MB, GB, TB, PB by base-1024
function file_size_convert($size) {
   $unit=array('Byte','KB','MB','GB','TB','PB');
   return
@round($size/pow(1024,($i=floor(log( abs($size==0?1:$size) ,1024)))),2).'
'.$unit[$i];
}

//last page
function last_page(){
	if (isset($_SERVER['HTTP_REFERER'])){
		return $_SERVER['HTTP_REFERER'];
	}
	else{
		global $after_login_redirect;
		return $after_login_redirect;
	}
}

//compare radio button value and output checked or not
function checked ($option_this, $option_value) {
	if ($option_this == $option_value) {
		return "checked=\"checked\"";
	}
	else {
		return '';
	}
}

//code translation of operations
function operation_code ($op) {
	$name='';
	switch ($op) {
		case 0:
			$name= "No Change";
			break;
		case 1:
			$name= "Create";
			break;
		case 2:
			$name= "Delete";
			break;
		case 3:
			$name= "Update";
			break;
```

```
                case 4:
                        $name= "Move";
                        break;
                default:
                        $name= "Undefined";
        }
        return $name;
}

?>
```

f. identify.inc.php

```
<?php
//start session
session_start();

//if user session exist
if (!empty($_SESSION ["user"] ["uid"])){

}
//if user session not exist, but cookie exist
elseif (!empty($_COOKIE['id'])){
        $_SESSION ["system"] ["login_from"]=$_SERVER["REQUEST_URI"];
        header("Location: login_check.php?cookie=1&");
}
//if no session and no cookie
elseif (file_name()!="login.php") {
        header("Location: login.php");
}
?>
```

g. initial.inc.php

```
<?php
include 'config.inc.php';
include 'libraries/general.lib.php';
include 'libraries/identify.inc.php';
?>
```

h. login_check.lib.php

```
<?php
//receive login details from post data
function receive($var){
        if ($var=='id' && !empty($_POST['id'])){
                return format_sql($_POST['id'], 1);
        }
        elseif ($var=='password' && !empty($_POST['password'])){
                return format_sql($_POST['password'], 0);
```

```
	}
	elseif ($var=='remember'){
		if (!empty($_POST['remember'])){
			return true;
		}
		else {
			return false;
		}
	}
	else {
		login_return();
	}
}

//if login fail, return back to login page
function login_return(){
	if (isset($_POST['id'])){
		$url="login.php?error=1&u=".$_POST['id'];
	}
	elseif (!isset($_POST['id'])){
		$url="login.php?error=1";
	}
	header("Location: $url");
}

//check login details in DB
//write login record in DB
//set cookie
function cookie_set($name, $value){
	global $cookie_valid;
	setcookie($name, $value, time()+($cookie_valid*3600*24), '/');
}

//check cookie

?>
```

i. login.lib.php

```
<?php
//function of dealing with login.php

//if login has been returned when error, high light error field by pre-
defined css tag
function login_error_hl() {
	if (isset ( $_GET ['error'] )) {
		echo ' error';
	}
}
//if login has been returned when ID error, show comment under ID field
function login_id() {
	if (isset ( $_GET ['error'] )) {
		echo '<br /><span class="help-inline">Please check your ID.
Maybe your e-mail address.</span>';
	}
	elseif (!isset ( $_GET ['error'] )) {
```

```
            echo '<br /><span class="help-inline">Maybe your e-mail
address.</span>';
      }
}
//if login has been returned when password error, show comment under
password field
function login_pw() {
      if (isset ( $_GET ['error'] )) {
            echo '<br /><span class="help-inline">Please check your
password.</span>';
      }
}
//highlight error ID
function default_id() {
      if (isset ($_GET['u'])){
            echo 'value="'.$_GET['u'].'" ';
      }
}

?>
```

j. project.lib.php

```
<?php
//**limit UID of login when view task

//get logged in user
function user_logged_in(){
      if (isset($_SESSION ["user"] ["uid"])){
            return $_SESSION ["user"] ["uid"];
      }
      else {
            return FALSE;
      }
}

