Drawdiculous

[Project Plan]
Version 1.0

By: Bian Hengwei, He Yinan, Jin Han, Melvin, Rama, Shipra

Revision History

Revision Number	Date	Primary Author(s)	Comments
1.0	Oct 13, 2021	Bian Hengwei, He Yinan, Jinhan, Melvin, Rama, Shipra	First version

Table of Contents

1 Introduction

1.1 Project Overview

This project is a realtime, multiplayer mobile gaming application that allows players to draw and guess certain words. In each round of the game, one of the players draws a certain word while others try to guess what that word is. This game aims to cater to the social and cognitive needs of the elderly.

1.2 Project Description and Scope

This project proposes the creation of an easy, fun and sociable drawing game that can train the capabilities of the brains of the elderly, especially their retrospective/autobiographical memory. At the same time, this game keeps them mentally and socially active. Our proposed solution leverages the widespread availability of smartphones in Singapore, even among the edlerly, to implement a drawing game that is online and can be connected to other players. For our project, the main target will thus be the elderly in Singapore.

The design of our mobile game has the following main objectives:

1. To enable the game to be easy to use for the elderly.
2. To allow the elderly to train their memory.
3. To allow the elderly to interact with fellow players.

However, in order to achieve the above objectives, there are some limitations:

1. The words must be familiar and relatable to a typical Singaporean elderly.
2. The words implemented must be available in several languages.

The words chosen for the drawing game must be familiar to the players, even if they are from different socioeconomic backgrounds. As such, the range of words that our team can choose to implement is limited to those that are short and common. Furthermore, as many elderly do not have English as their first language, our game will be available in several other languages including Mandarin and Malay. Hence, the word prompts must be available in those languages as well, further limiting the range of words due to system, budget and time constraints.

2 Project Organization

2.1 Team Structure

The following is the list of member roles.

- Project Manager: Bian Hengwei
- Lead Developer: Renganathan Ramasamy
- Front-end Developer: Jin Han
- Back-end Developer: He Yinan
- Quality Assurance Engineer & Manager: Lee Yu Jie Melvin
- Release Engineer & Manager: Sannabhadti Shipra Deepak

2.2 Roles and Responsibilities

Project Manager: Bian Hengwei

- Oversees project progress
- Approves and executes project plan
- Assigns tasks and reports status of project to team members
- Manages and motivates team members
- Represents the team to the outside world

Lead Developer: Renganathan Ramasamy

- Overall technical lead, leads backend and frontend developer
- Responsible for technical aspects of product release
- Creates requirement specification document

Front-end Developer: Jin Han

- Designs user interface of application
- Developing features to enhance user experience
- Maintain brand consistency throughout design

Back-end Developer: He Yinan

- Integration of user-facing elements with server side logic
- Database integration and management
- Ensure high performance and responsiveness to requests from front-end

Quality Assurance Engineer & Manager: Lee Yu Jie Melvin

- Ensures acceptable software quality
- Ensures proper implementation of the quality assurance process.
- Designs testing strategies
- Creates and manages test plan
- Verify software requirements
- Executes test procedures

Release Engineer & Manager: Sannabhadti Shipra Deepak

- Identify the configuration items
- Manage configuration records and release of product
- Create baselines and build and integrate changes for delivery.

2.3 Team Communication

Team Lucks' communication channels include the following:

- Weekly Zoom meetings, usually held on Sundays.
- Group announcements and updates are sent through Telegram group.
- Jira Software and Github to keep track of individual responsibilities.
- Split up into subgroups as necessary, in order to work more cooperatively on specific problems.

3 Process Definition

3.1 Lifecycle Model

Team Luck plans to use the Iterative and Incremental Model for developing Drawdiculous. Compared to the traditional Formal Document Driven Models such as the Waterfall Model, while the latter provides limited flexibility and makes it difficult to change scope, Iterative and Incremental Model benefits the team by allowing partial development in each iteration, and subsequently allowing flexible adjustment of the project scope and design according to the time and resource constraints.

