

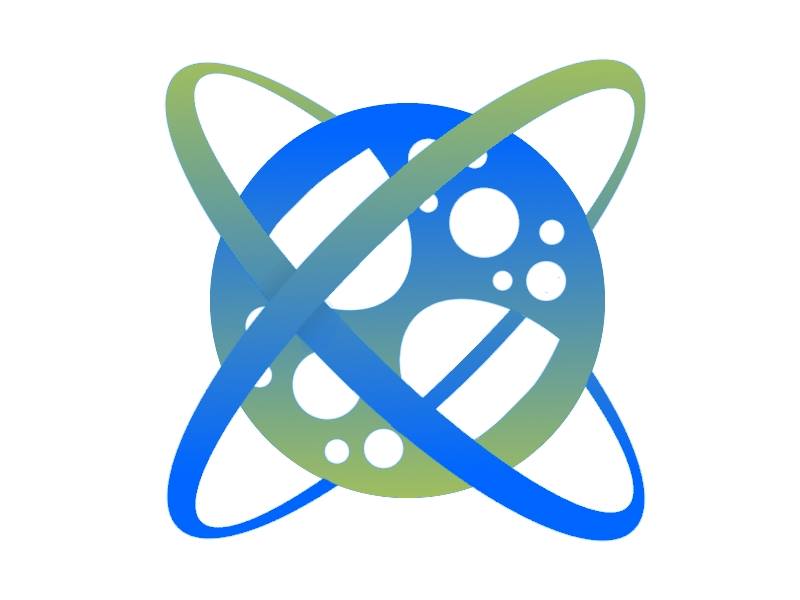
**Software Process, SOEN 341/4 S, Winter 2016**

Dr. Shang

Dr. Fancott

Mr. Morse

**TimeTurner** by team YAWD

****

**Project Scope and Plan Document – Deliverable 1**

|  |  |
| --- | --- |
| **Team members information** | |
| **Name** | **SID** |
| **Daniel Di Corpo** | **26331602** |
| **Dimitri Topaloglou** | **29358269** |
| **Claudia Della Serra** | **26766048** |
| **Philip Lim** | **27485506** |
| **Aline Koftikian** | **27764162** |
| **Ryan Lee** | **27752504** |
| **Erin Benderoff** | **27768478** |
| **Ideawin-Bunthy Koun** | **26314155** |
| **Kevin Yasmine** | **27195346** |
| **Marc-Andre Leclair** | **27754876** |
| **Lori Dalkin** | **27738293** |
| **Bryce Drewery-Schoeler** | **27283199** |

**Grading Sheet**

|  |  |  |
| --- | --- | --- |
| **Section** | **Evaluation criteria (see instructions in the template for details)** | **Grading** |
| all | 10 marks are allocated for excellence, professionalism and quality of work above and beyond the correct meeting of specifications.. | /10 |
| 1 | Presentation of this document | /5 |
| 2 | Completeness and accuracy with regard to initial project description | /1 |
| 3.1 .  .  3.2  3.3 | Completeness and accuracy of the project functional requirements expressed as formal use cases, including difficulty and importance indicators  completeness and accuracy of the diagram and description of the domain model  completeness and accuracy with regard to initial project description accuracy with regard to initial project description, difficulty and importance ratings | /15  .  /3 .  /1 |
| 4.1 | Description of all team members’ capacities and schedule restrictions | /1 |
| 5 | List of goals removed from the project.  For each goal removed, give justifications in light of the resources available | /`1 |
| 6.1 .  6.2 | Clarity of textual description, validity of rationale, clarity and appropriateness of diagram, list of modules responsibilities  List of technologies used, validity of rationale | /2 .  /1 |
| 7.1 .  7.2 .  7.3 .  7.4  7.5  7.6 | Completeness of list of activities, clarity of their stated purpose, as well as statement of what artifacts they are producing  Completeness of list of artifacts to be produced during the project, validity of roles description of each artifact  Cost estimation of each individual artifact, validity of explanation of cost estimation, total cost estimate  Mapping of activities to individual project members  Accurate and complete presentation of milestones  *Assessment of risks `* | /1 .  /2 .  /2 .  /1  /1  /1 |
| 8 | Early Prototyping | /2 |
| Total |  | /50 |

