Contents

Introduction

This is a linear path game where you try to make it to the next checkpoint before running out of food. Each decision you make only affects how much time it takes to get to the next checkpoint or your food rations. If you take too long, you'll run out of food and die. There are also special scenarios where you'll get more food, lose food, or randomly die. It all is related to the food countdown or randomized death.

Pick your character

- Banker
    - Starts with the most money ($2000)
    - No useful skills or wilderness experience (only relates to special scenarios)
- Merchant
    - Middle amount of money ($1200)
    - 5-10% discount on purchases
    - Average skills and experience (only relates to special scenarios)
- Farmer
    - Lowest amount of money ($700)
    - Knowledge of plants/wildlife (only relates to special scenarios)

Mechanics behind character selection during the game

Normally, the players proceed through each day with randomized outcomes. However, some days may present a scenario where their character choice will affect their odds of being successful in a special event. Each decision has different odds for different characters depending on the above information. There is a percentage out of 100 that is needed for success assigned to each character in each outcome. A random percentage is generated when approaching the scenario that needs to pass a certain threshold to be successful. For example, “you’ve run out of food and need to forage. Begin searching for food.” A farmer would only need to hit 30% or higher to be successful. A banker would need to hit 70% to be successful. If the randomized percentage falls within the successful range, the character lives and continues on in the game. During development, each special scenario will have these predetermined ranges and the game will run a randomized percentage to determine success.

Money

Money is only used to allow you to make some choices, but they’re all still predetermined. You can only use money to buy food or allow you to pick a different path (purchase a ferry across the river). You can only buy food at certain checkpoints.

Time

Each decision/scenario is one day. There is a counter on the backend that tracks the distance until the next checkpoint, decreasing randomly withing a range with each day traveled. The distance between checkpoints is predetermined, but the distance traveled is randomized. Each day you can choose to travel or hunt for more food. If you chose to hunt, there is no other presented scenario, but you still consume food. Some days there will be no decision when you travel, just a decrease in food as you travel. If you have an “add

an extra day" scenario, the miles counter until the next checkpoint does not decrease. There are 3 checkpoints between start and finish plus a predetermined amount of rivers in between checkpoints.

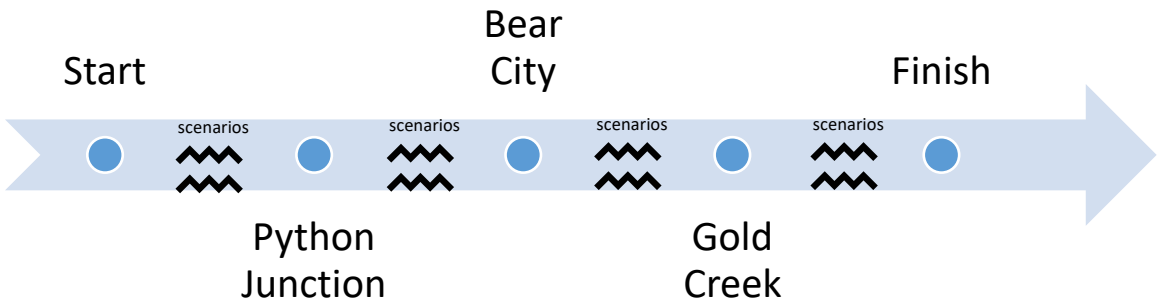
Food

There is a food counter with the amount of food consumed each day. Normally, consumption decreases your food. There may be some scenarios built in that decrease food, as well. For example, one of the day's scenarios may be that a python eats some of your food. You can buy additional food at certain checkpoints using your predetermined amount of money. Food counter at start depends on how much food you purchased.

Travel to destination

The scenarios will be presented by randomization between checkpoints. There will be a set of standard scenarios for the game to present, some requiring a choice and others not requiring a choice. It will randomly select one scenario which, when based on your choice, will add time to your trip, remove food, add food, or kill you directly. Each day is a turn where a player can choose to hunt or travel. If they choose to travel, a scenario will be shown. When the player is successful in the scenario, the distance counter towards the next checkpoint will decrease by a randomized amount of miles in a range.