Team Luck decides to start with a minimum working product, and add functionalities to the product during each iteration. Taking advantage of the Incremental Model, we could revisit our system to add more functions if we have extra time. This ensures that we can have a working demo before the deadline.

Team Luck intends to develop basic client functions and basic backend in parallel during the first iteration. The code is then merged for full functionality. Then, further iterations should occur as necessary.

Further iterations could work on improving the throughput of the game server, allowing more users to play at the same time. More interesting game mode may also be implemented for better game experience.

4 Schedule

4.1 Activity Dependencies and Schedule

There are five phases for our project, Project Initiation, Project Planning, Project Execution, Project Testing and Project Close-Out. We divide our phases into very detailed subtasks with detailed start and end time to ensure that our project is well managed.

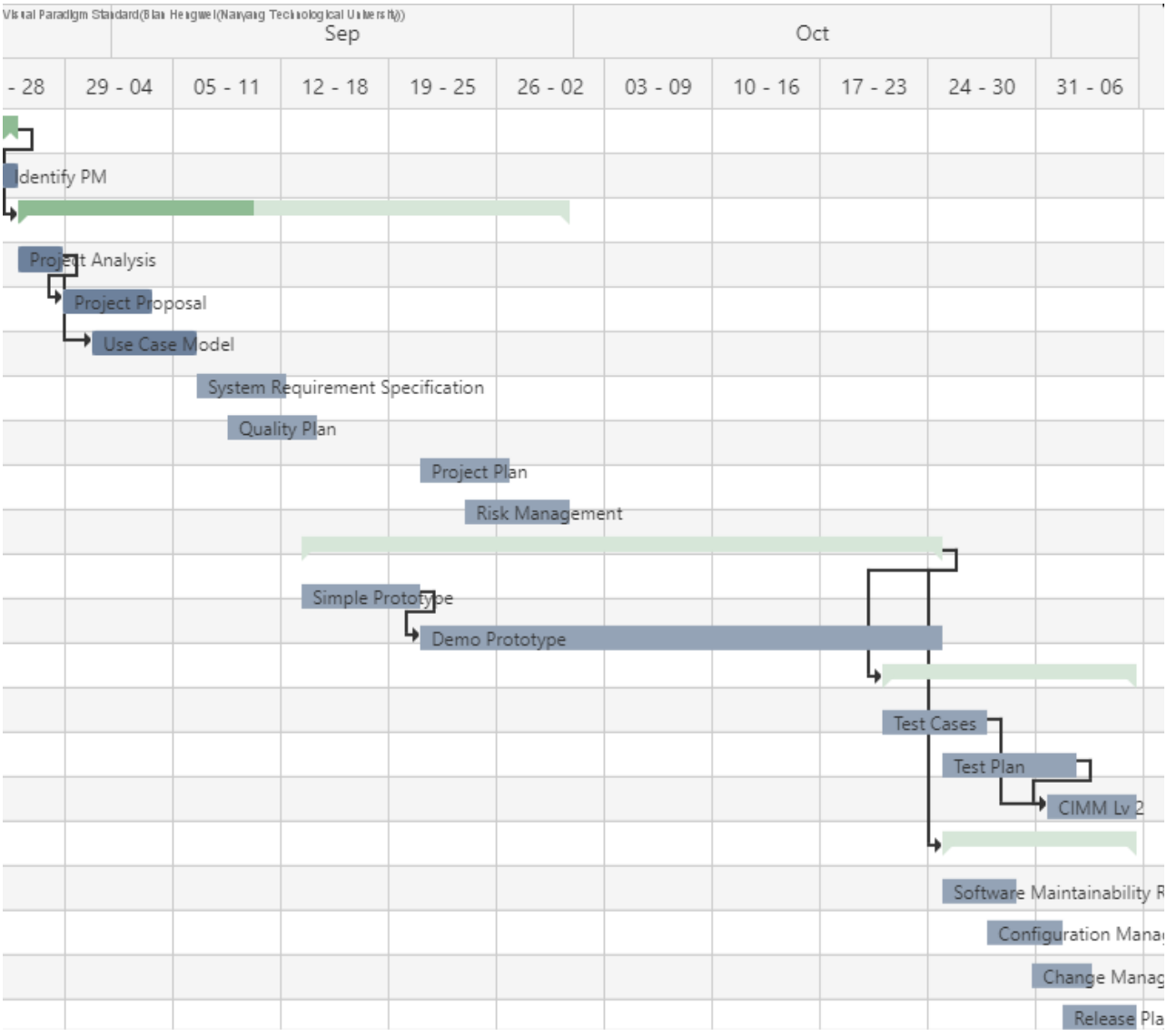
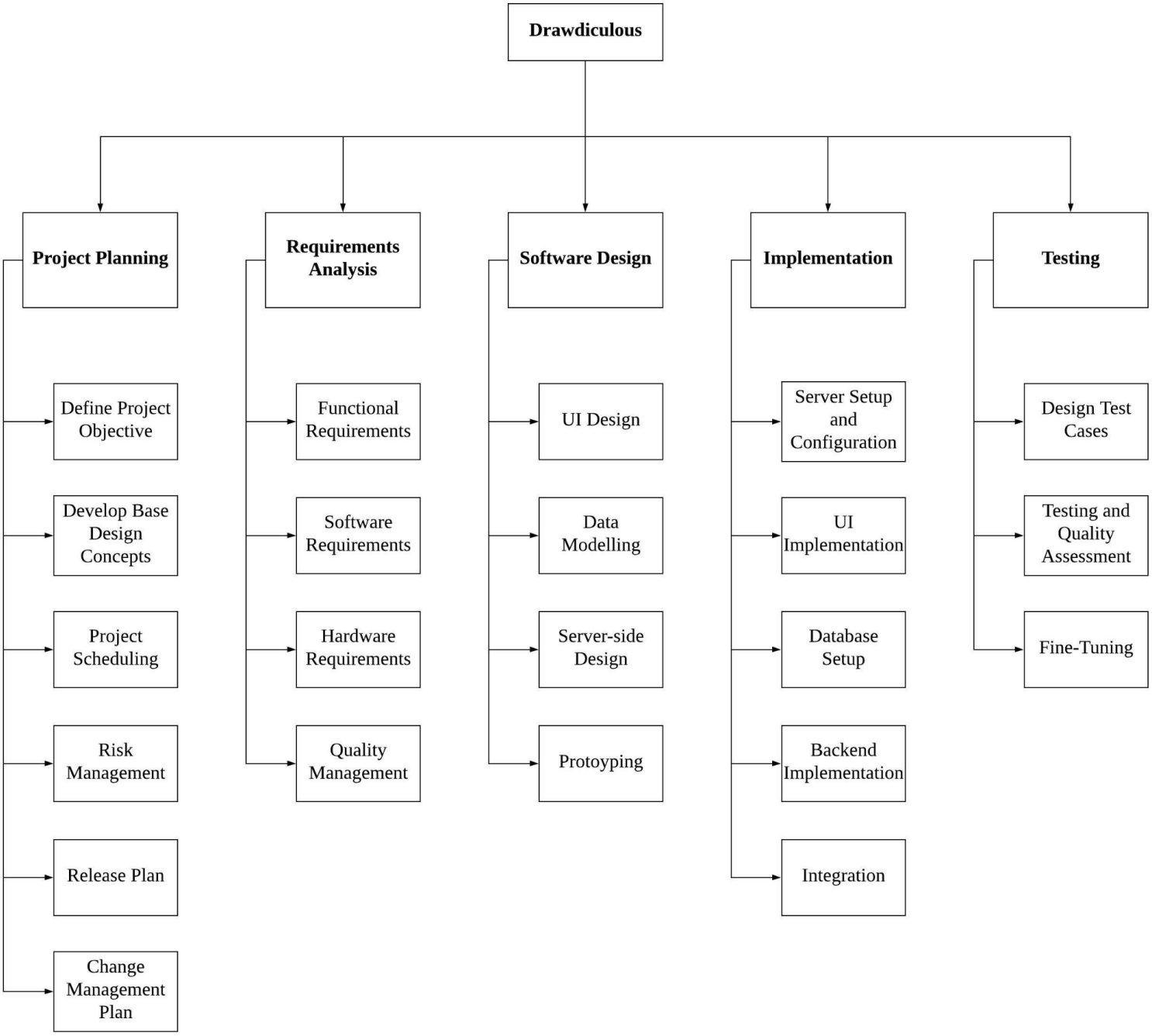