**DO NOT REMOVE THIS PAGE WHEN SUBMITTING YOUR DOCUMENT**

**TABLE OF CONTENTS**

[1. Introduction 5](#_Toc442097147)

[2. Project Description 5](#_Toc442097148)

[3. Goals and Constraints 6](#_Toc442097149)

[3.1 Functional Requirements 6](#_Toc442097150)

[3.2 Domain Model 6](#_Toc442097151)

[3.3 Constraints and Qualities 7](#_Toc442097152)

[3.3.1 Security 7](#_Toc442097153)

[3.3.2 Performance 7](#_Toc442097154)

[3.3.3 Cross-Browser Compatibility 8](#_Toc442097155)

[3.3.4 Ease of Use 8](#_Toc442097156)

[4. Resource Evaluation 8](#_Toc442097157)

[4.1 Human Resources 8](#_Toc442097158)

[4.2 Technical Resources 8](#_Toc442097159)

[5. Scoping 9](#_Toc442097160)

[6. Solution Sketch 9](#_Toc442097161)

[6.1 Architecture 9](#_Toc442097162)

[6.2 Technologies in Use 10](#_Toc442097163)

[6.2.1 Programming Languages 10](#_Toc442097164)

[6.2.1.1 Server-side 10](#_Toc442097165)

[6.2.1.2 Client Side 10](#_Toc442097166)

[6.2.2 IDEs 11](#_Toc442097167)

[6.2.2.1 PhpStorm 10 11](#_Toc442097168)

[6.2.3 Database Management System 11](#_Toc442097169)

[6.2.3.1 MySQL 11](#_Toc442097170)

[6.2.4 Web Server 11](#_Toc442097171)

[6.2.4.1 XAMPP (v1.8.1) 11](#_Toc442097172)

[6.2.5 Source Code Revision Management 11](#_Toc442097173)

[6.2.5.1 GitHub 11](#_Toc442097174)

[6.2.6 Deployment Software 11](#_Toc442097175)

[6.2.6.1 DeployHQ 11](#_Toc442097176)

[6.2.7 Team Collaboration 12](#_Toc442097177)

[6.2.7.1 Trello 12](#_Toc442097178)

[7. Plan 12](#_Toc442097179)

[7.1 Activities 12](#_Toc442097180)

[7.2 Artifacts 17](#_Toc442097181)

[7.3 Project Estimates 17](#_Toc442097182)

[7.4 Activities Assignments 17](#_Toc442097183)

[7.5 Schedule 20](#_Toc442097184)

[7.6 Risk 20](#_Toc442097185)

[8. Prototyping 20](#_Toc442097186)

# Introduction

Every year, newly admitted students at Concordia University are delegated a course sequence based on their matriculated program. Students are required create a schedule for themselves based on this course sequence and follow it thoroughly until the end of their degree. New students doing this on their own for the first time, and even experienced students, may encounter a number of difficulties when making schedules to fit this sequence, such as struggling to meet proper course prerequisites and managing time conflicts amongst multiple courses. This process is often long, tedious, and takes away from valuable time students may instead spend studying, working, or engaged in extracurricular activities. The following project, developed by a group of 12 undergraduate Software Engineering students at Concordia University, was created to simplify this process.

TimeTurner will be a web application implemented by Concordia University in order to aid students in planning and creating a schedule of courses for the duration of their degrees. Important factors to be kept in mind throughout the creation of this software system are ease of use, efficiency, speed, and overall effectiveness of the application in making scheduling of classes quick, easy, and personalized to students’ needs. Students will be given a sequence of classes recommended to them by their faculty, much the same as current students in the Faculty of Engineering and Computer Science are given at the start of their degrees. Based on this sequence and each student’s preferences, personalized schedules will be generated for the students’ entire degree, thus saving the students much time and energy.

# Project Description

The proposed and outlined web application, known as TimeTurner, is designed to auto-generate a student’s course sequence from their first semester up until the end of their degree. It takes into account user input preferences and and any previously completed courses or course prerequisites before creating this sequence. Preferences can be made by the student and include options such as night classes or having particular days off. The application will notify the user if a certain preference suggested results in an impossibility or conflict in the sequence. This sequence generator will be able to create a sequence at any point throughout the user’s degree, if sudden change in circumstances were to arise.

