1 | Architecture and Traceability Report

Inherited Architecture

We made no significant architectural changes to the architecture create by Coffeeless for assessment 3. This Architecture is shown here:
https://drive.google.com/file/d/1eWgu62ZKsu3NizFCoDU2wmtsJlXwUa1H/view

Following a maxim of attempting to minimise changes we attempted to make as few changes to the inherited system as possible, ensuring that any changes we did make didn't cause any inherited tests to fail (as detailed in our testing report).

To minimise changes we focused on reuse of pre-existing code and extension of pre-existing interfaces. We also wanted to minimise changes not only to preserve Architectural integrity of the code but also to ensure that previously met requirements are still met. This means that previous team's traceability of requirements is still valid.

Changes to Architecture

During Assessment 4 we made two changes to the architecture we received from Coffeeless, both of which are directly tied to the new requirements which were added, namely the new "Gorilla" character and the designable "Quests". These new Requirements are broken down in the table below:

Requirement ID	Description	Criterion	Assumptions, Risks or Alternatives
F17	Game must contain a Gorilla character type	The Gorilla has its own unique ability, but will randomly turn evil for 1-round and attack teammates.	We assume the Gorilla's abilities are good enough to risk having Gorillas on a team. A risk is that they are too overpowered, or mostly unused and unwanted by players because of the risk of turning evil.
F18	The game must have a quest design menu	The quest choice must happen before the game begins, and must involve at least 3 objectives.	A risk of this would be that the player finds the quests too easy or is able to complete them all meaning the game is less enjoyable. We assume the quests will have reasonable rewards so that there is motivation to do them.

F19	Game Quests must provide in-game rewards for the team	The quests must provide the user with in game currency or specific items upon completion.	We assume these rewards will be valuable to player to make the quests worth completing. A risk would the quests becoming obsolete because of inferior rewards.

The architecture changes associated with the new “Gorilla” requirement where very small, totalling to 4 changes to pre-existing files, one addition to the CombatAbility interface and one new file, shown in the table below:

File Name:	Change details:	Reason For Change:	Change ID	Associated requirement ID
ButtonManager.cs	Added “state.availibleCharacters.Add(CombatCharacterFactory.CombatCharacterPresets.Gorrila);	This gives the player access to the Gorilla character from the start of the game by adding it to their available characters.	4.01	F17
CombatCharacterFactory.cs	Added a combat character Preset for the Gorilla and its associated details i.e name string, Heal, Energy, Abilities, basic attack stats and, Sprites.	This was done to make the Gorilla usable in combat. Fulfilling the requirement of the Gorilla as a Playable character.	4.02	F17
CombatManager.cs	Added a randomization of selectable target group to “targetselectionstage” based on the new “isGorrila” bool	Done to fulfill the requirement of the Gorilla going “evil” and attacking the players.	4.03	F17
CombatAbility.cs, SimpleAttack.cs, SimpleHeal.c	Added new Bool “isGorrila”	This was done so the CombatManager would know when the Gorilla was attacking and therefore give it appropriate targets	4.04.01-4.04.04	F17

s, GiveEffect.cs				
GorrilaAttack.cs	New File, realization of CombatAbility.cs interface.	The GorrilaAttack file was created to add a new type of ability. This was done to have a CombatAbility with “isGorrila” set to true	4.05	F17

The above changes were done to implement the requirements (F17) for the Gorilla character. With changes 4.01 and 4.02 being the Adding of the character and changes 4.03 - 4.05 being changes to allow the Gorilla character to attack friendly characters and “Go Evil”

The Architecture changes associated with the Quest (F18) were similarly small, following the programming maxim of trying to make changes as small as possible. To accomplish this we built the quests on top of the pre-existing system of GameState Variables. A small amount of code was added to create the appropriate buttons using the Button manager Script. The actual quests themselves are started by loading specially created save files when the player selects to start a quest, following our maxim of minimised changes.