Module Guide for Bomber Clone Application

Software Engineering 3XA3: Project

Group 12 - The A Team
Gabriel Lopez De Leon, 1310514
Ren-David Dimen, 1222679
Jay Nguyen, 1327828

06 November 2015

Table of Contents

1 Revision History

Revision #	Revision Date	Description of Change	Author
6	Nov 06 2015	Added Section 8	Ren-David Dimen
5	Nov 06 2015	Added Section 9	Gabriel Lopez de Leon
4	Nov 05 2015	Added Section 4, 6, 7	Gabriel Lopez de Leon
3	Nov 02 2015	Added Section 5	Ren-David Dimen
2	Nov 02 2015	Added Section 1 and 2	Gabriel Lopez de Leon
1	Nov 02 2015	Added Section 3	Ren-David Dimen

2 Introduction

Our project is the recreation of a game called Bomberman, we are taking legacy code for a game very similar to the original and reimplementing it with proper code structure and documentation. Our Software Requirements Specification (SRS) document and Test Plan describe how we are planning on implementing the game and how it will be tested through the use of manual and automated tests. The game overall will have the same objectives as the original, to eliminate other players by placing bombs in the arena. We will be using threads in the implementation of the game and instead of using 2D arrays or an arena divided into tiles, we plan to use pixels for hit detection to make the game look and feel more smooth as it is played. To test the various functional and non-functional requirements we will be using black box, white box and unit testing as aforementioned in our Test Plan.

While the SRS focuses on the ”what”, the Design document focuses on the ”how.” The design documentation for our project is divided into the Module Guide (MG) and the Module Interface Specification (MIS). This document will focus on the MG specifically and will show the decomposition of the system into smaller subsections which can be discussed in further detail while also showing how these different modules all relate to one another. This decomposition of a system is a commonly accepted approach in software development. One of the design principles we want to use is Information Hiding. This supports design based on change as the various ”secrets” each module hides is equivalent to anticipated changes. Having a design follow this principle allows for quick and easy modification of code which we had already anticipated would change. Furthermore, there is a set structure to the code, parts that are unlikely to change a group together while as the once subject to likely change in the future are also grouped together.

The rest of the document will be organized as follows:
- Section 2 will list the anticipated and unlikely changes in the system.
- Section 3 will summarize the module decomposition based on likely future changes.
- Section 4 will specify the relation between requirements and the modules.
- Section 5 will be describing each module in further detail.
- Section 6 will show traceability matrices for relations between modules and requirements, and between modules and anticipated changes.

- Section 7 will describe the relationships between the modules.

3 Anticipated and Unlikely Changes

This section lists possible changes within the program according to the likeliness of the change. Possible changes are classified into two different categories.

3.1 Anticipated Changes

Anticipated changes are the source of the information that is to be hidden inside the modules. Ideally, changing one of the anticipated changes will only require changing the one module that hides the associated decision. The approach adapted here is called design for change.

AC1: The data structure in which the state of the board is saved.

AC2: The initial state of the board.

AC3: Character states.

AC4: How the program's state is displayed.

AC5: Character hit-box size.

AC6: Window output size.

3.2 Unlikely Changes

Modules with unlikely changes may not inherently follow the norms of information hiding. Interacting modules may need to be updated if one or more module is modified, therefore these changes must be dealt with care so as to not negatively affect the outcome of the program.

UC1: Input hardware devices. (ie keyboard and mouse)

UC2: Output hardware devices. (ie monitor)

UC3: User control inputs. (up, down, left, right arrow keys, etc.)

UC4: Data type of the character sprites.

4 Module Hierarchy

This section provides an overview of the module design. Modules are summarized in a hierarchy decomposed by secrets in Table 1. The modules listed below are the modules which will be implemented.

M1: Hardware-Hiding Module

M2: Screen Module

M3: Sprite Module

M4: SpriteSheet Module

M5: Keyboard Module

M6: Player Module

M7: CollisionTest Module

Level 1	**Level 2**
Hardware-Hiding Module	
Behaviour-Hiding Module	Screen Module Sprite Module SpriteSheet Module Keyboard Module Player Module
Software Decision Module	CollisionTest Module

Table 1: Module Hierarchy

5 Connection Between Requirements and Design

The design of the system is intended to satisfy the requirements developed in the SRS. In this stage, the system is decomposed into modules. The connection between requirements and modules are listed in section 7, Traceability Matrix.