The goal of this application is to simplify the method with which students may decide and schedule their courses. If a course must be redone, the generator can decide what other courses should move where in regards to the remaining courses to be completed, which can be done in seconds, rather than hours. It saves the time of the user, in a simple and efficient manner. Ultimately, the system’s end goal will be to simplify a student’s task of creating their own course schedule in order to allow students to redirect their time to other more important activities, thus making course registration much simpler, quicker, and easier.

# Goals and Constraints

## Functional Requirements

This section should describe the requirements that the software shall have. It should define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs.

The overall functional requirements shall be presented graphically as a use case model.

Each individual requirement shall be presented as a use case. Also, each requirement should be rated in terms of difficulty and importance in order to provide useful discrimination factors for the project scoping presented in section 5.

## Domain Model

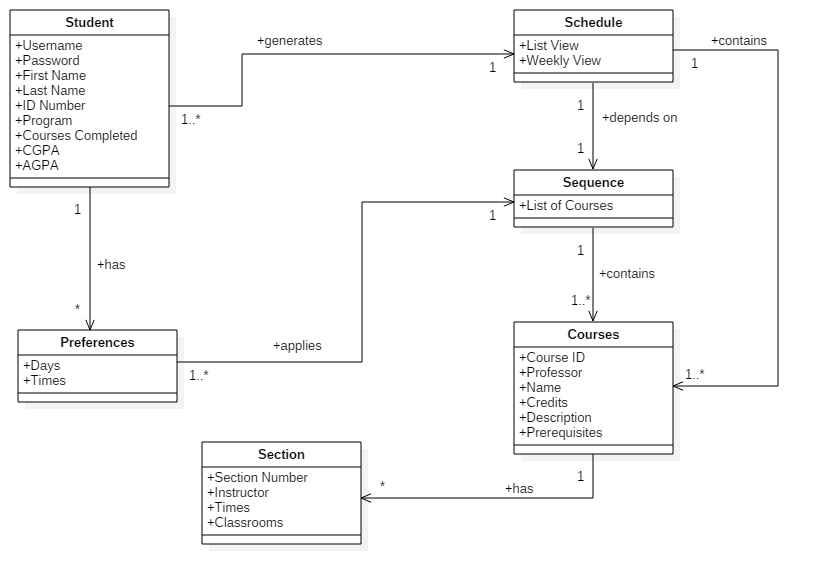


Figure 2. Domain Model

## Constraints and Qualities

### **Security**

Security is a crucial aspect of any software. One of the most important security requirements is to make sure there are no SQL injections that could breach and break our database. In order to assist in preventing SQL injections, we will be using prepared statements prior to query executing which is provided by PHP’s internal libraries. The possibility of having the database breached could potentially expose user personal user information or perhaps destroy content. The exposure of passwords is a concern in case of breach as they should be hashed and salted upon entry by using hashing algorithms such as *sha512* and *bcrypt*. This would essentially diminish any chance hackers have to brute force password entries.

### **Performance**

Performance is important especially when working with large data sets. The user must be able to navigate and use the site without having to wait for data generation. The application shall also ensure that database design is robust with the use of proper table indexing and storage engines and cache sizes. This will essentially allow for quicker SQL query executions that are expected to return data within 100 nanoseconds. The performance shall also account for larger data execution, such as the schedule generator with a performance factor speed of no more than 5 seconds.

### **Cross-Browser Compatibility**

There are cases where web applications do not necessarily comply with all possible web browsers. For example, certain aspects of CSS and HTML might work for one browser but break in another. This application shall ensure that the CSS and HTML model will be compliant with the browser’s boundaries. Ensuring browser compliance, the application shall be able to run on virtually any operating system that has any of the following browsers: Safari, Opera, Internet Explorer, Firefox and Chrome.

### **Ease of Use**

The application shall be straightforward and lightweight in order to ensure that students can start working on their schedule planner immediately. The use of Ajax technology will also minimize content and page refreshes thereby allowing users to navigate without losing the scope of the data they were working on.

# Resource Evaluation

## Human Resources

Present here all your team members in terms of their capacities (strengths, knowledge, experience), as well as an evaluation of the time they can consecrate to the project over the semester, and possible lack of availability during certain periods, which is important information to consider for concrete planning.

## Technical Resources

Each developer’s computer must have the same XAMPP version installed in order to align the Apache and PHP version as found in the production environment. This is to avoid any conflicts and bugs which can be caused due to incompatible versions of PHP or Apache’s server core files.