//function of load dir_changes information from database
function fetch_user($uid){
      $sql='SELECT * FROM user WHERE uid='.$uid;
      $db_array=db_query($sql);
      if (!array_isset($db_array)){
            return FALSE;
      }
      else{
            return $db_array;
      }
}

//function of list tasks by project ID
function fetch_user_task ($uid){
      $sql='SELECT * FROM task WHERE uid='.$uid.' ORDER BY pid DESC';
      $db_array=db_query($sql);
      if (!array_isset($db_array)){
            return FALSE;
      }
      else{
```

```
		return $db_array;
	}
}

//function of load project information from database by project ID
function fetch_project($pid){
	$sql='SELECT * FROM project WHERE pid='.$pid.' ORDER BY pid DESC';
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		return $db_array;
	}
}

//function of list tasks by project ID
function fetch_project_task ($pid){
	$sql='SELECT * FROM task WHERE pid='.$pid.' ORDER BY pid DESC';
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		return $db_array;
	}
}

//function of list task history by project ID
function fetch_project_task_history ($pid){
	$sql='SELECT tid FROM task WHERE pid='.$pid.' ORDER BY tid DESC';
	$db_array_tid=db_query($sql);
	if (!array_isset($db_array_tid)){
		return FALSE;
	}
	else{
		$task_count=count($db_array_tid);
		for ($i = 0; $i < $task_count; $i++) {

	$db_array_task=fetch_task_history($db_array_tid[$i]['tid']);
			$db_array_task_history[$i]=$db_array_task[0];
		}
		return $db_array_task_history;
	}
}

//function of load directory information from database by directory ID
function fetch_directory($did){
	$sql='SELECT * FROM directory WHERE did='.$did.' ORDER BY did DESC';
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		return $db_array;
	}
}

//function of load dir_changes information from database by directory ID
```

```
function fetch_directory_change($did){
	$sql='SELECT * FROM directory_change WHERE did='.$did.' ORDER BY
time DESC';
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		return $db_array;
	}
}

//function of combine a full directory from database by directory ID
function fetch_directory_full($did){
	if ($did==NULL){
		return "/";
	}
	elseif (fetch_directory_change($did)==FALSE){
		return FALSE;
	}
	else{
		$db_array=fetch_directory_change($did);
		$directory_name=$db_array[0]['name'];
		$directroy_base_id=$db_array[0]['did_base'];
		if ($directroy_base_id==NULL){
			return "/".$directory_name."/";
		}
		elseif ($directroy_base_id!=NULL){
			return
$directory_full=fetch_directory_full($directroy_base_id).$directory_name."/
";
		}
	}
}

//function of read file information from database by file ID
function fetch_file($fid){
	$sql='SELECT * FROM file WHERE fid='.$fid.' ORDER BY fid DESC';
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		return $db_array;
	}
}

//function of read file_changes information from database by file ID
function fetch_file_change($fid){
	$sql='SELECT * FROM file_change WHERE fid='.$fid.' ORDER BY time
DESC';
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		return $db_array;
	}
}
```

```
//function of search task_histroy information from database by keywords
function fetch_task_history_search($s){
	$sql='SELECT tid FROM task WHERE name LIKE "%'.$s.'%" ORDER BY tid
DESC';
	$db_array_tid=db_query($sql);
	if (!array_isset($db_array_tid)){
		return FALSE;
	}
	else{
		$task_count=count($db_array_tid);
		for ($i = 0; $i < $task_count; $i++) {

	$db_array_task=fetch_task_history($db_array_tid[$i]['tid']);
			$db_array_task_history[$i]=$db_array_task[0];
		}
		return $db_array_task_history;
	}
}

//function of search task information from database by keywords
function fetch_task_search($s){
	$sql='SELECT * FROM task WHERE name LIKE "%'.$s.'%" ORDER BY tid
DESC';
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		return $db_array;
	}
}

//function of read task_histroy information from database by task history
ID
function fetch_task_history($tid){
	$sql='SELECT * FROM task_history WHERE tid='.$tid.' ORDER BY time
DESC';
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		return $db_array;
	}
}