Figure 1: *Gantt chart for the project. The solid bars indicate the portions of the tasks that we have accomplished. Go to Gantt Chart.png for a clearer image.*

4.2 Work Breakdown Structure



4.3 Work Packages

The entire project work is broken down by the important phases of the software development life cycle. They include the following:

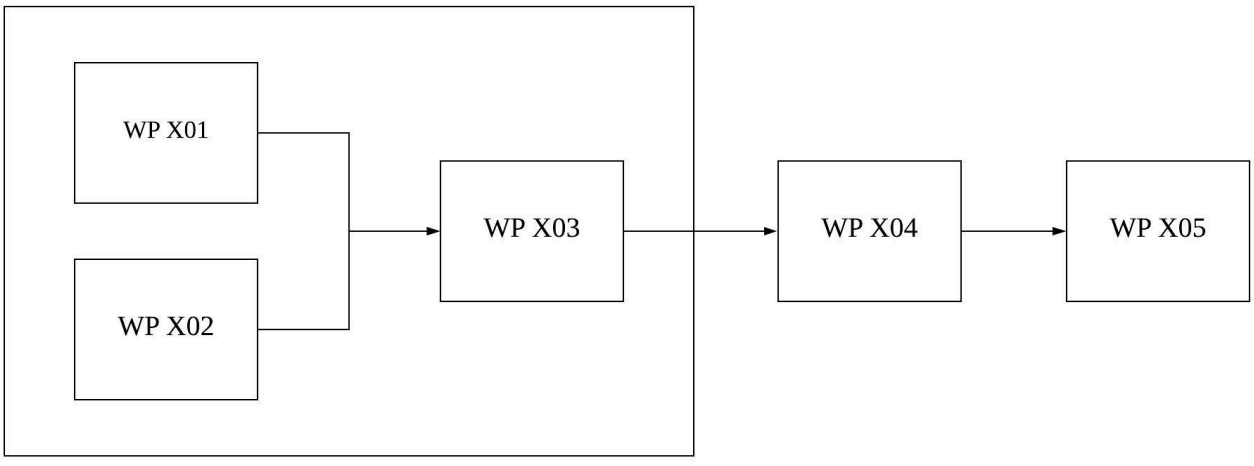
1. Project Planning
2. Requirements Specification
3. System Design
4. Software Development
5. Testing & Quality Assurance

4.4 Activity Dependencies

The following table describes the dependencies of the deliverable work packages:

Work Package #	Work Package Description	Duration	Dependencies
X01	Project Planning	14 days	--
X02	Requirements Specification	10 days	--
X03	System Design	14 days	X01, X02
X04	Software Development	25 days	X03
X05	Testing & Quality Assurance	10 days	X04

The following Activity Network Diagram describes the above in more graphical detail:



Note that work package X05 is dependent on all work packages encapsulated by the larger boxes linked to its left. For instance, WP X05 may not start until WP X01-X04 has been finished.

4.5 Work Package Details

Work packages are listed below. The team members indicated in bold will be deemed primarily responsible for the relevant work package.

Project	Drawdiculous Android Application
Work Package	X01— Project Planning (1 of 5)
Assigned To	**Bian Hengwei, He Yinan, Jinhan, Melvin, Rama, Shipra**
Effort	14PD
Start Date	26/08/2021
Purpose	To determine an overview of the project. Clarifies relevant details such as schedule, resources and risk before official project initiation. To be refined in later work packages.
Inputs	None
Activities	Brainstorm project concept, finalise an overview for the project. Discuss project details such as schedule, resources, estimated cost and risk. Finalise deliverables in each stage of the software development lifecycle. Draft discussions from group meetings into a formal report.
Outputs	Relevant Project Plan documents.