6 Module Decomposition

Modules are decomposed following the principle of "information hiding" which was proposed by David Parnas. The secrets listed within the module decomposition will provide information on the design decision hidden by its respective module. The "service" section will be describing what each module does without specifying how it will do it. For each of the following modules, a suggestion for the implementation software used is given under the section "Implemented By." An entry of OS refers to modules provided by the operating system/standard programming language libraries and a dash (-) means the module will not have to be implemented.

6.1 Hardware Hiding Modules (M1)

Secrets: The data structure and algorithm used to implement the virtual hardware.

Services: Serves as virtual hardware used by the rest of the system. This module provides the interface between the hardware and the software. So, the system can use it to display outputs or to accept inputs.

Implemented By: OS

6.2 Behaviour-Hiding Module

Secrets: The contents of the required behaviours.

Services: Includes programs that provide externally visible behaviour of the system as specified in the software requirements specification (SRS) document. This module serves as a communication layer between the hardware-hiding module and the software decision module. Programs in this module will need to change if there are changes in the SRS.

Implemented By: -

6.2.1 Screen Module (M2)

Secrets: The format and structure of the display (output) data.

Services: This module is used to render images and output them into a display.

Implemented By: Eclipse

6.2.2 Sprite Module (M3)

Secrets: The format and structure of the sprites (input) used.

Services: Obtains the sprites from a set path and uses them for different character actions/movement.

Implemented By: Eclipse

6.2.3 SpriteSheet Module (M4)

Secrets: The format and structure used to obtain the sprite sheet.

Services: Sets the main path to the sprite sheet which will be loaded and used in other modules.

Implemented By: Eclipse

6.2.4 Keyboard Module (M5)

Secrets: The format and structure of user input.

Services: Takes input date from keyboard keys and converts it to data to be used by other modules.

Implemented By: Eclipse

6.2.5 Player Module (M6)

Secrets: The format and structure of the player entity.

Services: Links input data to the player sprite.

Implemented By: Eclipse

6.3 Software Decision Module

Secrets: Design decisions based on mathematical theorems, physical facts, or programming considerations are found here. The various secrets of this module are not described in the SRS.

Services: Includes data structure and algorithms used in the system that do not provide direct interaction with the user.

Implemented By: -

6.3.1 CollisionTest Module (M7)

Secrets: The format and structure of the object states in the game.

Services: Handles the collision detection (hit tests) between objects.

Implemented By: Eclipse

7 Traceability Matrix

This Section shows traceability matrices between modules and requirements, and between modules and anticipated changes. Note that requirements R1 to R5 are listed on the SRS.

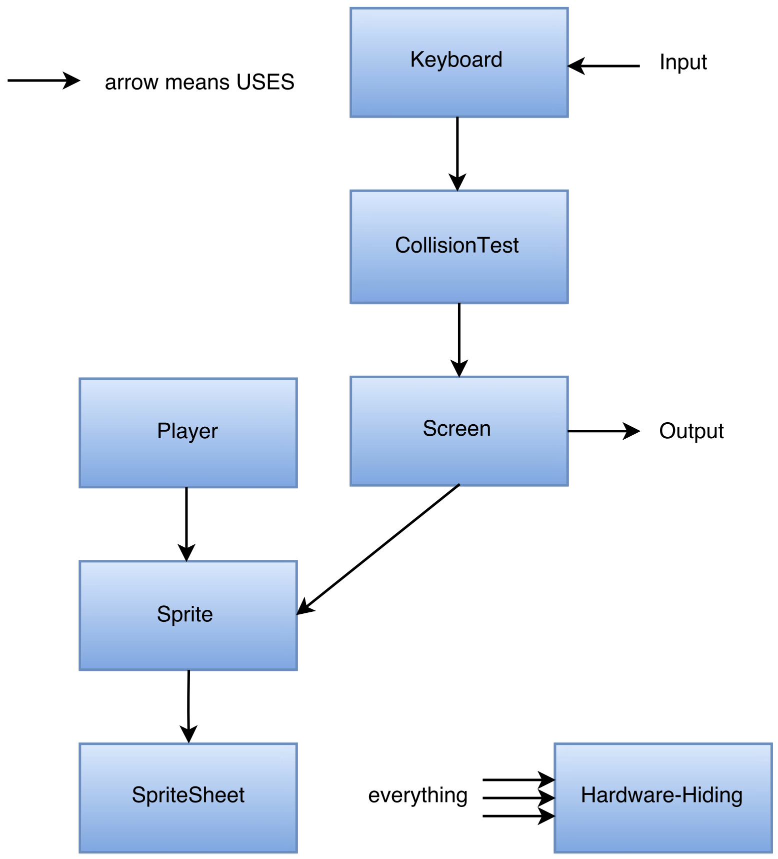
Req.	Modules
R1	M2, M7
R2	M2
R3	M2, M3, M4
R4	M5
R5	M6