//function of read task_histroy information from database by task ID
function fetch_task($tid){
	$sql='SELECT * FROM task WHERE tid='.$tid.' ORDER BY end DESC';
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		return $db_array;
	}
}
```

```
//function of list tasks by project ID
//**limit UID of login at this SQL statement
function fetch_task_project ($tid){
	$sql='SELECT pid FROM task WHERE tid='.$tid.' ORDER BY tid DESC';
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		$sql_p='SELECT * FROM project WHERE
pid='.$db_array[0]['pid'].' ORDER BY pid DESC';
		$db_array_p=db_query($sql_p);
		if (!array_isset($db_array_p)){
			return FALSE;
		}
		else {
			return $db_array_p;
		}
	}
}

//function of identify task finished or not from database by task ID
function fetch_task_finished($tid){
	$sql='SELECT * FROM task WHERE tid='.$tid.' ORDER BY end DESC';
	$db_array_task=db_query($sql);
	$sql2='SELECT * FROM task_history WHERE tid='.$tid.' ORDER BY
version DESC';
	$db_array_task_history=db_query($sql2);
	if (isset($db_array_task[0]['end']) and
strtotime($db_array_task[0]['end'])<time()){
		return TRUE;
	}
	elseif (isset($db_array_task_history[0]['percent']) and
$db_array_task_history[0]['percent']>=100){
		return TRUE;
	}
	else{
		return FALSE;
	}
}

//function of task expired or not
function fetch_task_ready ($tid){
	$sql='SELECT * FROM task WHERE tid='.$tid.' ORDER BY end DESC';
	$db_array_task=db_query($sql);
	$sql2='SELECT * FROM task_history WHERE tid='.$tid.' ORDER BY
version DESC';
	$db_array_task_history=db_query($sql2);
	if (isset($db_array_task[0]['start']) and
strtotime($db_array_task[0]['start'])<time() and
count($db_array_task_history)<1){
		return TRUE;
	}
	else{
		return FALSE;
	}
}

//function of task expired or not
```

```
function fetch_task_doing ($tid){
	$sql='SELECT * FROM task WHERE tid='.$tid.' ORDER BY end DESC';
	$db_array_task=db_query($sql);
	$sql2='SELECT * FROM task_history WHERE tid='.$tid.' ORDER BY
version DESC';
	$db_array_task_history=db_query($sql2);
	if (isset($db_array_task[0]['start']) and
strtotime($db_array_task[0]['start'])<time() and
count($db_array_task_history)>=1){
		if ($db_array_task_history[0]['percent']<100){
			return TRUE;
		}
		else {
			return FALSE;
		}
	}
	else{
		return FALSE;
	}
}

//function of read task's predecessor task's ID from database by task ID
function fetch_task_predecessor_tid($tid){
	$sql='SELECT * FROM task WHERE tid='.$tid.' ORDER BY end DESC';
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		if ($db_array[0]['predecessor']==NULL){
			return FALSE;
		}
		else{
			$tid_p=$db_array[0]['predecessor'];
			return $tid_p;
		}
	}
}

/////////function of identify task of waiting predecessor or not from
database by task ID
function fetch_task_predecessor_wait($tid){
	if (fetch_task_predecessor_tid($tid)){
		$tid_p=fetch_task_predecessor_tid($tid);
		if (fetch_task_finished($tid_p)){
			return FALSE;
		}
		elseif (!fetch_task_finished($tid_p)){
			return TRUE;
		}
	}
	else {
		return FALSE;
	}
}

//function of read task's predecessor task's name from database by task ID
function fetch_task_predecessor_name($tid){
	$sql='SELECT * FROM task WHERE tid='.$tid.' ORDER BY end DESC';
```

```
	$db_array=db_query($sql);
	if (!array_isset($db_array)){
		return FALSE;
	}
	else{
		if ($db_array[0]['predecessor']==NULL){
			return FALSE;
		}
		else{
			$tid_p=$db_array[0]['predecessor'];
			$db_array=fetch_task($tid_p);
			return $db_array[0]['name'];
		}
	}
}