XAMPP (64-bit environment) includes the following pre-installed software packages:

* Database: MySQL 5
* Web Server: Apache HTTP Server 2.0
* Primary language: PHP 6.0 and Python 2.7
* Database User Interface: phpMyAdmin

The production environment in question has the following infrastructure:

* CPU: Intel i7
* OS: Windows Server 2012
* RAM: 16GB
* DISK DRIVE: 500 GB (Solid State)
* Network Speed: 10 Mbps

The production environment is hosted on a team member’s own computer. This decision was made in order to have a better overlook of the server and have complete control of the environment.

Main software packages used in the project include:

* Languages: HTML, CSS, PHP, JQuery, JavaScript, SQL
* IDE: PhpStorm 10
* Version Control Systems: GitHub, Gitbash, GitHub Desktop
* Team Collaboration Tools: GitHub, Facebook Chat, Trello
* OS: Windows, Linux
* UML and Diagrams: StarUML
* Documentation tools: Google Drive, Microsoft Word
* Graphical Design Tool: Photoshop

# Scoping

With regard to the cost/time necessary to meet all the goal presented in section 3, and given the resources available presented in section 4, give here the scope of the solution that you will implement in the project.

List all features/goals/qualities that will be left out because of lack of time/resources.

# Solution Sketch

## Architecture

Explain the high-level system architecture that you plan for the project, giving explanations as to why you chose such an approach, e.g. a design rationale. Include a high-level architecture diagram (e,g, a high-level UML class diagram). For each module presented in the architecture, also give its rationale, as well as list their individual responsibilities.

## Technologies in Use

### **Programming Languages**

Server-side

6.2.1.1.1 PHP

PHP is the main server-side language that will be handling business logic and SQL database queries. PHP was chosen due to the syntax similarities of other major Object-Oriented programming languages, allowing for a quicker learning experience.

#### ***Client Side***

6.2.1.2.1 JavaScript

JavaScript a multi-paradigm scripting language that is supported and natively embedded by all major browsers including Internet Explorer, Firefox, Safari and Chrome. It is mainly used for dynamic client side manipulation of the DOM and allows for a dynamic browsing experience.

6.2.1.2.2 jQuery

jQuery is essentially a JavaScript script library that allows programmers to write less lines of code in order to perform the same functionality in the native JavaScript environment. It allows for quick DOM manipulation with very few lines of code, animations, client side validation, form input handling and quick Ajax calls.

6.2.1.2.3 HTML (Hyper Text Markup Language)

HTML is the primary markup language used to create web layout and provide structure to web content. Of course, it is also natively supported in every major web browser.

6.2.1.2.4 CSS (Cascading Style Sheet)

CSS is used primarily for styling purposes that is supported by all major browsers with some detailed exceptions in portability. The essence of CSS is to separate styling from the markup language, such as HTML.

### **IDEs**

#### ***PhpStorm 10***

PhpStorm is a powerful IDE provided by JetBrains that is integrated with an extensive Git support system and provides direct database management without the use of any other third party application.

### **Database Management System**

#### ***MySQL***

MySQL is an open-source database management system that provides more than enough features required to for the scope of the web application. It also provides InnoDB storage engine access which supports transactions, enabling to undo database query commits.

### **Web Server**

#### ***XAMPP (v1.8.1)***

XAMPP 1.8.1 is a free open-source, cross-platform web server solution package which includes Apache HTTP Server, MySQL and PHP functionalities.

### **Source Code Revision Management**

#### ***GitHub***

GitHub is a popular source code revision management system that allows for developers to clone source code repositories and provide a non-destructive means of working in isolation from the original repository. In turn, this provides a better overview of which developer is working on which portion of the code, thereby providing a flexible version control workflow.

### **Deployment Software**

#### ***DeployHQ***

DeployHQ is an online service that allows automatic and manual deployment of sources code from any type of repository service such as GitHub. It is used to automatically deploy source code directly to the main hosting server from a selected GitHub branch of your choice in a seamless manner. The service is installed via a webhook service, integrated within GitHub and source code deployment occurs upon a commit done in the master branch. It also provides rollback options.

### **Team Collaboration**

#### ***Trello***

Trello is a free team collaboration tool that organizes task into board spaces. It provides an overview of what tasks need to be done, tracks due dates and assigns tasks to collaborators. It also provides a mobile app version which allows to push notifications to users’ mobile when the project board was updated, allowing collaborators to respond to tasks accordingly.