Table 2: Trace Between Requirements and Modules

AC.	Modules
AC1	M7
AC2	M2
AC3	M6
AC4	M4
AC5	M3
AC6	M1

Table 3: Trace Between Anticipated Changes and Modules

8 Use Hierarchy Between Modules


arrow means USES
Keyboard
Input
CollisionTest
Player
Screen
Output
Sprite
SpriteSheet
everything
Hardware-Hiding

9 Schedule

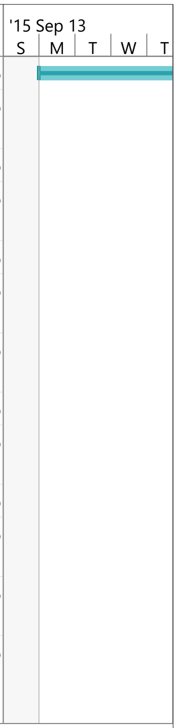
9.1 Milestones

Note that these milestones also appear on the Gantt and Pert charts found in the design document folder in the repository.

A. testing - to be completed by November 25, 2015
B. Test Report Revision 0 - November 27, 2015
C. Complete implementation of code - complete by November 27, 2015
D. Final Demonstration Revision 1 - November 30 - December 4, 2015
E. Final Documentation Revision 1 - December 8, 2015