//function of list files by task ID
function fetch_task_file ($tid){
	$sql='SELECT fid FROM file WHERE tid='.$tid.' ORDER BY fid DESC';
	$db_array_fid=db_query($sql);
	if (!array_isset($db_array_fid)){
		return FALSE;
	}
	else{
		$file_count=count($db_array_fid);
		for ($i = 0; $i < $file_count; $i++) {

	$db_array_file=fetch_file_change($db_array_fid[$i]['fid']);
			$db_array_file_change[$i]=$db_array_file[0];
		}
		return $db_array_file_change;
	}
}

//function of list directories by task ID
function fetch_task_directory ($tid){
	$sql='SELECT did FROM directory WHERE tid='.$tid.' ORDER BY did
DESC';
	$db_array_did=db_query($sql);
	if (!array_isset($db_array_did)){
		return FALSE;
	}
	else{
		$directory_count=count($db_array_did);
		for ($i = 0; $i < $directory_count; $i++) {

	$db_array_directory=fetch_directory_change($db_array_did[$i]['did']);
			$db_array_directory_change[$i]=$db_array_directory[0];
		}
		return $db_array_directory_change;
	}
}

//function of task_status
function task_status ($tid){
	$status='';
	$db_array_task=fetch_task($tid);
	$db_array_task_history=fetch_task_history($tid);
```

```
	if (isset($db_array_task[0]['start']) and
strtotime($db_array_task[0]['start'])>time()){
		$status.='Not yet start. ';
	}
	if (isset($db_array_task_history[0]['version']) and
count($db_array_task_history)>=1){
		$status.='Ver '.$db_array_task_history[0]['version'].',
('.$db_array_task_history[0]['percent'].'%) ';
	}
	if (isset($db_array_task[0]['end']) and
strtotime($db_array_task[0]['end'])<time()){
		$status.='Expired. ';
	}
	if (fetch_task_predecessor_wait($tid)){
		$status.='Waiting for predecessor. ';
	}
	return $status;
}

//function of delete project in database by project ID
function delete_project($pid){
	$sql='DELETE FROM project WHERE pid='.$pid;
	$db_array=db_query($sql);
}

//function of delete task in database by task ID
function delete_task($tid){
	$sql='DELETE FROM task WHERE tid='.$tid;
	$db_array=db_query($sql);
}

//function of delete file in database by file ID
function delete_file($fid){
	$sql='DELETE FROM file WHERE fid='.$fid;
	$db_array=db_query($sql);
}

//function of delete directory in database by directory ID
function delete_directory($did){
	$sql='DELETE FROM directory WHERE did='.$did;
	$db_array=db_query($sql);
}

//function of create task in database
function create_task($pid, $uid, $name, $priority, $start, $end,
$description){
	$sql="INSERT INTO task (pid, uid, name, priority, start, end,
description) VALUES ($pid, $uid, '$name', $priority, '$start', '$end',
'$description')";
	$db_array=db_query($sql);
	$tid=mysql_insert_id();
	$time=time_db(time_this());
	$ip=$_SERVER['REMOTE_ADDR'];
     $sql="INSERT INTO task_history (tid, uid, version, percent, time, ip,
description) VALUES ($tid, $uid, 1, 0, '$time', '$description')";
	$db_array=db_query($sql);
	return $tid;
}
```

```
//function of create task in database.
function create_file($uid, $t, $name, $directory, $fid, $description){
	$sql="INSERT INTO file (tid, description) VALUES ($t,
'$description')";
	$db_array=db_query($sql);
	$fid=mysql_insert_id();
	$time=time_db(time_this());
	$ip=$_SERVER['REMOTE_ADDR'];
	$sql="INSERT INTO file_change (tid, uid, version, time, ip,
description) VALUES ($t, $uid, 1, '$time', '$ip', '$description')";
	$db_array=db_query($sql);
	return $fid;
}

?>
```

k. style.lib.php

```
<?php
//function of dealing with style-based code

//set icon image address
function icon_address(){
	$base='images/icon/';
	switch (file_name()) {
		case "summary.php":
			$name= "summary.gif";
			break;
		case "project.php":
			$name= "project.gif";
			break;
		case "project_list.php":
			$name= "project_list.gif";
			break;
		case "task.php":
			$name= "task.gif";
			break;
		case "file.php":
			$name= "file.gif";
			break;
		case "directory.php":
			$name= "directory.gif";
			break;
		case "search.php":
			$name= "search.gif";
			break;
		default:
			$name= "default.gif";
	}
	return $base.$name;
}