# Plan

## Activities

|  |  |  |
| --- | --- | --- |
| Activity | Purpose/Description | Artifact(s) produced |
| 7.1.1  Form and divide team | Forming a team allows the project to start. Dividing the team into a coding and documentation sub-teams allows members to focus on particular tasks. Such division requires the evaluation of the strengths and weaknesses of all members to determine their role. | 1.4.1 Human Resources |
| 7.1.2  Complete the system overview | The system overview consists of finding a name for the system, determining the main purposes and elements involved in it, and drawing a domain model of the system along with the description of each entity found in the domain model. | Deliverable 0  1.3.2 Domain Model |
| 7.1.3  Write a project description | The project description introduces the entire document. It informs the reader about its goals, the information to be found in it, as well as the purposes and objectives of the project. | 1.2 Project Description |
| 7.1.4  Determine the functional requirements and draw use case diagrams accordingly | Determine the requirements that the system should have, including those that may later be scoped out, their relative importance and relative difficulty. Define the actions that must take place in the software and illustrate each requirement by the mean of a use case diagram. | 1.3.1 Functional Requirements |
| 7.1.5  Review and correct domain model | Make any necessary changes according to the feedbacks of Deliverable 0. | 1.3.2 Domain Model |
| 7.1.6  Find and describe constraints and qualities | Describe any design constraints, qualities and non-functional requirements that the system should meet. | 1.3.3 Constraints and Qualities |
| 7.1.7  Review and format human resources document | After the profile of each team member has been determined, the document should be revised and formatted correctly. | 1.4.1 Human Resources |
| 7.1.8  Determine technical resources | List the computer resources and tools available to complete the project. | 1.4.2 Technical Resources |
| 7.1.9  Reduce the amount of features to be realistic according to time and resources | Outline the scope of the software. List all features, goals and qualities that are scoped out, and provide an explanation for each element that is scoped out. | 1.5 Scoping |
| 7.1.10  Choose the architecture of the system | Explain the high-level architecture of the system, and give a design rationale. A diagram should accompany the explanation. | 1.6.1 Architecture |
| 7.1.11  Decide which technologies are relevant to the project development | List technologies that will be used in the project and give a rationale for each of them. | 1.6.2 Technologies in Use |
| 7.1.12  List and describe the main activities involved in the project | List all activities that produce at least one artifact and include a description for each activity. | 1.7.1 Activities |
| 7.1.13  List all artifacts | List and describe all artifacts that need to be produced for the project. | 1.7.2 Artifacts |
| 7.1.14  Estimate the cost and time of production | Estimate the cost and time of production of each artifact and sum them up. | 1.7.3 Project Estimates |
| 7.1.15  Delegate activities to different team members | Assign each activities of the project to a team member or to a group of team members. | 1.7.4 Activities Assignments |
| 7.1.16  Produce a schedule for the project | Create a schedule/timeline of all activities for the project using a Gantt chart. Start dates, due dates, names of activities, and participants are all information to be included in the schedule. | 1.7.5 Schedule |
| 7.1.17  Determine the risks of the project | Outline and explain the risks associated and presented by this project. | 1.7.6 Risk |
| 7.1.18  Produce a prototype report | Describe the work done in the development of the prototype. Explain how the chosen technologies are appropriate, and how the team members are comfortable using such technologies. | 1.8 Prototyping |
| 7.1.19  Ensure professionalism of the document | Assemble all parts of deliverable 1. Ensure the document is professional looking and well-organized. | Deliverable 1 |
