Team 15  
Project Plan

short line

Table of Contents:

[Project Scope: 2](#_f8vroxg94z7u)

[Deliverables: 2](#_l2nrununoj1n)

[Project Process Plan: 3](#_4bhn5mxy2jvn)

[Design Document Plan: 4](#_hgqitvrgmhy2)

[Code Base Plan: 4](#_oigu4c10jpta)

[Testing Schedule Plan: 4](#_yp5xpv8wg8jq)

[Group Report Plan: 4](#_453pb3tfqzj6)

[Peer Assessment Plan: 4](#_o95edt6hoah5)

[Video Plan: 5](#_q1h9vcwzp582)

[Time Plan: 5](#_v6opymz2kaku)

[Resource Plan: 5](#_z26klwaxouaq)

[Risk Analysis: 6](#_cdur2upj2uyo)

[Requirement Analysis: 7](#_1ah1tjdw6kz)

Team Members:

Noah (230986)

Yash (249763)

Eddie (246498)

Nelson (246731)

Ryan (246663)

Morgan (246709)

# Project Scope:

The project scope refers to the deliverables which our team aims to produce at the end of this project. The main deliverable/ objective is to deliver a computer game version of the board game “Catan”, within this main deliverable, there are quite a few smaller objectives, which we will identify and attempt to plan the execution of them.

One deliverable is to develop a software development process document using some elements of the methodology, Agile, potentially with a hint of Waterfall, another model used for software development. To generate this document, we will be completing a number of sprint cycles, which will be documented. These sprint cycles will allow us to have working prototypes for feedback until our client (Watson Games), is satisfied with the game.

Another deliverable is to provide a plan for the design, implementation, testing, and delivery of the software system. In order to complete this, we are going to document our progress and revise any plan simultaneously. The plans will include, risk analysis, time and cost estimates, resources, and most importantly the deliverables. The design document will include a number of low-level and high-level designs of the software system, examples include, class diagrams, UMLs, E/R Diagrams, etc. The implementation plan will include our decision on which language to use, roles and responsibilities, and resources that we would use. As a team, we will make sure testing is done effectively and well-documented. For the delivery of the game, we will take a short video, showcasing the system and game states, including extra features and a voiceover.

# Deliverables:

Below we created broad tasks for the required deliverables:

1. Project Plan

* List of tasks identified
* Effort Planned for each task identified
* Schedule (PERT or Gantt)
* Team member assigned for each task
* Updating Plan if executions were off predictions
* Submit revised and initial plans
* Commentary on why the revision was needed

1. Project Process Document

* Sprint Document
* Sprint Document including a record of action for each phase

Each Phase should have:

* Target Objective
* User Requirements
* Functional, non-functional or domain Requirements
* Key design (if applicable)
* Implementation decisions
* Review on the level of success

1. Design Document

* Drawing of Low-level designs
* Drawing (Physical or Digital) high-level designs
* UML style Diagrams
* Class diagrams
* Sequence Diagrams
* Activity Diagrams
* State transition diagrams
* Design for each incremental phase (if phase requires one)

1. Code Base (Implementation)

* Code files and Non-Code files
* Check if files are easily compiled and executed by others
* Code be documented and commented
* Snapshot the codebase which was related to phase

1. Testing Schedule

* Showing how we tested the software (Unit testing and System Test levels)
* Identify and Document bugs (can remain in the software)
* Bugs found are to be explained and what we would do about them

1. Group Report

* Document how the project went for you (Individually) and critical analysis of the performance of your team (What went well and Even Better If (WWW and EBI))
* Group meeting documents included

1. Peer Assessment

* Each person is assessed and marked based on their contributions as a whole

1. Video

* Evidence of the final working state of the program
* includes a voiceover to narrate any features

## Project Process Plan:

During our Agile sprint cycles we are expecting to split our group into roles such as designers, programmers, testers, and project managers which will be in charge of the documentation of the whole cycle. We also plan to rotate around these roles, so that everyone gets a taste of each role and also allow everyone to have an equal opportunity to contribute to the project.

Agile Sprints Plan:

As a team, we will assign each other to the appropriate roles and identify tasks to pursue in our sprint cycles in order to create a working prototype. After we have identified the tasks we will get our designers to start creating class diagrams and APIs for some of the key classes we will have. We may also create low-level designs of our prototype to aid the objectives of the sprint. The design phase will also allow us to focus on each of the class’ requirements, supporting the documentation process and our programmers to work together.