//set icon image alter message
function icon_alt(){
	switch (file_name()) {
		case "summary.php":
```

```
                $alt= "Summary";
                break;
            case "project.php":
                $alt= "Project";
                break;
            case "project_list.php":
                $alt= "Project List";
                break;
            case "task.php":
                $alt= "Task";
                break;
            case "file.php":
                $alt= "File";
                break;
            case "directory.php":
                $alt= "Directory";
                break;
            case "search.php":
                $alt= "Search";
                break;
            default:
                $alt= "Welcome";
        }
        return $alt;
}

//set top bar button status to active by get value
function active($active_string) {
        $name=file_name();
        if ($active_string=="private" and isset($_GET['private'])) {
                echo ' class="active"';
        }
        elseif ($active_string == $name) {
                echo ' class="active"';
        }
        elseif (($name=="project.php" or $name=="task.php" or
$name=="file.php" or $name=="directory.php" or $name=="project_list.php")
and $active_string == "project") {
                echo ' class="active"';
        }
}

//topbar right align name display(login or logout)
function right_align(){
        if (isset($_SESSION ["user"] ["uid"])){
                echo '<a href="login_check.php?logout=1"><strong>'.$_SESSION
["user"] ["name_nickname"].'</strong>  Sign Out</a>';
        }
        else{
                echo '<a href="login.php">Sign In</a>';
        }
}

//set html title, page title, page sub title by category value
function page_title($title_cat){
        global $system_name;
        global $system_name_short;
        global $system_name_version;
        global $page_title;
```

```
	global $page_title_sub;
	$output='';
	if ($title_cat=='html'){
		if (isset($page_title) && isset($page_title_sub)){
			$output=$page_title.' - '.$page_title_sub;
		}
		elseif (isset($_GET['f'])){
			$output='file name - task name';
		}
		elseif (isset($_GET['d'])){
			$output='directory name - task name';
		}
		elseif (isset($_GET['t'])){
			$output='task name - project name';
		}
		elseif (isset($_GET['p'])){
			$output='project name by due date';
		}
		else{
			$output=$system_name;
		}
	}
	elseif ($title_cat=='main'){
		if (isset($page_title)){
			$output=$page_title;
		}
		elseif (isset($_GET['f'])){
			$output='file name';
		}
		elseif (isset($_GET['d'])){
			$output='directory name';
		}
		elseif (isset($_GET['t'])){
			$output='task name';
		}
		elseif (isset($_GET['p'])){
			$output='project name';
		}
		else{
			$output=$system_name;
		}
	}
	elseif ($title_cat=='sub'){
		if (isset($page_title_sub)){
			$output=$page_title_sub;
		}
		elseif (isset($_GET['f'])){
			$output='task name';
		}
		elseif (isset($_GET['d'])){
			$output='task name';
		}
		elseif (isset($_GET['t'])){
			$output='project';
		}
		elseif (isset($_GET['p'])){
			$output='by due date';
		}
		else{
```

```
                        $output=$system_name_short;
                }

        }
        else {
                $output='error';
        }
        echo $output;
}
?>
```

D.2 Styles

l. common.css

```
@CHARSET "UTF-8";
body {
        padding-top: 52px;
}
.footer {
        margin-top: 12px;
        padding-top: 8px;
        padding-left: 12px;
        background-color: #f7f7f7;
}
.footer a{
color: #555588;
}
.login {
        margin-top: 24px;
}
.td_link {
        cursor:pointer;
}
table .black {
        color: #333333;
}
.main_content{
        min-height:400px;
}
.underline{

}
.float_left{
        float:left;
}
.float_right{
        float:right;
}
.page_icon{
        height:64px;
        weith:64px;
        border:0px;
}
.input-medium_small,
```