| 7.1.20  Update the introduction for Deliverable 2 | Update the introduction to give the reader an overview of the content found in Deliverable 2: Architecture and Design | 2.2 Introduction to Part 2 |
| 7.1.21  Produce an architecture diagram | Produce a 4+1 Architectural View. Include a rationale for the design. Explain any differences between the old design and this one. | 2.3.1 Architecture Diagram |
| 7.1.22  Specify the interactions between the components of the software | Describe the interactions between the components of the software through their interfaces. Include the function calls, the description of the parameters and the range of accepted values of those parameters. | 2.3.2 Subsystem Interfaces Specifications |
| 7.1.23  Illustrate the internal structure of the system | Provide a graphical representation of the structure of each subsystem by means of a UML class diagram. A description of each class should be included. | 2.4.1 Detailed Design Diagram |
| 7.1.24  Describe each class in the subsystem | Provide the programmers with the descriptions of each class in the UML diagram along with any necessary detailed relevant to the development. | 2.4.2 Unit Description |
| 7.1.25  Produce dynamic design scenarios | Draw the dynamic design of two use cases. | 2.5 Dynamic Design Scenario |
| 7.1.26  Update project estimates | Update the cost of the project for each module and include the cost of integration, testing and documentation. | 2.6 Estimation |
| 7.1.27  Produce a report about the prototype and risk | List and describe all elements that have been implemented and describe the effect that those implementations had on the design decisions, risks, estimate and scope. | 2.7 Rapid Prototyping and Risk |
| 7.1.28  Ensure professionalism of the document | Assemble all parts of deliverable 2. Ensure the document is professional looking and well-organized. | Deliverable 2 |
| 7.1.29  Update introduction | Update the introduction so that it includes an overview of the content presented in deliverable 3. | 3.2 Introduction |
| 7.1.30  List all tested items | Create a list of all items that have been tested, the test cases that were used, and a rationale for the test. | 3.3.1.1 Tested Items |
| 7.1.31  List all untested items | Create a list of items that are to be tested, an explanation of why those items should be tested, and how they could be tested. | 3.3.1.2 Untested Items |
| 7.1.32  Unit testing report | Describe the test cases, the technique used, the code used and the result of the testing for two units. | 3.3.2.1 Unit Testing |
| 7.1.33  Requirements testing report | Provide a list of test cases for all tested requirements by means of scenario of system usage and system reaction. | 3.3.2.2 Requirements Testing |
| 7.1.34  Stress testing report | Describe situations of extreme system usage, the tests designed to evaluate the performance of the system under such condition, and the test results. | 3.3.2.3 Stress Testing |
| 7.1.35  Security testing report | Describe tests performed to protect the system against security threats such as SQL injections and automated tools. | 3.3.2.4 Security Testing |
| 7.1.36  Installation manual redaction | Guide any administrators to install and execute the software. | 3.4.1 Installation Manual |
| 7.1.37  User’s manual redaction | Guide any users to use the system with all of its features. | 3.4.2 User’s Manual |
| 7.1.38  Update cost estimate | Update the table of costs with all components of all phases. | 3.5 Final Cost Estimate |
| 7.1.39  Ensure professionalism of the document | Assemble all parts of deliverable 2. Ensure the document is professional looking and well-organized. | Deliverable 3 |
| 7.1.40  Correction of various part of the deliverables | Make all necessary changes in the deliverables for the final report. | Corrected deliverables |
| 7.1.41  Assemble all deliverables | Put all corrected/updated deliverables together for the final report. Ensure professionalism of the document. | Final report |