We will then move on to the main execution of the sprint cycle, the development of the prototype. Our strategy will involve pair programming. This is where we will get two programmers and they will code together parts of the tasks for the prototype. This will allow knowledge to be spread across the team and boosts refactoring. Currently, our strategy for testing is the TDD approach, however, this could change and we will revise the plan as we make any changes to our decisions. We will surely do some unit-level testing and system-level testing. Finally, our project managers will continue to monitor each phase, review the cycle at the end and get all the documentation done, getting the team to deliver the working prototype for feedback.

## Design Document Plan:

This document will be worked on simultaneously with the sprint cycles, Noah and Eddie will be our main designers, The document will be split into low-level designs and high-level designs. It will discuss the role of each high-level component in the designs and attempt to have low-level designs such as class diagrams that correspond to the code base.

## Code Base Plan:

This will be a folder for the game to be placed in. It will consist of the main class, and other helper classes, each class will be carefully documented with DocStrings as we are going to use Python to build Catan. We are planning to assign Morgan and Ryan as the main programmers, and Noah and Eddie assist with the documentation.

## Testing Schedule Plan:

This is going to be the main document in which we will provide evidence of tests for our prototypes. Eddie and Noah will be able to discuss any issues with code, System level testing aimed to reflect user stories and unit tests should reflect design documentation.

## Group Report Plan:

This document will be used as a place where each team member will reflect on the performance of each other, in terms of what went wrong and well. This document will also have all the team meetings we had with details about what each meeting covered and the next steps. This document is aimed to have a critical analysis of each key achievement and/or key issue, for each team member. This is so that we can assess what could have been done differently or how they achieved success. We plan to start the group report a week before the submission deadline, so we can cover the whole lifetime of the project, however, the team meetings will be held each week, to cover any risks, any changes to the project, and be able to communicate progress. The team meetings will be taking place mostly in person, and they will be led by Yash and Nelson.

## Peer Assessment Plan:

This is going to be a simple document noting each person’s score on how they performed in the project, each member of the team will score their peers in terms of their performance.

## Video Plan:

Near the end of the project lifetime (preferably a working week before the submission date), a walkthrough of the final project will be recorded by Ryan with him giving a voiceover to explain some features of the game.

# Time Plan:

This project will have a lifespan of 11 weeks, A Gantt chart has been created to give a visual representation of some milestones needed to complete the project. There will be more specific time plans for each Sprint cycle in the Project Process Document. The Gantt Chart is in this zip file as a pdf. We plan to have 3 to 4 Sprint cycles, which will last between 1 to 3 weeks on each cycle. This will leave us with enough time to also review our project in the last week. We are currently in the first week and we expect to use the remainder of this week and the next two weeks to complete any plans. The plan is to create PERT Charts or Tables for each sprint cycle in order to stay on track with the cycle and derive any critical paths for decision-making.

# Resource Plan:

As we have now identified the scope of the project, we can now look at the resources that we will need for the project. Different types of resources such as expertise, software, hardware, and other generic supplies. Here is a table of resources we need for this project:

| Resource | Resource Type | Use of Resource |
| --- | --- | --- |
| Programmers, etc | Staff | Code Generation |
| PyCharm | Equipment/ Software | Game development |
| GitHub/ Drive | Equipment/ Software | Documents |

GitHub and Drive:  
GitHub is going to be our main platform in which we will use to share our code on, we are using this as we are all familiar with this software and it offers great tools in which we can work more efficiently such as their pull request features. We will also use a Kanban View to create task cards and assign one another to different tasks. A Kanban approach will help us be more proactive and have a clear task to work on. Google Drive will be our primary location to store our documents, this is due to the fact that we can easily share, contribute and access the documents.

Experience:  
We are Team 15, the team members are Nelson, Eddie, Morgan, Noah, Ryan, and Yash. We all have our strengths and weaknesses and for our project to be successful we determined who will be responsible for what. Nelson and I will be the main Project Managers, this is because we both are the least confident programmers in the group. Morgan and Ryan will be the programmers, as they are the most experienced and enthusiastic members of the team regarding coding. Eddie and Noah will both work together to complete the Testing and Designing phases of each cycle.

