Name: Jon Kopf Mark:\_\_\_\_/50

# 1.Brief introduction \_\_/3

I will be working on the games NPCs and the quest system. The NPC includes a superclass that all non-player characters will inherit from, including enemies. I will be creating several different types of NPCs that serve different purposes. One type of NPC will give the player quests after interacting with the NPC and meeting certain requirements. Another type of NPC will allow the player to exchange gold for different items. The status of quests will be updated through triggers. This means that once an action associated with the next step in the quest occurs, a message is sent to the quest manager to update the quest.

# 2.Use case diagram with scenario \_\_/14]

##### **Use Case Diagrams**Diagram Description automatically generated

##### **Diagram Description automatically generated**

##### **Scenarios**

Scenario 1 (1st Use case Diagram)

**Name**: Interact with neutral NPC

**Summary**: The player interacts with a neutral NPC, which could have different functionality depending on the type of NPC.

**Actors**: The Player

**Preconditions**: This requires the player to be loaded in the game world, and not in an active battle. The player must also be near an NPC and press the interaction button.

**Basic sequence:**

**Step 1:** The type of NPC is determined by its class

**Step 2:** Dialogue is displayed for the interaction with the NPC

**Step 3:** Allow the player to interact with the dialogue

**Step 4:** Update a quest

**Exceptions:**

**Step 2:** Dialogue will be different for shop NPC and quest NPC

**Step 3**: A quest will not always be changed when interacting with dialogue, only if that dialogue interaction is a trigger for the quest.

**Post Conditions:** The player has stopped interacting with the NPC and is back into the game world, where they can move and interact with other objects.

**Priority:** 1

**ID:** IN1

**Scenario 2(2nd Use Case Diagram)**

**Name:** Update Quest

**Summary:** When a player takes an action in the game world it may trigger an update to a quest.

**Actors:** Player, Interact Dialogue

**Preconditions:** The player is not currently in a fight.

**Basic Sequence:**

**Step 1:** Add a new quest to the active quest list

**Step 2:** Lock an active quest

**Step 3:** Fail a locked or active quest

**Step 4:** Change quest data of an existing quest

**Step 5:** Set the current step of the quest

**Exceptions:**

**Step 1:** If the update is not to add a new quest, don’t add a new quest.

**Step 2:** Don’t lock a quest unless the update is to lock a quest

**Step 3:** A quest shouldn’t be failed unless that is what the update is for.

**Step 4:** Changing an existing quests data should not always happen on updating a quest.

**Post Conditions:** The Quest structure is different from before

**Priority:** 1

**ID:** UQ1

# 3.Data Flow diagram(s) from Level 0 to process description for your feature\_\_\_\_\_\_\_14

In the DFDs below, I will be depicting the Manage NPC process. I will also describe the “Determine Update Type” sub-process with psuedocode. This involves handling both quest and NPC interactions.

##### **Data Flow Diagrams**

Diagram, schematic

Description automatically generated

Diagram 0

# Diagram Description automatically generated

Diagram For Manage NPCs (2)

Diagram

Description automatically generatedDiagram

Description automatically generated

Diagram for Manage Quests (2.2)

Diagram for Interact with NPCs (2.1)

##### **Process Descriptions**

The process description for “Determine Update Type” is described in the following pseudocode.

If update\_data = 0 then

add a new quest

If update\_data = 1 then

lock quest

If update\_data = 2 then

fail quest

If update\_data = 3 then

change an existing quest

# 4.Acceptance Tests\_\_\_\_\_\_\_\_9

This feature has many different aspects to test, like where a quest is updated from, what kind of update is occurring, if an NPC is a shop or not, or whether the player can trigger a certain quest trigger. I think it is most important to test the reliability of updating quests. To do this, the test will check each function that can affect the status of the available quests. So adding quests, failing quests, or changing quest data will all be checked.

To add quests, I will do tests providing information that I expect to work in the system, but I will also provide information that should be rejected, just as a special character in the quest ID. When removing a quest, I will provide IDs that I know are valid, as well as IDs that I know cannot be valid. When updating a quest, I will go through each step of every quest using their associated trigger functions. These trigger functions are what is going to be called when a certain interactable is used by the character.

# 5.Timeline\_\_\_\_\_\_\_\_\_/10

##### **Work Items**

|  |  |  |
| --- | --- | --- |
| Task | Duration (Hours) | Predecessor Task(s) |
| 1.Implement Quest System | 6 | - |
| 2.Implement Simple dialogue System | 6 | - |
| 3.Map out main storyline | 6 | 1,2 |
| 4.Create element to display NPC text | 3 | 3 |
| 5.Add essential NPC’s to main story | 6 | 4 |
| 6.Map out different side quests | 9 | 4 |
| 7.Add Side quest NPC’s | 9 | 5,6 |

##### **Pert Diagram**

Diagram

Description automatically generated

##### **Gantt ChartTimeline Description automatically generated with medium confidence**