Project	Drawdiculous Android Application
Work Package	X02— Requirements Specification (2 of 5)
Assigned To	**Bian Hengwei, He Yinan, Jinhan, Melvin, Rama, Shipra**
Effort	10PD
Start Date	01/09/2021
Purpose	To establish an understanding of elderly customer’s needs in order to adequately address their requirements in the project design.
Inputs	Customer’s Requirements
Activities	Identify the customer and inspect the customer’s requirements, in addition to determining base-level build requirements.

Outputs	A written document of requirements specification.

Project	Drawdiculous Android Application
Work Package	X03— System Design (3 of 5)
Assigned To	**Bian Hengwei, He Yinan, Jinhan, Melvin, Rama, Shipra**
Effort	14PD
Start Date	11/09/21
Purpose	To determine the high-level architecture , user interface and database design.
Inputs	Work Package X01-X02
Activities	The team will decide on the system's software infrastructure, operating system requirements, user interface and design the flow of the application. Relevant diagrams will cement the final design choices.
Outputs	High Level design and architectural specification (in the form of diagrams), UI and Database outline.

Project	Drawdiculous Android Application
Work Package	X04— Software Development (4 of 5)
Assigned To	**Bian Hengwei, He Yinan, Jinhan,** Melvin, **Rama, Shipra**
Effort	25PD
Start Date	25/09/21
Purpose	To implement the system based on the requirements and design specifications.
Inputs	Work Package X01-X03
Activities	Programmers will implement the relevant frontend and backend modules according to the requirement and design specifications. They will then integrate the modules to produce a preliminary version of the application.
Outputs	Application source code.

Project	Drawdiculous Android Application
Work Package	X05— Testing & Quality Assurance (5 of 5)
Assigned To	Bian Hengwei, He Yinan, Jinhan, **Melvin**, Rama, Shipra
Effort	10PD
Start Date	20/10/21
Purpose	To create unit tests in order to identify and fix application errors and ensure a quality product.
Inputs	Work Package X01-X04
Activities	This work package includes developing unit and integrated tests for the application. The application will be fine tuned at this stage. The configuration management and tests plans will also be created.
Outputs	Test code and report.

5 Project Estimates

5.1 Code Size Estimation using Function Points

We calculated unadjusted function point based on the complexity of functions provided by this system. Code size is then estimated by adjusted function point.

5.1.1 Unadjusted Function Points

Drawdiculous android application supports the following proposed functions:

User:

- User authentication
- Create a game room
- Join a game room
- Leave a game room
- Play the game
- View players in the room
- View leaderboard of the game

Room admin:

- All user functions
- Remove players in the room
- Start a game

The measure of unadjusted function points is based on five primary component elements of these functions: Inputs, Outputs, Inquiries, Logical Files, and Interfaces. Each element ranges from Low Complexity, Medium Complexity to High Complexity. The detailed evaluation of the complexity is as follows:

Rating Inputs:

- Gather user information (i.e. user email, user password)
- Gather room information (i.e. room ID, room password)
- Gather player answers

Files Type Referenced (FTR)	**Data Elements**		
	1-4	5-15	Greater than 15
Less than 2	Low (3)	Low (3)	Average (4)
2	Low (3)	Average (4)	High (6)
Greater than 2	Average (4)	High (6)	High (6)

Rating Outputs:

- Display list of players in game
- Display game leaderboard
- Display game prompt

- Display player answer attempts
- Display player drawings in real time
- Display player room ID
- Display game timer

File Types Referenced (FTR)	Data Elements		
	1-5	6-19	Greater than 19
less than 2	Low (4)	Low (4)	Average (5)
2 or 3	Low (4)	Average (5)	High (7)
Greater than 3	Average (5)	High (7)	High (7)

Rating Inquiries:

- Select actions to perform in home page (create game, join game, join random game)
- Select actions to perform in room page (start game, leave room)

File Types Referenced (FTR)	Data Elements		
	1-5	6-19	Greater than 19
less than 2	Low (3)	Low (3)	Average (4)
2 or 3	Low (3)	Average (4)	High (6)
Greater than 3	Average (4)	High (6)	High (6)