## Artifacts

Describe what artifacts you are going to produce in this project. Do not limit yourself to “deliverable 1, deliverable 2 and deliverable 3”, but rather split them into lower level units, e.g. “class diagram”, “list of requirements”, “test cases”, etc. Explanations describing the role of each artifact in the production process.

## Project Estimates

Provide a realistic estimated cost and schedule for the project, as well as the basis for those estimates, and the points and circumstances in the project when re-estimation might occur.

Evaluate the cost of production of each artifact, as described in the previous section, and then adding up the numbers.

## Activities Assignments

|  |  |
| --- | --- |
| Activity | Assigned member(s) |
| 7.1.1  Form and divide team | Claudia Della Serra  Dimitri Topaloglou  Philip Lim  Ryan Lee  Daniel Di Corpo  Erin Benderoff  Aline Koftikian  Ideawin-Bunthy Koun  Kevin Yasmine  Marc-Andre Leclair  Lori Dalkin  Bryce Drewery-Schoeler |
| 7.1.2  Complete the system overview | Claudia Della Serra  Dimitri Topaloglou  Philip Lim  Ryan Lee  Daniel Di Corpo  Erin Benderoff  Aline Koftikian  Ideawin-Bunthy Koun  Kevin Yasmine  Marc-Andre Leclair  Lori Dalkin  Bryce Drewery-Schoeler |
| 7.1.3  Write a project description | Ryan Lee |
| 7.1.4  Determine the functional requirements and draw use case diagrams accordingly | Dimitri Topaloglou  Bryce Drewery-Schoeler  Ideawin-Bunthy Koun  Erin Benderoff  Lori Dalkin  Marc Leclair  Claudia Della Serra  Daniel Di Corpo |
| 7.1.5  Review and correct domain model | Aline Koftikian |
| 7.1.6  Find and describe constraints and qualities | Dimitri Topaloglou |
| 7.1.7  Review and format human resources document | Lori Dalkin |
| 7.1.8  Determine technical resources | Dimitri Topaloglou |
| 7.1.9  Reduce the amount of features to be realistic according to time and resources | Kevin Yasmine |
| 7.1.10  Choose the architecture of the system | Daniel Di Corpo |
| 7.1.11  Decide which technologies are relevant to the project development | Dimitri Topaloglou |
| 7.1.12  List and describe the main activities involved in the project | Philip Lim |
| 7.1.13  List all artifacts | Aline Koftikian |
| 7.1.14  Estimate the cost and time of production | Daniel Di Corpo |
| 7.1.15  Delegate activities to different team members | Philip Lim |
| 7.1.16  Produce a schedule for the project | Ryan Lee |
| 7.1.17  Determine the risks of the project | Dimitri Topaloglou  Kevin Yasmine |
| 7.1.18  Produce a prototype report | Lori Dalkin |
| 7.1.19  Ensure professionalism of the document | Claudia Della Serra |
| 7.1.20  Update the introduction for Deliverable 2 | Ryan Lee |
| 7.1.21  Produce an architecture diagram | Aline Koftikian  Dimitri Topaloglou  Daniel Di Corpo |
| 7.1.22  Specify the interactions between the components of the software | Ideawin-Bunthy Koun  Marc-Andre Leclair  Philip Lim |
| 7.1.23  Illustrate the internal structure of the system | Erin Benderoff  Aline Koftikian |
| 7.1.24  Describe each class in the subsystem | Kevin Yasmine  Lori Dalkin  Bryce Drewery-Schoeler |
| 7.1.25  Produce dynamic design scenarios | Ryan Lee  Claudia Della Serra |
| 7.1.26  Update project estimates | Daniel Di Corpo |
| 7.1.27  Produce a report about the prototype and risk | Dimitri Topaloglou |
| 7.1.28  Ensure professionalism of the document | Claudia Della Serra |
| 7.1.29  Update introduction | Ryan Lee |
| 7.1.30  List all tested items | Claudia Della Serra |
| 7.1.31  List all untested items | Ryan Lee |
| 7.1.32  Unit testing report | Lori Dalkin  Aline Koftikian |
| 7.1.33  Requirements testing report | Erin Benderoff  Philip Lim |
| 7.1.34  Stress testing report | Ideawin-Bunthy Koun  Kevin Yasmine |
| 7.1.35  Security testing report | Bryce Drewery-Schoeler  Marc-Andre Leclair |
| 7.1.36  Installation manual redaction | Claudia Della Serra  Dimitri Topaloglou  Philip Lim  Ryan Lee  Aline Koftikian  Kevin Yasmine |
| 7.1.37  User’s manual redaction | Daniel Di Corpo  Claudia Della Serra  Philip Lim  Ryan Lee  Aline Koftikian  Kevin Yasmine |
| 7.1.38  Update cost estimate | Daniel Di Corpo |
| 7.1.39  Ensure professionalism of the document | Claudia Della Serra |
| 7.1.40  Correction of various part of the deliverables | Claudia Della Serra  Dimitri Topaloglou  Philip Lim  Ryan Lee  Daniel Di Corpo  Erin Benderoff  Aline Koftikian  Ideawin-Bunthy Koun  Kevin Yasmine  Marc-Andre Leclair  Lori Dalkin  Bryce Drewery-Schoeler |
| 7.1.41  Assemble all deliverables | Claudia Della Serra |

## Schedule

Diagrams or tables showing target dates for completion of iterations and phases, release points, demos, and other milestones, e.g. a Gantt chart.

## Risk

List and comment all the elements of the project that present a risk. Explain why they represent a risk.

# Prototyping

*Describe any work undertaken in development of a prototype. A prototype should be developed during this phase both to prove the technologies used are appropriate to the task and to assess the competence of the team to work with the selected technologies.*