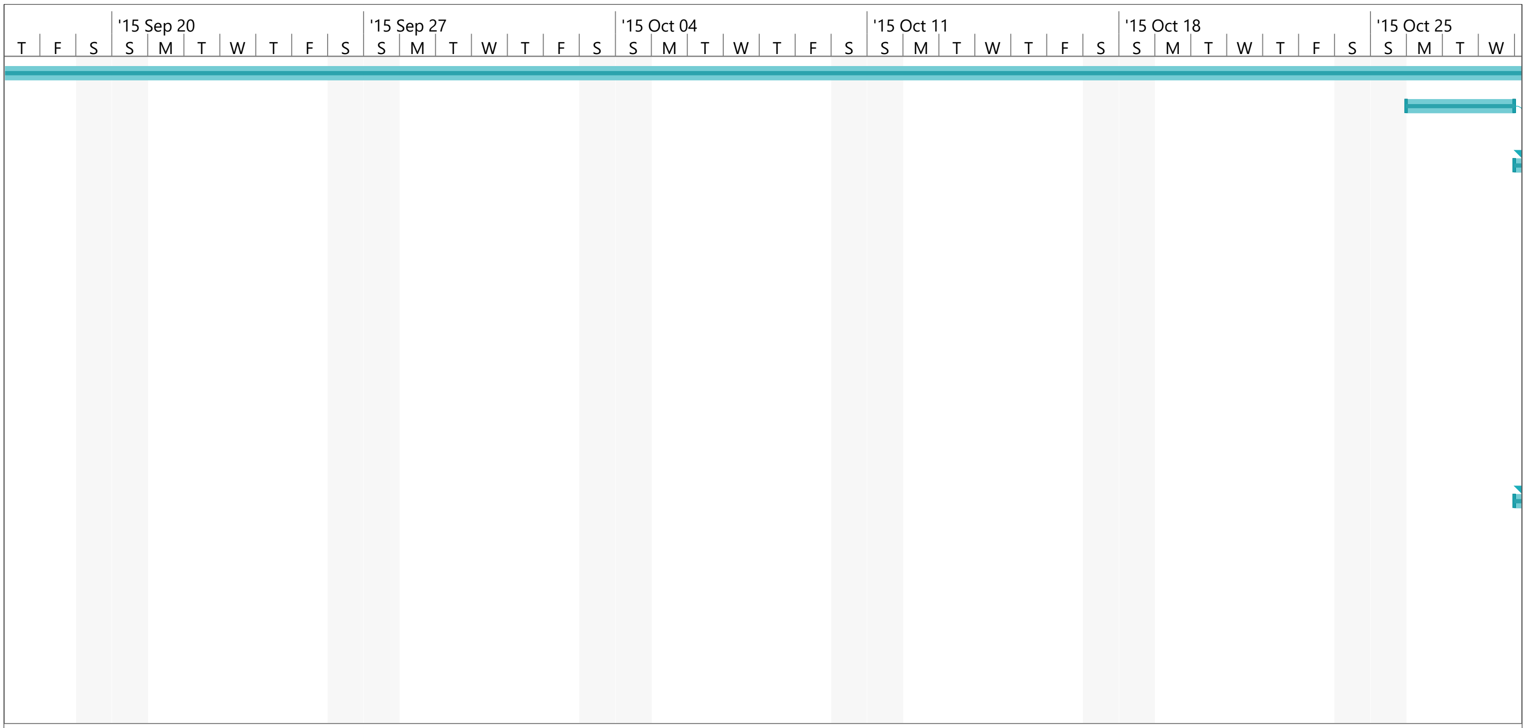
9.2 Gantt Chart

ID	ℹ	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names	% Complete
1	✓	📌	Lab Exercises	40 days	Mon 15-09-1	Fri 15-11-06		All	100%
2	✓	📌	Proof of Concept Demonstration	3 days	Mon 15-10-26	Wed 15-10-28		All	100%
3	✓	📌	Test Plan Revision 0	3 days	Thu 15-10-29	Mon 15-11-0	2	Gabriel,Ren-Davic	100%
4	●	📌	Complete Module Guide	5 days	Sun 15-11-01	Thu 15-11-05		Gabriel,Ren-Davic	48%
5	●	📌	Complete MIS	5 days	Sun 15-11-01	Thu 15-11-05		Gabriel	0%
6	●	📌	Design Document Revision 0	1 day	Fri 15-11-06	Fri 15-11-06	4,5	All	40%
7		📌	Revision 0 Demonstration	3 days	Mon 15-11-16	Wed 15-11-18		All	0%
8	●	📌	Testing	12 days	Mon 15-11-0	Tue 15-11-24		Jay,Ren-David	0%
9		📌	Test Report Revision 0	1 day	Fri 15-11-27	Fri 15-11-27	8	All	0%
10	●	📌	Coding	22 days	Thu 15-10-29	Fri 15-11-27	2	Jay	33%
11	●	📌	Peer Evaluation of Other Teams Rev 0	5 days	Mon 15-11-30	Fri 15-12-04		All	0%
12	●	📌	Final Demonstration (Revision 1)	5 days	Mon 15-11-30	Fri 15-12-04	10	All	0%
13		📌	Final Documentation (Revision 1)	1 day	Tue 15-12-08	Tue 15-12-08		All	0%