Rating Logical Files:

- User account saved in phone
- Appconst file
- Server const file

Record Element Types (RET)	Data Elements		
	1 to 19	20 - 50	51 or More
1 RET	Low (7)	Low(7)	Average (10)
2 to 5 RET	Low (7)	Average (10)	High (15)
6 or More RET	Average (10)	High (15)	High (15)

Rating Interfaces:

- Firebase API
- Client - server interface

Record Element Types (RET)	Data Elements		
	1 to 19	20 - 50	51 or More
1 RET	Low (7)	Low(7)	Average (10)
2 to 5 RET	Low (7)	Average (10)	High (15)
6 or More RET	Average (10)	High (15)	High (15)

Summary of above analysis:

Element	Complexity	Detail
Inputs	Low	Gather user information
	Low	Gather room information
	Low	Gather player answers
Logical Files	Low	User account saved in phone
	Low	Appconst file
	Low	Server const file
Outputs	Low	Display list of players in game
	Low	Display game leaderboard
	Low	Display game prompt
	Medium	Display player answer attempts
	Medium	Display player drawings in real time
	Low	Display player room ID
	Low	Display game timer
Inquiries	Low	Select actions to perform in home page
	Low	Select actions to perform in room page
Interfaces	Low	Firebase API
	Low	Client - server interface

Calculation of Unadjusted Function Points:

Characteristic	**Low**		**Medium**		**High**	
Inputs	3	× 3	0	× 4	0	× 6
Outputs	5	× 4	2	× 5	0	× 7
Inquiries	2	× 3	0	× 4	0	× 6
Logical Files	3	× 7	0	× 10	0	× 15
Interfaces	2	× 5	0	× 7	0	× 10
Unadjusted FP	66		10		0	
Total=L+M+H	76					

5.1.2 Adjusted Function Points

Influence Factors	Score	Detail
Data Communications	5	Application consists of frontend and backend. Frontend and the server communicate to do the basic functions of the application.

Distributed Functions	0	Distributed process is not used in the application
Performance	4	Response time is critical at all times to achieve the real time display of drawings. Data transfer for achieving real time drawing display is designed to achieve high performance.
Heavily used	1	Non-blocking input output is used to manage more users at the same time.
Transaction rate	0	Transaction peaks are not considered.
On-line data entry	2	100% of data entry are on-line data entry as Drawdiculous requires the internet to operate.
End-user efficiency	2	Application is designed to achieve high efficiency for the end-user.
On-line data update	2	Online update of internal logical files is included.
Complex processing	0	No extensive logic processing or complex calculation is done.
Reusability	2	The application was specifically packaged and/or documented to ease re-use, and the application is customized by the user at source code level. bull shit
Installation Ease	0	No special setup is needed to install the application.
Operational Ease	1	Effective start-up is provided. Users stay logged in and do not need to login again unless the user logs out of the application.
Multiple sites	0	User requirements do not require considering the needs of more than one installation site.
Facilitate change	3	Application is designed and developed to support easy change and update.
Total score	22	
Influence Multiplier = Total score × 0.01 + 0.65 = 22 × 0.01 + 0.65 = 0.67		
Adjusted FP = Unadjusted FP × Influence Multiplier = 76 × 0.67 = 50.92		

Scoring (0 – 5)
0 = No influence
1 = Insignificant influence
2 = Moderate influence
3 = Average influence
4 = Significant influence

5 = Strong influence

5.1.3 Lines of Code

According to Capers Jones statistics, each Function Point requires 53 lines of code if the application is implemented using Java.
Therefore, we have:

Lines of Code = 50.92 FP × 53 LOC/FP = **2698 LOC**

5.2 Efforts, Duration and Team Size Estimation

To estimate the effort and duration required for the project, we use function points as the basis to calculate Effort, Duration, Team size and finally the schedule. The estimates are expanded to account for project management and extra contingency time to obtain the total average effort estimates. From these averages, the duration of each work package in working days is estimated based on the following calculations.