Checkpoint schedule



You can purchase more food at each checkpoint. It is between $7-9 per food unit. The number of scenarios between checkpoints is determined by the mileage counter and the time that it takes to get to the next checkpoint. At the checkpoints, the merchant gets a randomized 5-10% discount on food units, rounded to the nearest whole dollar. In between checkpoints, there will be a predetermined amount of rivers to cross, so that the player does not continuously get rivers in their randomized scenarios. It will be similar to a checkpoint but without the ability to buy food.

Stretch goals

If time allows, we can make our game further complete. Some of the stretch goals we could aspire to reach include adding wagon parts purchase and breakdown, adding additional robbery scenarios, and adding family members to the initial character set up. These will be explored further if time allows during the project period.

Project Organization

POSITION	NAMES	DESCRIPTION
Organizer	Matteo Catalano	Ensures that the team meets the required task to complete the project and keeps all materials in check. Also unifies the game components and ensures logic across programming. Writes game engine. Testing
GUI Design	Emmanuel Adebanjo	Develops and implements graphics for the game. Testing
Programmer	Nathan Able	Writes the scenario code for the program. Writes game engine. Testing
Programmer	Chris Hochgesang	Writes the scenario code for the program. Writes game engine. Testing. Develops and implements graphics for the scenarios.
Tester	Alvaro Miranda	Analysis and ensures that the program does not have any flaws in its design and that it runs smoothly.

Risk Assessment

RISK	DESCRIPTION	RISK PRIORITY	RISK PLANNING
Schedule Risk (The project cannot be completed and tested in the allocated time)	- Coding the game to full functionality will require a great deal that might not be able to be completed during the allotted time.	High Risk	- The game will be simplified to adhere to schedule. - A detailed schedule will be made with set checkpoints to keep development on track.
Skills Resource Risk (The group does not have the proper skills to complete the project)	- Group coding is still a new skill for most members of the group, and we will be working in a short timeframe.	High Risk	- Our group has 2 or more people who have skills in each of the areas of development, allowing for shared responsibility as well as backfill if necessary. - The team will develop and adhere to a schedule to ensure there is time to complete each part.
Cost Risk (The project will exceed the allotted budget)	- Our investor has agreed to fund a fixed budget, so it is important to manage costs effectively to stay within the budget. - Cost benefit analysis is important when considering paid resources that would save time in development.	Moderate Risk	- Free or low-cost options will be used as much as possible. - Thorough cost benefit analysis will be performed on high budget resources.
Scope Creep Risk (Changes will be made to the project scope during development without proper planning)	- The project is a parody of an existing software. The intent is to be similar in style and not a faithful reproduction. - The project needs to be scaled down to a project that can be completed in a fixed timeframe of development.	Moderate Risk	- The flow of the program will be mapped out prior to coding. This will maintain the structure without the temptation to overly rely on the source materials. - The same flow will help in keeping the content to an achievable level.

Operational Risk (Work and people processes may be insufficient or degraded)	- The team has to set up and figure out a workflow. - Some team members are unfamiliar with group coding. - Unsure of group availability and final size of group.	Moderate Risk	- Several communication and workflow management tools have already been set up (Trello, Discord) to keep everyone connected and on the same page. GitHub and Google Drive will aid collaboration. - Project planning will further connect team members. - Group members have overlapping skills and can cover for others as necessary. - Cloud backups help build resilience in case of natural disaster or other emergencies.
Performance Risk (The product does not meet User Requirements)	- Project may not meet client requirements. - Project may be buggy or otherwise non-functional.	Low Risk	- Should the project be properly completed, risk of not meeting client requirements is low due to having the project approved. - Conversing with clients can ensure requirements are met. - Proper planning can reduce bug risks and ensure requirements are met. - Functionality can be confirmed and improved through rigorous testing throughout the project lifecycle.
Technology Risk (The potential for any technological failure)	- Technology used to develop the project becomes outdated or faulty thereby triggering additional problems. - Technology usually always carries a certain amount of risk, like malware. - Technological risks can result in interruptions, which could slow down our project's operations and processes.	Low Risk	- Team members should be aware of the root causes of the technological risks, the effects of those risks that have been discovered, and the likelihood that the risk will materialize. - Members of the team should keep an eye on each risk factor and be prepared to transition to a different strategy if the technology employed is shown to be ineffective. - Include time to test software and anticipate lags or unexpected delays. - Using up-to-date applications can prevent lags or unexpected delays.