```
input.medium_small,
textarea.medium_small,
select.medium_small {
  width: 120px;
}
```

m. header.inc.php

```
<?php
include 'libraries/style.lib.php';
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<!--header-->
<html xmlns="http://www.w3.org/1999/xhtml">
	<head>
		<meta http-equiv="Content-Language" content="en" />
		<meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
		<meta name="keywords" content="Version Control, Cloud, Task-
oriented, Project, Teamwork" />
		<meta name="author" content="Sheng Yu, sy595@cs.york.ac.uk" />
		<meta name="description" content="WVCS 1.0; Working in the
Cloud: Web-based Version Control System for Task-oriented Group and
Individual Projects">
		<link rel="stylesheet" type="text/css" href="style/bootstrap-
1.2.0.css" />
		<link rel="stylesheet" type="text/css" href="style/common.css"
/>
		<script type="text/javascript" src="style/jquery-
1.6.2.min.js"></script>
		<script type="text/javascript"
src="style/jquery.tablesorter.min.js"></script>
		<!-- Le HTML5 shim, for IE6-8 support of HTML elements -->
		<!--[if lt IE 9]>
			<script src="style/html5.js"></script>
		<![endif]-->
		<title><?php page_title('html');?></title>
	</head>
	<body>
		<div class="topbar">
			<div class="fill">
				<div class="container">
					<h3><a href="./"><?php echo
$system_name_short.' '.$system_version;?></a></h3>
					<ul>
						<li<?php active('summary.php');?>><a
href="summary.php">Summary</a></li>
						<li<?php active('project');?>><a
href="<?php echo $after_login_redirect;?>">Group Projects</a></li>
					</ul>
					<form action="task_search.php"
method="get">
						<input type="text" name="s"
placeholder="search tasks">
```

```
                                    </form>
                                    <ul class="nav secondary-nav">
                                          <li<?php active('private');?>><a
href="project.php?private=1">My Private Projects</a></li>
                                          <li><?php right_align();?></li>
                                    </ul>
                              </div><!-- /container -->
                        </div><!-- /fill -->
                  </div><!-- /topbar -->
                  <div class="container main_content">
                  <div class="page_icon float_right">
                  <img class="page_icon" src="<?php echo icon_address (); ?>"
alt="<?php echo icon_alt (); ?>" />
                  </div>
                  <div class="page-header">
                        <h1><?php page_title('main');?> <small><?php
page_title('sub');?></small></h1>
                  </div>
                  <!--/header-->
```

n. footer.inc.php

```
                  <!--footer-->
                  </div><!-- /container -->
                  <div class="container footer">
                        <p>
                              Designed and built with all the love in the world
at <a href="http://www.cs.york.ac.uk/" target="_blank">Department of
Computer Science</a>, <a href="http://www.york.ac.uk/" target="_blank">The
University of York</a> by <a href="mailto: sy595@cs.york.ac.uk"
target="_blank">Sheng Yu</a>.<br>
                              Powered by <a href="http://jquery.com/"
target="_blank">jQuery</a>, <a href="https://github.com/twitter/bootstrap"
target="_blank">Bootstrap</a>, <a
href="http://code.google.com/p/html5shim/" target="_blank">html5shim</a>,
<a href="http://code.google.com/p/google-code-prettify/" target="_blank">

                  google-code-prettify</a>, <a
href="http://tablesorter.com/docs/" target="_blank">tablesorter</a>.
Licensed under the <a href="http://www.apache.org/licenses/LICENSE-2.0"
target="_blank">Apache License v2.0</a>.
                        </p>
                  </div>
            </body>
</html>
<!--/footer-->
```

D.3 Controller pages

o. *config.inc.php*

```
<?php
//database connection information
$db_server = 'localhost';
$db_name = 'wvcs';
$db_username_read = 'root';
$db_password_read = '';
$db_username_write = 'root';
$db_password_write = '';

//email of administrator, use for receive error message, and for "from"
label on email sends to user
$administrator_email = 'sy595@cs.york.ac.uk';

//system name
$system_name = 'Web-based Version Control System';
$system_name_short = 'WVCS';
$system_version = '1.0';