- Working days include 5 days in a week.
- Effort = Size / Production Rate = (2698 LOC) / (45 LOC/PD) = 60 PD
- Duration = 3 × (Effort) $^{1/3}$ = 3 × (60) $^{1/3}$ = 11.7 Days
- Initial schedule = 11.7 Days / 5 days a week = 2.35 Weeks
- Team size = 60 PD / 11.7 D = 5.12 P = 6 Persons
- Working hours include 8 hours in a working day.
- Total person-hours (PH) = 60 PD × 8 hours = 480 PH

5.2.1 Distribution of Effort

1990's Industry Data	**Work Package**	**Distribution**	**Estimates**
Preliminary Design 18 %	Project Plan	10%	48.00
	Requirement Specification	8%	38.40
Detailed Design 24 %	User Interface	8%	38.40
	Technical Architecture	10%	48.00
	Data Modeling	6%	28.80
Code & Unit Testing 29 %	Code & Unit testing	22%	105.60
	Online Documentation	7%	33.60
Integration & Test 29 %	Integration & Quality Assurance	29%	139.20
	Extrapolated total effort		480
	2% for project management		9.60
	3% for contingency		14.40
	Total effort		504

These duration estimates are based on the assumption that each team member works an equal amount on any given work package.

5.3 Cost Estimates

Hardware:

Developer workstation:

6 - Lenovo laptop	$9,594.00

Software:

Server:

Ubuntu Server	$303.44

Software License Provided by Third Party:

Visual Paradigm	$1,999.00
IntelliJ IDEA Ultimate	$649.00

Other Resources:

Staff:

1 Product manager with 504 working hours and $80.00/hour	$40320.00
5 Employees with 504 working hours with $48.00/hour	$120,960.00

Total:

$173825.44

The users will supply the required hardware and software necessary to run Drawdiculous application. Team Luck is not responsible in any way for supplying said systems. Drawdiculous's hardware and software responsibilities relate only to our own development needs to accomplish the project, which has been described in the introduction section of this document. Drawdiculous will demonstrate the completed product.

6 Product Checklist

The plan is that the items listed below will be delivered on the stated deadlines.

Project Deliverable	Estimated Deadline
Project Proposal	Sept 8th, 2021
Quality Plan	Sept 21st , 2021
Requirement Specification	Sept 30th, 2021
Project Plan	Oct 12th , 2021
Simple Prototype	Oct 14th , 2021
System Release (Demo)	Oct 27th, 2021
All wiki documents compiled	Oct 30th 2021
SVN Documents checked in	Oct 30th 2021

7 Best Practice Checklist

Practice	11
We will document everything properly while we develop our Drawdiculous Application. All the documents will be standardised.	
We will pay attention to requirements, check for ambiguity, completeness, accuracy, and consistency. The requirement documentation will contain a complete functional specification.	
Drawdiculous Application will be kept simple. Complexity management is one of the major challenges. Strive to: • Minimize interfaces between modules, procedures and data. • Minimize interfaces between people, otherwise exponential communication cost • Avoid fancy product functions, design as long as the functionality meets the customer requirements	
We will ensure Visibility. We must see what we build otherwise we can measure the progress and take management action. This includes: the team leader must have good communication with his or her team members; require developers to make code available for review; review design for appropriateness.	
Drawdiculous team will also plan for continuous change. We must: • All manuals designs, test, source code should have revision numbers and dates revision history comments, change marks to indicate the changes • New revisions should be approved before being made and checked for quality and compliance after being made • Use a configuration management system and make processes • Required maintenance	
We will set realistic estimates. We must be careful to obtain realistic estimates for: time, effort, overhead, meeting time, and especially effort on integration, testing, documentation and maintenance.	
Code reviews are a much more efficient method to find software defects. Plan and manage code reviews between team members	
Software testing will use both black box and white box testing. It will involve unit, functional, integrating and acceptance testing.	
We will ensure that we meet up regularly every week to discuss any updates regarding the Drawdiculous project. This will help to ensure that all our members are up to date in what is happening.	
We will also make sure that we test the software properly by deliberately testing with bugs and making sure it fails.	
We will consult our teachers regularly so that we can be confident that we are on the right track in developing our application.	