Communication Risk (The group fails to communicate effectively, which leads to mistakes and a project that is headed in the wrong path)	- Establishing a communication vehicle. Communication vehicle is a way we convey information to team members. - Availability of team members.	Low Risk	- We must communicate the project plan, timelines, and requirements plainly and clearly from the team leader down to every member of the team, leaving no space for misunderstanding or error. - Everyone working on the project should be able to access the most recent information by ensuring that the same information is consistently given and kept in the same manner across communication tools (Trello, Discord, Google Drive). - Team members should be informed beforehand of the agendas and the meeting's main objective. - Team members must communicate information, facts, questions, and concerns to all team members in an open and straightforward manner. Each message that needs to be conveyed must be understood completely.

Hardware and Software Requirements

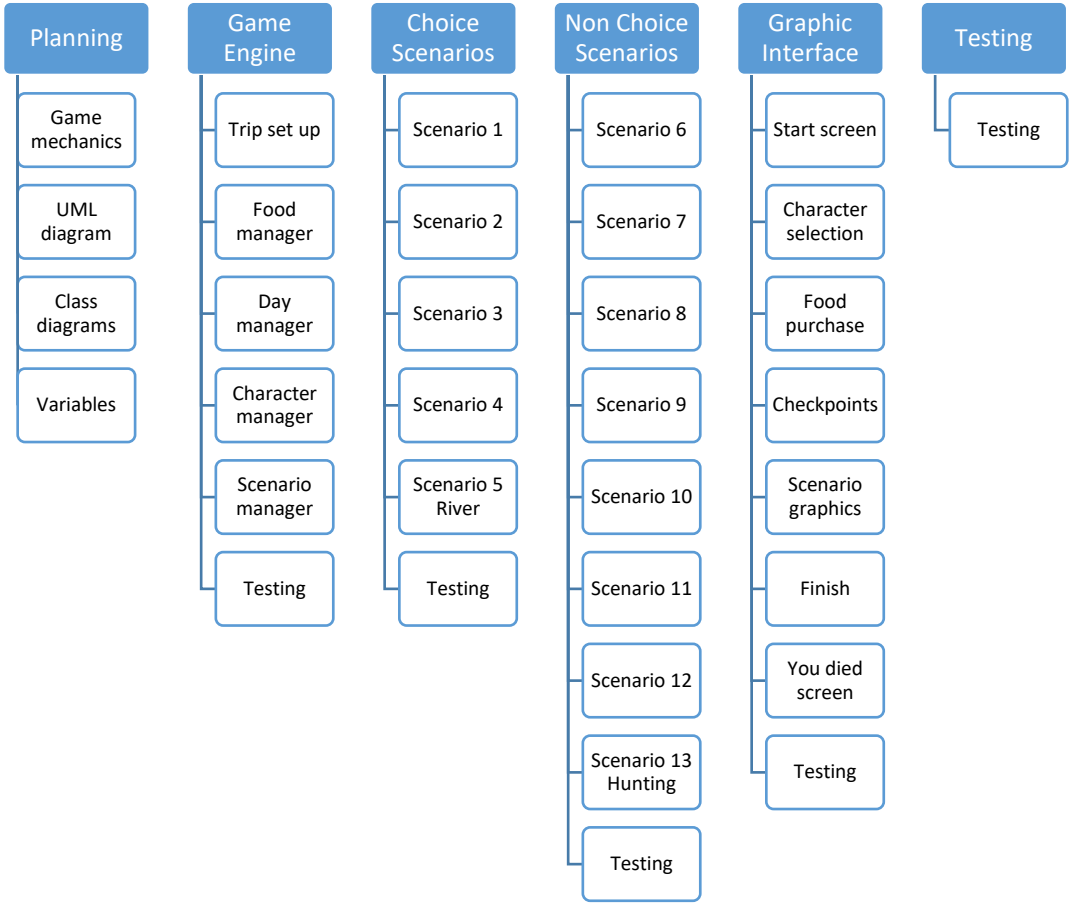
Hardware Requirement

- Windows Computer
- Monitor
- Keyboard and mouse

Software Requirement

- Operating system - Windows
- Python
- Visual studio code or other IDE

Work Breakdown



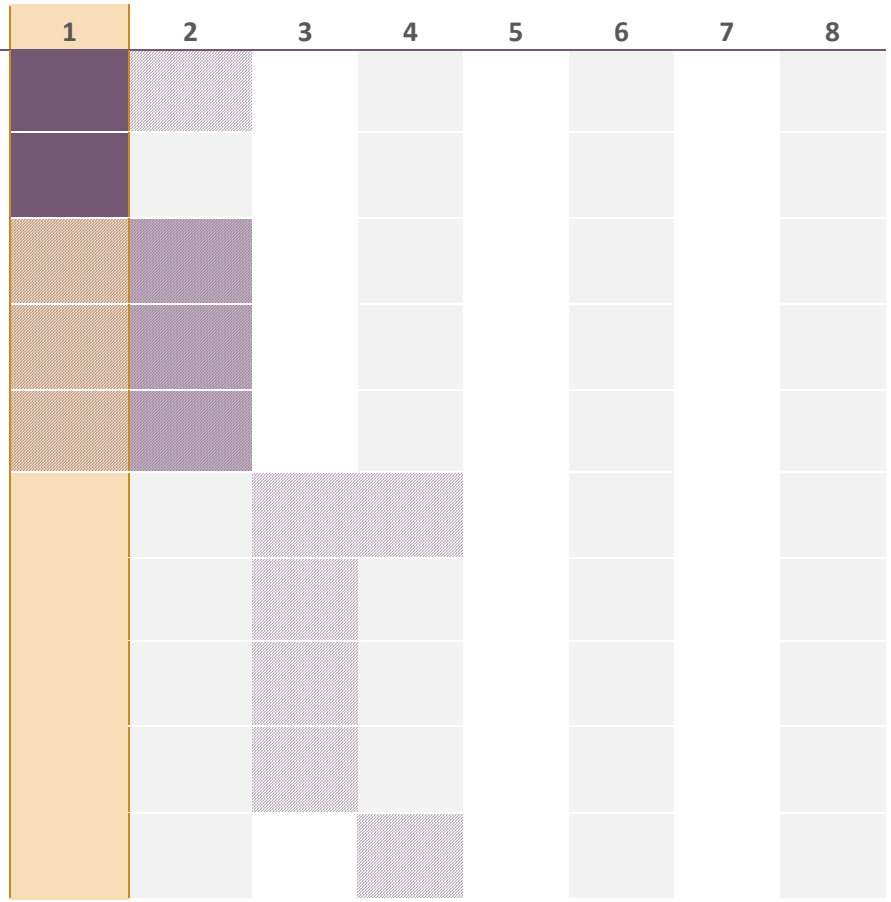
Project Schedule

Project Planner

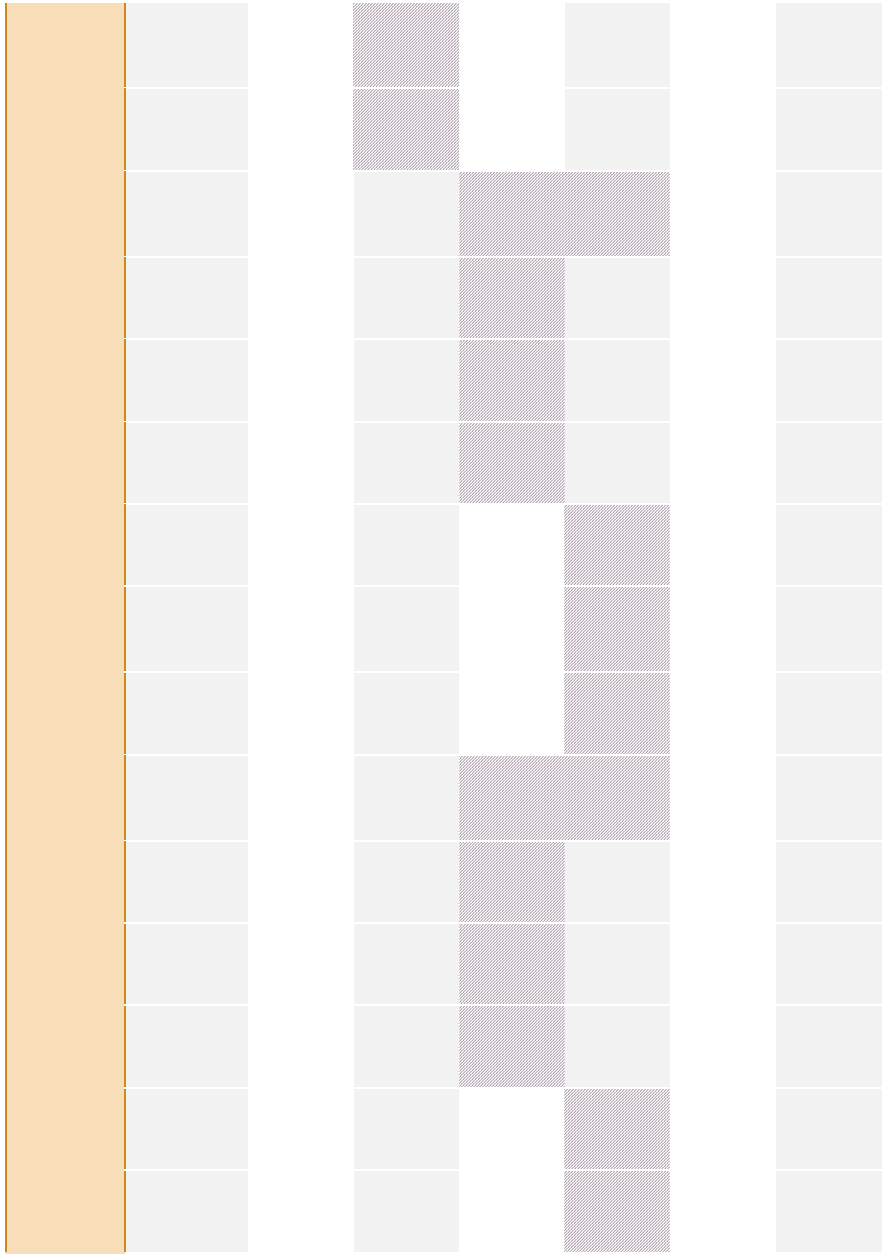
Select a period to highlight at right. A legend describing the charting follows.