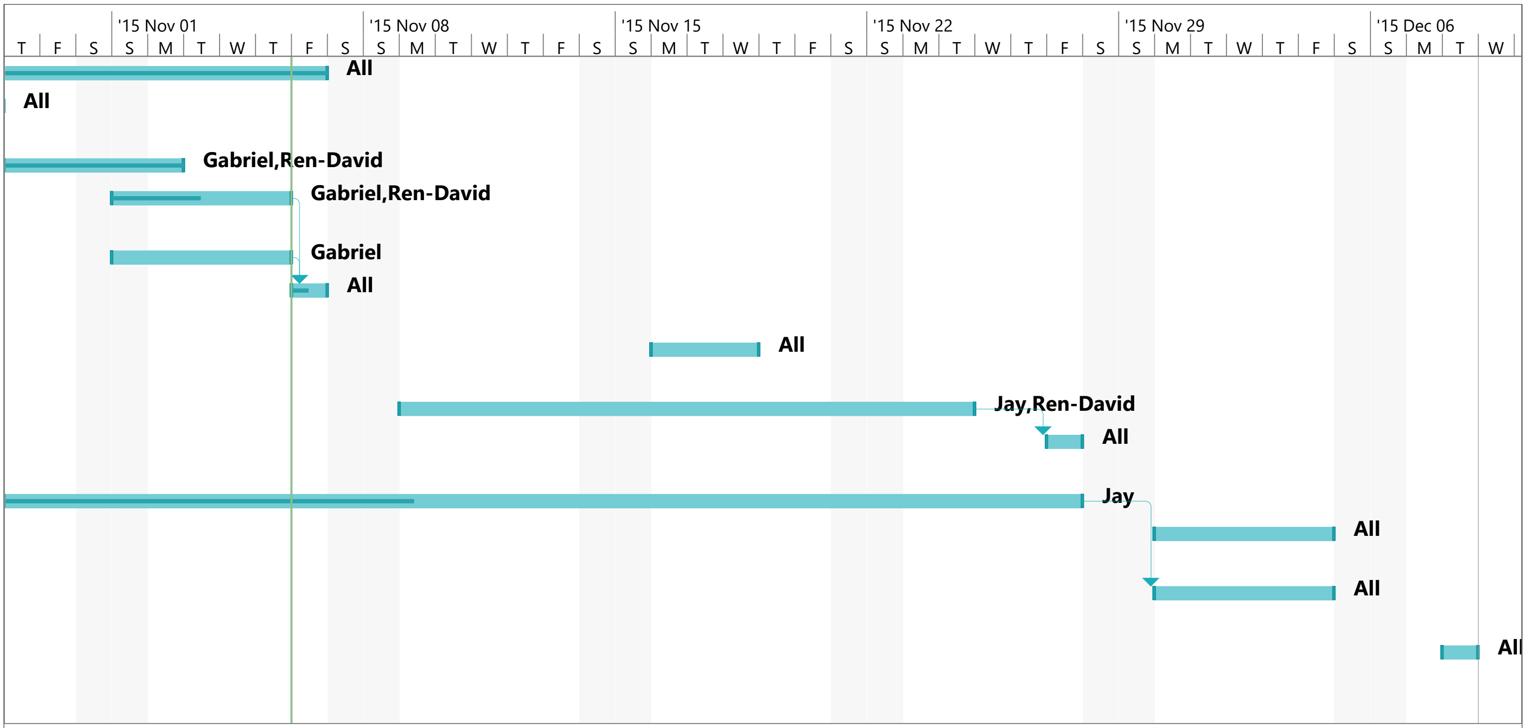
Project: projectCharts
Date: Fri 15-11-06

Task	▬	Inactive Summary	┌─┐	External Tasks	▬
Split	┈┈┈	Manual Task	▬	External Milestone	◇
Milestone	◆	Duration-only	▬	Deadline	⬇
Summary	┌─┐	Manual Summary Rollup	▬	Progress	─
Project Summary	┌─┐	Manual Summary	┌─┐	Manual Progress	─
Inactive Task	▭	Start-only	[		
Inactive Milestone	◇	Finish-only	]		