8 Risk Management

The following risks have been identified for the Drawdiculus project.

S/N	Risk Identification		Risk Evaluation			Risk Control				Follow Up & Dates	Remarks
	Risk	Description	S[1]	P[2]	RPN[3]	Measured Taken	S[1]	P[2]	RPN[3]		
1	More changes to the requirement than anticipated	Depending on the stage at which changes occurs, could range from needing to update the requirement documents for a complete design	4	3	15	Be rigorous in designing all possible scenarios/requirements. During the review stage, ensure customers are aware of potential repercussions of requirements changes.	4	1	4		
2	Specification Delay	Delay in specification will push the schedule for subsequent stages	4	1	4						Monitor the progress at all time through frequent meetings and updates

Notes:

[1] - Severity of the risk

[2] - Probability of the risk happening

[3] - Risk Probability Number

S/N	Risk Identification		Risk Evaluation			Risk Control				Follow Up & Dates	Remarks
	Risk	Description	S[1]	P[2]	RPN[3]	Measured Taken	S[1]	P[2]	RPN[3]		
3	Underestimated system size	More time will be spent to recode and redesign.	3	3	9	Provide estimates as we goes along to prevent exceeding	3	2	6		
4	Poor teamwork coordination	Members are unaware of what is expected and hence unable to measure progress properly.	3	3	9	Improve briefing & communication, team leader / manager should remind the requirement	3	2	6		
5	Database / Server failure	Engineers/Players are unable to access the server/database for information retrieval and storage	4	3	12	Create multiple instances of server/database, with backup frequently done for rollback	2	2	4		
6	Staff leaving	Other members will need to do the workload of others, specialised skills or knowledge may be lost.	5	1	5	Offer benefits and incentive	5	1	5		
7	Cancellation of project	Work wasted	5	1	5	Keep closed contact with customers. Ensure market research had been done before commencing this project	5	1	5		

9 Quality Assurance

The project will achieve quality assurance by following the standard set by the company. The specific procedures and details are provided in the Quality Plan. Specific test procedures and details shall be provided in the Test Plan.

In addition, Drawdiculus will make use of the following testing methodologies:

- Unit testing involves testing system components individually; and
- In-place testing involves testing of the whole system as a unit.

Furthermore, these methodologies will be used to test the 2 important aspects of the application:

- System function will be tested to ensure all software flaws are eliminated;
- Algorithmic function will be tested to ensure that heuristic aspects of the project, such as leaderboard ranking, perform correctly.

Drawdiculus's methodology makes broad use of realistic test cases. Detailed test data is an important part of the final project delivery. Drawdiculus will attempt to provide a comprehensive list of words in multiple languages for testing and actual gameplay purposes. In addition, Drawdiculus will validate code and heuristic leaderboard ranking using realistic gameplay scenarios. Extreme cases (such as cheating) will be used to ensure that the system detects ingame cheaters to prevent a high-score leaderboard.

10 Monitoring & Control

Many procedures are required in order to be able to successfully monitor the progress of Drawdiculous. Some of the most important are:

Assuring product quality: Overall quality of the product is tracked and maintained by Team Luck. The Quality Assurance section in this document briefly specifies some of the important methodologies that will be used to test/maintain the application quality. More detailed plan is written in the Quality Plan document.

Identification of major project risks: Major risks are identified during the design phase of the project. They are evaluated to identify how to deal with these risks and to prevent any possible crucial risks from happening. The Risk Management section identifies those major risks and lists possible measures needed to be taken to avoid the risks.

Regular reviews of project progress: Throughout the duration of Drawdiculous project development, Team Luck is meeting at least twice every week to keep track of the progress of app development, documents, testing, etc.

Timeline planning and task decomposition: An estimated schedule for this project is included in the Schedule section of this document. We also decompose each task into subcomponents to ensure accurate time estimation of the whole task. Meanwhile, the decomposition of tasks also helps with planning the tasks and balancing the workload over time.