# Risk Analysis:

We also aim to make a thorough analysis of the risks which might arise during the process of each cycle. We will predict any risks before the execution of the cycle and review them during each task of the cycle to assess how likely each risk will occur and its impact on the process. Here are some risks which were identified for the project as a whole:

1. Risk Identification

* Financial Risks: High costs, unrealistic budget/overspending, additional labor and resources, low sales
* Performance Risks: Missed deadlines & deliverables, outdated market research, the underperforming end result
* Strategic Risks: Project management methodology, Project planning, daily operation, project dependencies, use of technology, group culture
* External Risks: Teammate illness, leave-taking, Emergencies, Major weather events,

2. Risk Analysis

* Financial Risks: unlikely to be a risk in our project as our project requires no financial budget.
* Performance Risks: High likelihood to be a big risk. Due to the relative inexperience of team members in software development.
* Strategic Risks: High risk. If we use or implement the wrong methodology in our project, we risk missing key deadlines and having unreliable code. Group culture is also important as we must maintain a functional and cooperative group mentality where everyone is contributing
* External Risks: Low to medium levels of risk. If one of our team members cannot work due to external factors, we should have provisions for dealing with such a situation.

3. Risk Impact

* Financial risk: The impact level will be none.
* Performance risks: Huge impact on the project and will be highly detrimental to the quality of code and documentation.
* Strategic risks: High impact as we would miss key deadlines and team morale would drop, causing the team to be less efficient.
* External risks: Slight impact as long as everyone works an even amount, the workload won't increase a ton for each person if only one person has to stop.

4. Risk Planning

* Financial risk: Nothing of note to plan.
* Performance risks: Plan weekly meetings to monitor the progress of the team and each individual's role within the team. Job for project manager.
* Strategic risks: Ensure everyone has got a defined role in the team and is doing their job well.
* External risks: Ensure everyone has an equal workload so that if one person has to stop then there won't be a huge and sudden shift in workload.

5. Risk Monitoring

* Financial risk: The impact level will be none.
* Performance risks: The project manager should monitor individual and team progress.
* Strategic risks: regular reviews for team progress.
* External risks: Make sure there is regular contact between team members.

# Requirement Analysis:

We believe it’s important to identify requirements that represent the project as a whole. Therefore, we derived a number of requirements and analysed them from the main specification. We split them into functional, Non-functional, and domain requirements, which are types of system requirements. System requirements are the tasks that we obtained from the user requirements. In addition, each cycle will have its own set of requirements, here is the list of requirements for the project as a whole:

| Functional requirements |
| --- |

| Reference | Description | Mandatory/Desirable |
| --- | --- | --- |
| F1(Elements of the software system) | The ability to upload initial data, for customization | M |
| F2(Elements of the software system) | The ability for the simulation to show player’s performance, e.g. assets they own | M |
| F3(Elements of the software system) | Ability in being tested | M |
| F4(The game player agent) | game to have a player agent so the game is playable regardless skilled or not | M |
| F5(The game player agent) | Player agent could be based on random decisions (Initially) | D |
| F6(The game player agent) | Player agent to be able to play respectively like a human | D |
| F7(The game player agent) | If time permits, sophisticate the agent | D |
| F8(Integrity of Game) | Players are not permitted to borrow or lend assets to one another | M |
| F9(Integrity of Game) | the public record of all players’ assets | M |
| F10(Integrity of Game) | Dice to be a fair dice | M |

| Non-Functional requirements |
| --- |

| Reference | Description | Mandatory/Desirable |
| --- | --- | --- |
| NF1(version of game) | Game played until a player reaches 10 victory points | D |
| NF2(version of game) | Game played with a time limit agreed by all players, if time limit reached player with most points wins | D |
| NF3(other user requirements) | Require at least one autonomous computer player agent | M |
| NF4(other user requirements) | Design with flexibility | D |
| NF5(other user requirements) | Game for Desktops, MAC, and PCs, but PC’s at least | D |
| NF6(other user requirements) | Game to be colourful | D |
| NF7(other user requirements) | Intuitive Interface | D |

| Domain requirements |
| --- |

| Reference | Description | Action needed |
| --- | --- | --- |
| D1 |  |  |
| D2 |  |  |

Any Domain Requirements found will be inserted in the table above.