Project: projectCharts
Date: Fri 15-11-06

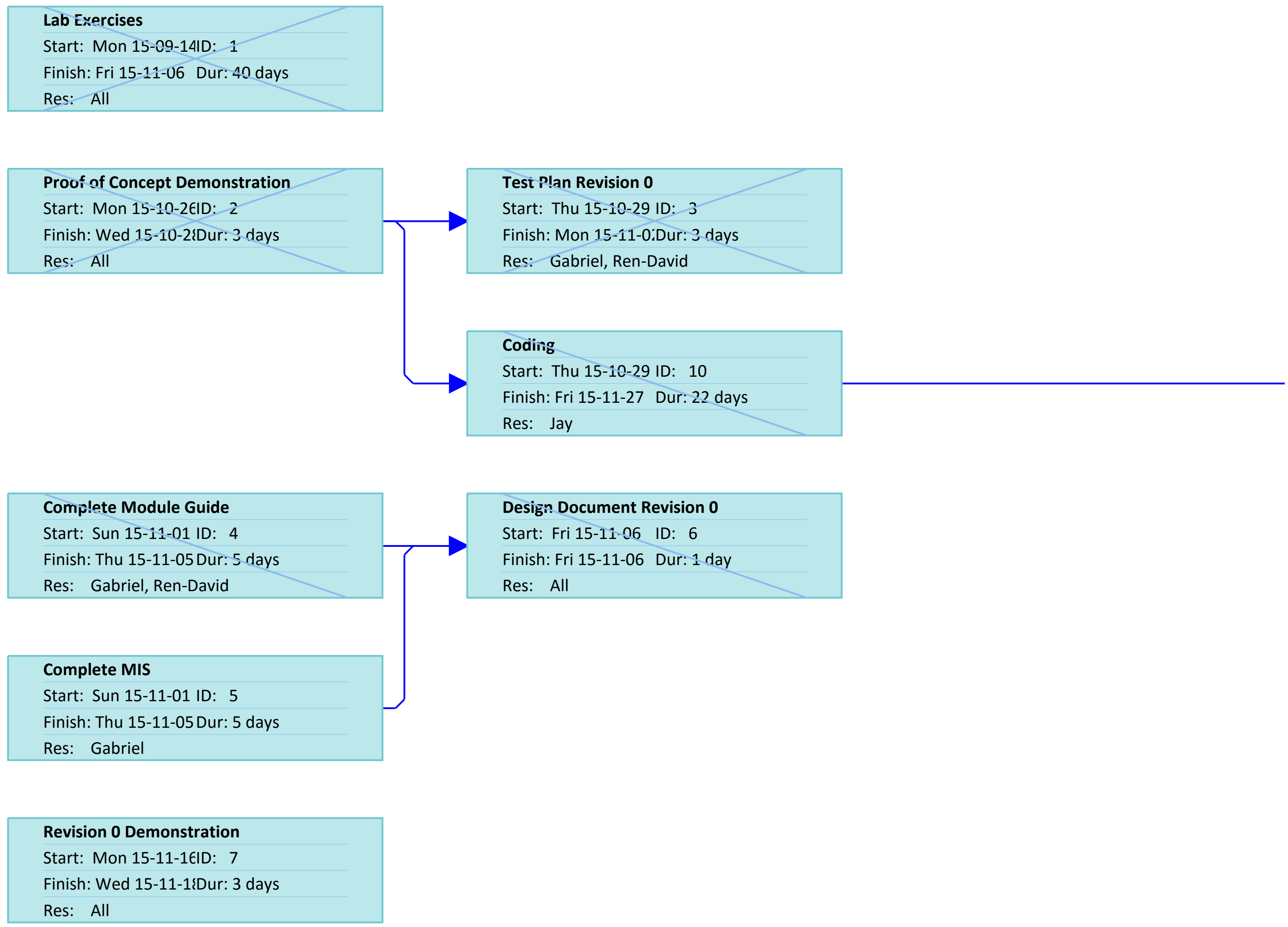
Task	▬	Inactive Summary	▭	External Tasks	▬
Split	┅	Manual Task	▬	External Milestone	◇
Milestone	◆	Duration-only	▬	Deadline	⬇
Summary	┏━┓	Manual Summary Rollup	━	Progress	━
Project Summary	┏━┓	Manual Summary	┏━┓	Manual Progress	━
Inactive Task	▭	Start-only	[		
Inactive Milestone	◇	Finish-only	]		



Project: projectCharts
Date: Fri 15-11-06

Task	▬	Inactive Summary	⊓	External Tasks	▬
Split	┈	Manual Task	▬	External Milestone	◇
Milestone	◆	Duration-only	▬	Deadline	⬇
Summary	⊓	Manual Summary Rollup	▬	Progress	━
Project Summary	⊓	Manual Summary	⊓	Manual Progress	━
Inactive Task	▭	Start-only	[		
Inactive Milestone	◇	Finish-only	]		