//login by: email/uid/name_nickname
$login_by = "email";

//redirect to page after successful login
$after_login_redirect = "summary.php";

//cookie valid days
$cookie_valid = 60;

//default time zone
//location strings can be found at
http://www.php.net/manual/en/timezones.php
date_default_timezone_set("Europe/London");

//Terms and Conditions (HTML tags needed)
$terms_conditions = '
<p><b>WVCS Terms and Conditions</b></p>
<p>According to Data Protection Act 1998, your personal data will be well
protected and will not be passed to third party.</p>
';

?>
```

p. *login.php*

```
<?php
include 'libraries/initial.inc.php';
include 'libraries/login.lib.php';
include 'style/header.inc.php';
?>
<div class="row login">
<div class="span4 columns">Welcome back!</div>
<div class="span12 columns">
```

```
<form action="login_check.php" method="post">
<fieldset><legend>Sign in <?php echo $system_name_short; ?></legend>
<div class="clearfix<?php login_error_hl(); ?>">
<label for="xlInput"><?php echo $system_name_short; ?> ID</label>
<div class="input"><input <?php default_id(); ?>class="xlarge" id="id"
name="id" size="50" type="text" /><?php login_id (); ?>
</div>
</div>
<!-- /clearfix -->
<div class="clearfix<?php login_error_hl (); ?>">
<label for="xlInput">Password</label>
<div class="input">
<input class="xlarge" id="password" name="password" size="30"
type="password" /><?php login_pw (); ?>
</div>
</div>
<!-- /clearfix -->
<div class="clearfix"><label id="optionsRadio"></label>
<div class="input">
<ul class="inputs-list">
       <li><label> <input type="checkbox" name="remember" value="1">
<span>Remember me</span>
          <a href="mailto: <?php echo
$administrator_email; ?>">Forgot login details?</a> </label></li>
</ul>
</div>
</div>
<!-- /clearfix -->
<div class="actions">
<button type="submit" class="btn large primary">Sign in</button>
 
<button type="reset" class="btn large"
onclick="location.href='login.php'">Cancel</button>
</div>
</fieldset>
</form>
</div>
</div>
<?php
include 'style/footer.inc.php';
?>
```

q. login_check.php

```
<?php
include 'config.inc.php';
include 'libraries/general.lib.php';
include 'libraries/login_check.lib.php';
$logout=0;
//if redirect from log out button
if (isset($_GET['logout'])){
       logout();
       header("Location: login.php");
       $logout=1;
}
//if redirect from cookie detecter
```

```
elseif (isset($_GET['cookie'])){
	if (!empty($_COOKIE['id'])){
		$id=$_COOKIE['id'];
		$password=$_COOKIE['password'];

	}
	else {
		header("Location: login.php");
	}
}
else{
	$id=receive('id');
	$password=md5(receive('password'));//md5
}
if ($logout==0){
	$sql='SELECT * FROM user WHERE '.$login_by.'="'.$id.'" and
password="'.$password.'"';
	$db_array=db_query($sql);

	//if login fail
	if (!array_isset($db_array)){
		logout();
		login_return();
	}
	//if login OK
	else{
		//set session for login
		session_start();
		$_SESSION ["user"] ["uid"] = $db_array[0]['uid'];
		$_SESSION ["user"] ["title"] = $db_array[0]['title'];
		$_SESSION ["user"] ["name_first"] = $db_array[0]['name_first'];
		$_SESSION ["user"] ["name_middle"] =
$db_array[0]['name_middle'];
		$_SESSION ["user"] ["name_last"] = $db_array[0]['name_last'];
		$_SESSION ["user"] ["name_nickname"] =
$db_array[0]['name_nickname'];
		$_SESSION ["user"] ["email"] = $db_array[0]['email'];
		$_SESSION ["user"] ["password"] = $db_array[0]['password'];
		$_SESSION ["user"] ["type"] = $db_array[0]['type'];

		//record user's ip and login time
		$sql='UPDATE user SET
lastlogin_ip="'.$_SERVER['REMOTE_ADDR'].'",
lastlogin_time="'.time_db(time_this()).'" WHERE '.$login_by.'="'.$id.'" and
password="'.$password.'"';
		db_query($sql);