Period Highlight: 1

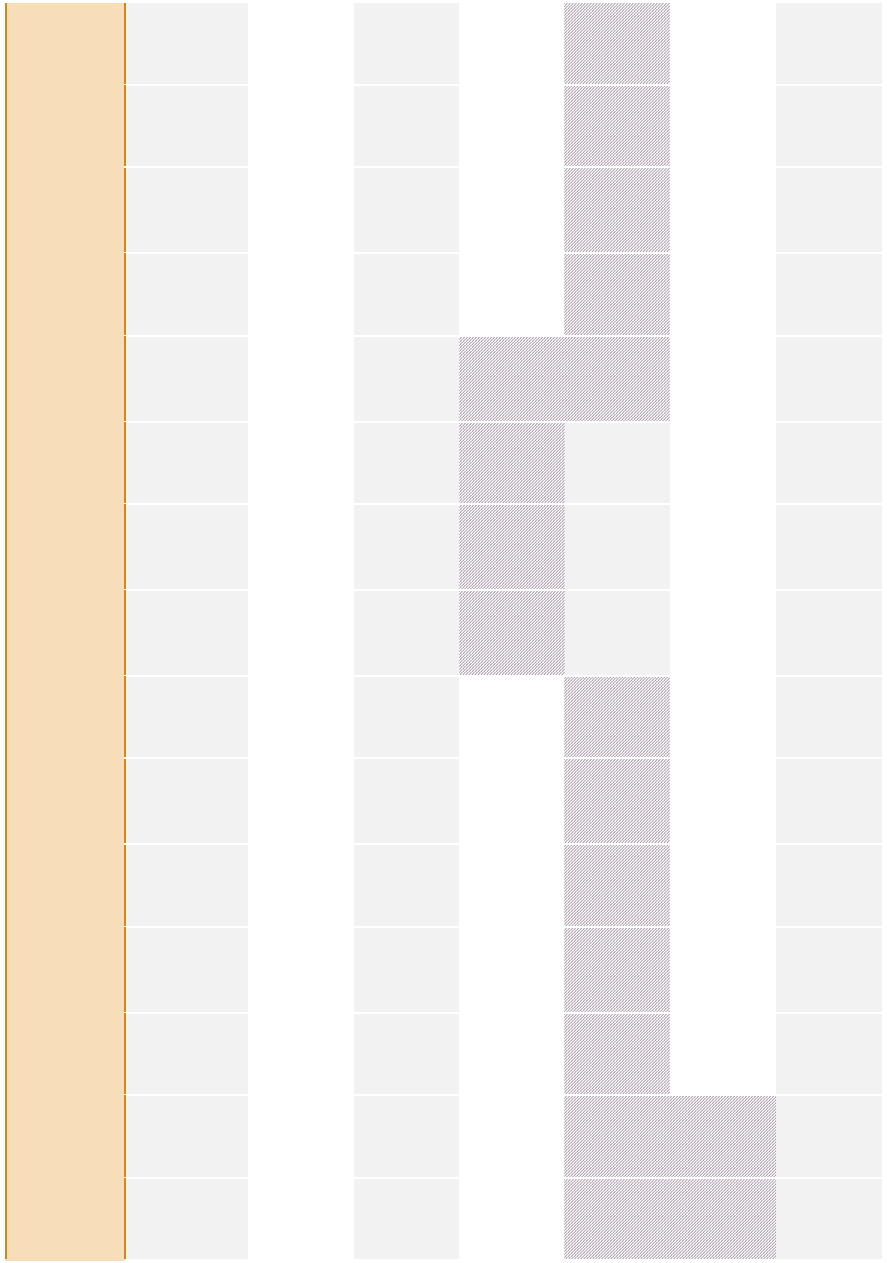
ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Planning	1	2	1		**25%**
Game mechanics	1	1	1	1	**100%**
UML diagram	2	1	2		**0%**
Class diagrams	2	1	2		**0%**
Variables	2	1	2		**0%**
Game Engine	3	2			**0%**
Trip set up	3	1			**0%**
Food manager	3	1			**0%**
Day manager	3	1			**0%**
Character manager	4	1			**0%**



Scenario manager	4	1	0%	
Testing	4	1	0%	
Choice Scenarios	5	2	0%	
Scenario 1	5	1	0%	
Scenario 2	5	1	0%	
Scenario 3	5	1	0%	
Scenario 4	6	1	0%	
Scenario 5 River	6	1	0%	
Testing	6	1	0%	
Non Choice Scenarios	5	2	0%	
Scenario 6	5	1	0%	
Scenario 7	5	1	0%	
Scenario 8	5	1	0%	
Scenario 9	6	1	0%	
Scenario 10	6	1	0%	



Scenario 11	6	1	0%	
Scenario 12	6	1	0%	
Scenario 13 Hunting	6	1	0%	
Testing	6	1	0%	
Graphic Interface	5	2	0%	
Start screen	5	1	0%	
Character selection	5	1	0%	
Food purchase	5	1	0%	
Checkpoints	6	1	0%	
Scenario graphics	6	1	0%	
Finish	6	1	0%	
You died screen	6	1	0%	
Testing	6	1	0%	
Testing	6	2	0%	
Testing	6	2	0%	



Monitoring and Reporting Mechanisms

Communication

- Discord server
- After-class meetings
- Discussion boards

Collaboration

- After-class meetings
- Google drive folder
- Discord server
- Trello board
- GitHub