9.3 Pert Chart



Lab Exercises
Start: Mon 15-09-14ID: 1
Finish: Fri 15-11-06 Dur: 40 days
Res: All

Proof of Concept Demonstration
Start: Mon 15-10-26ID: 2
Finish: Wed 15-10-28Dur: 3 days
Res: All

Test Plan Revision 0
Start: Thu 15-10-29 ID: 3
Finish: Mon 15-11-02Dur: 3 days
Res: Gabriel, Ren-David

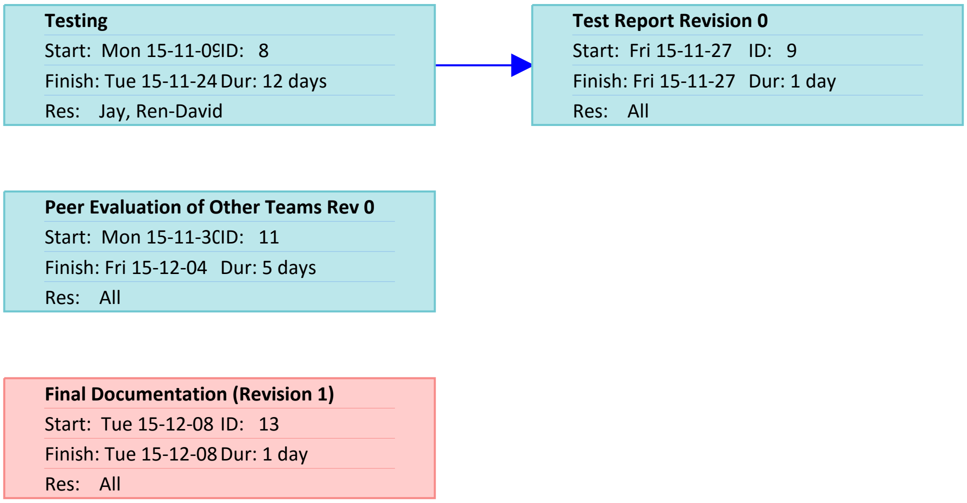
Coding
Start: Thu 15-10-29 ID: 10
Finish: Fri 15-11-27 Dur: 22 days
Res: Jay

Complete Module Guide
Start: Sun 15-11-01 ID: 4
Finish: Thu 15-11-05 Dur: 5 days
Res: Gabriel, Ren-David

Design Document Revision 0
Start: Fri 15-11-06 ID: 6
Finish: Fri 15-11-06 Dur: 1 day
Res: All

Complete MIS
Start: Sun 15-11-01 ID: 5
Finish: Thu 15-11-05 Dur: 5 days
Res: Gabriel

Revision 0 Demonstration
Start: Mon 15-11-16ID: 7
Finish: Wed 15-11-18Dur: 3 days
Res: All

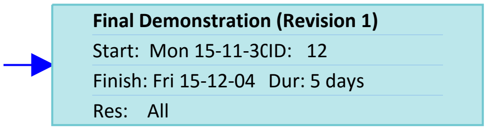


Testing
Start: Mon 15-11-09ID: 8
Finish: Tue 15-11-24 Dur: 12 days
Res: Jay, Ren-David

Test Report Revision 0
Start: Fri 15-11-27 ID: 9
Finish: Fri 15-11-27 Dur: 1 day
Res: All

Peer Evaluation of Other Teams Rev 0
Start: Mon 15-11-30ID: 11
Finish: Fri 15-12-04 Dur: 5 days
Res: All

Final Documentation (Revision 1)
Start: Tue 15-12-08 ID: 13
Finish: Tue 15-12-08 Dur: 1 day
Res: All



Final Demonstration (Revision 1)

Start: Mon 15-11-3CID: 12

Finish: Fri 15-12-04 Dur: 5 days

Res: All

Project: projectCharts
Date: Fri 15-11-06

Critical	🟥	Summary	▰	Critical External	▢
Noncritical	□	Critical Inserted	▱	External	▢
Critical Milestone	⬢	Inserted	▱	Project Summary	▱
Milestone	⬡	Critical Marked	⬢	Highlighted Critical	▨
Critical Summary	▰	Marked	⬡	Highlighted Noncritical	▨