		//if ticked "remember me" then set cookie for next auto-login
		if (receive('remember')){
			cookie_set('id', $id);
			cookie_set('password', $password);
		}
		//if login by cookie, back to original page
		if (isset($_GET['cookie'])){
			//back to the page before redirect to here by
HTTP_REFERER
			if (isset($_SERVER['HTTP_REFERER'])){
				$url=$_SERVER['HTTP_REFERER'];
			}
```

```
                        //back to the page before redirect to here by SESSION
                        elseif (isset($_SESSION ["system"] ["login_from"])){
                                $url=$_SESSION ["system"] ["login_from"];
                        }
                        else{
                                $url=$after_login_redirect;
                        }
                }
                //if login from input, back to pre-defined page
                else{
                        $url=$after_login_redirect;
                }
                header("Location: $url");
        }
}

?>


<?php
?>
```

r. file.php

```
<?php
include 'libraries/initial.inc.php';
include 'libraries/project.lib.php';
if(isset($_GET['f'])){
        $f=$_GET['f'];
}
else{
        $f=0;
}
if(fetch_file_change($f)==FALSE){
}
else{
        //file name, last commit time, page title, page sub title
        $db_array=fetch_file_change($f);
        $file_name=$db_array[0]['name'];
        $file_last_update=time_uk($db_array[0]['time']);
        $page_title=$file_name;
        $page_title_sub="last commit : [".$file_last_update."]";
        $file_change_number=count($db_array);
        if ($file_change_number<=1){
                $version_title="version";
        }
        else {
                $version_title="versions";
        }
}
include 'style/header.inc.php';
?>
<?php
if(fetch_file_change($f)==FALSE){
        ?>
        <div class="alert-message info">
```

```
        <a class="close" href="<?php echo $after_login_redirect; ?>">×</a>
        <p><strong>Oops!</strong> File not exist or have not any changes,
please create before use.</p>
    </div>
      <?php ;
}
else{

      //file change history list/table
      ?>
      <h3 class="underline"><?php echo ucfirst($version_title).' of
"'.$file_name;?>"  <small>(<?php echo $file_change_number."
".$version_title;?>)</small></h3>
      <ul class="tabs">
      <li class="active"><a href="file.php?f=<?php echo $f;?>">File
versions list</a></li>
      <li><a href="file_info.php?f=<?php echo $f;?>">Information</a></li>
      <li><a href="task_operation.php?t=<?php echo
'1';?>">Operations</a></li>
      </ul>
      <script type="text/javascript">
            $(document).ready(function()
                {
                    $("table#file_changes").tablesorter( {sortList:
[[0,1]]} );
                }
            );
      </script>
      <table id="file_changes" class="zebra-striped">
      <thead>
      <th class="yellow">Ver</th>
      <th class="red">Location & File Name</th>
      <th class="blue">Size</th>
      <th class="green">Operation</th>
      <th class="green">Time</th>
      <th class="purple">Description</th>
      </thead>
      <?php
      for ($i = 0; $i < $file_change_number; $i++) {
            $fcid=$db_array[$i]['fcid'];
            $version=$db_array[$i]['version'];
            $name=$db_array[$i]['name'];
            $directory=fetch_directory_full($db_array[$i]['did']);
            $size=file_size_convert($db_array[$i]['size']);
            $operation=$db_array[$i]['type'];
            $time=$db_array[$i]['time'];
            $description=$db_array[$i]['description'];
            echo '<tr class="td_link"
onclick="location.href=\'download.php?fc='.$fcid.'&fn='.$name.'\'"><td>';
            echo $version;
            echo '</td><td class="black">';
            echo $directory."<strong>".$name."</strong>";
            echo "</td><td>";
            echo $size;
            echo "</td><td>";
            echo operation_code($operation);
            echo "</td><td>";
            echo '['.time_uk($time).']';
            echo "</td><td>";
```

```
                echo $description;
                echo "</td></tr>";
        }
        echo '</table>';
}
?>


<?php
include 'style/footer.inc.php';

?>
```