Scheduling

- Excel project planner in google drive
- Trello board
- Discord communications

Appendix

CODE	ACTIVITY	DESCRIPTION	PLAN START	PLAN DURATION	DEPENDENCIES
	Planning		1	2	
P01	Game mechanics	Define and plan the rules and mechanics of the game, including the linear path gameplay, decision-making, and random events.	1	1	
P02	UML diagram	Create a Unified Modeling Language (UML) diagram to visually represent the structure and relationships of the game components.	2	1	
P03	Class diagrams	Design class diagrams to illustrate the classes, attributes, and relationships of the game objects.	2	1	
P04	Variables	Identify and define the variables necessary for tracking game progress, character attributes, scenarios, and resources.	2	1	
	Game Engine		3	2	
GE01	Trip set up	Implement the initialization of the game trip, including setting up the starting conditions, character selection, available resources, and checkpoint distance.	3	1	P01
GE02	Food manager	Develop the functionality to manage the food supplies, consumption rates based on character choices, and handling scenarios affecting food.	3	1	P01
GE03	Day manager	Create the logic for tracking the passage of time, updating the day counter, and managing the arrival at checkpoints.	3	1	P01
GE04	Character manager	Implement the character selection system, including their unique attributes, percentages of success, and effects on gameplay outcomes.	4	1	P01
GE05	Scenario manager	Build the system for selecting and presenting random choice and non-choice scenarios during the game.	4	1	P01

GE06	Testing	Conduct thorough testing of the game engine to ensure its functionality, stability, and proper integration of the different components.	4	1	GE01-05
	Choice Scenarios		5	2	
CS01	Scenario 1	Design and implement different choice scenarios	5	1	GE01-05
CS02	Scenario 2		5	1	GE01-05
CS03	Scenario 3		5	1	GE01-05
CS04	Scenario 4		6	1	GE01-05
CS05	Scenario 5 River		6	1	GE01-05
CS06	Testing	Test each choice scenario to ensure they function correctly, provide appropriate choices, and produce the intended results.	6	1	CS01-05
	Non Choice Scenarios		5	2	
NCS01	Scenario 6	Develop non-choice scenarios where events occur without player input	5	1	GE01-05
NCS02	Scenario 7		5	1	GE01-05
NCS03	Scenario 8		5	1	GE01-05
NCS04	Scenario 9		6	1	GE01-05
NCS05	Scenario 10		6	1	GE01-05
NCS06	Scenario 11		6	1	GE01-05
NCS07	Scenario 12		6	1	GE01-05
NCS08	Scenario 13 Hunting	Create a specific scenario for when the player chooses to hunt for food and not travel that day	6	1	GE01-05
NCS09	Testing	Perform testing on non-choice scenarios to verify their proper execution, event triggers, and their impact on the game progression.	6	1	NCS01-08
	Graphic Interface		5	2	

GI01	Start screen	Create a visually appealing screen for the game's start, providing options to begin the journey or exit the game.	5	1	GE01-05
GI02	Character selection	Design an interface to allow the player to choose their character from the available options.	5	1	GE01-05
GI03	Food purchase	Develop a user interface for buying food at checkpoints and deducting money from the player's funds	5	1	GE01-05
GI04	Checkpoints	Design an interface to show the player's progress, display the current checkpoint, and provide relevant information.	6	1	GE01-05
GI05	Scenario graphics	Implement graphical elements to enhance the presentation of choice and non-choice scenarios.	6	1	GE01-05
GI06	Finish	Design an interface to display the completion of the game and the player's overall outcome.	6	1	GE01-05
GI07	You died screen	Create a screen to inform the player of their death and provide options for restarting or exiting the game.	6	1	GE01-05
GI08	Testing	Test the graphic interface components to ensure proper functionality, visual consistency, and user-friendliness.	6	1	GI01-07
	Testing		6	2	
T01	Testing	Conduct comprehensive testing of the entire game, including the game engine, choice and non-choice scenarios, and graphical interface components. Identify and fix any bugs or issues discovered during testing.	6	2	GE06, CS06, NCS09, GI08