Project Dreamer / Equipment Module

Architecture/Design Document

Change History

Version: 0.1
Modifier: Joshua Griffis
Date: 13/04/2022
Description of Change: Design Document started

Introduction

This document describes the architecture and design for the Project Dreamer application being developed by Radical Dreamers. Project Dreamer is a Third Person Role Playing Game where you explore the dream world and interact with its inhabitants and fight bad dreams.

The purpose of this document is to describe the architecture and design of the Project Dreamer application in a way that addresses the interests and concerns of all major stakeholders. For this application the major stakeholders are:

- Developers – they want an architecture that will minimise complexity and development effort.
- Project Manager – the project manager is responsible for assigning tasks and coordinating development work. He or she wants an architecture that divides the system into components of roughly equal size and complexity that can be developed simultaneously with minimal dependencies. For this to happen, the modules need well-defined interfaces. Also, because most individuals specialise in a particular skill or technology, modules should be designed around specific expertise. For example, all UI logic might be encapsulated in one module. Another might have all game logic.
- Maintenance Programmers – they want assurance that the system will be easy to evolve and maintain into the future.

Design Goals

The goals we decided on for the design were:

- When we interact with a weapon or accessory slot it will show a list of possible items to equip instead.
- The equipment will affect the stats of the character and will either add or subtract the equipment stats when equipped or unequipped.

System Behaviour

The use case view is used to both drive the design phase and validate the output of the design phase. The architecture description presented here starts with a review of the expected system behaviour in order to set the stage for the architecture description that follows. For a more detailed account of software requirements, see the requirements document (GDD).

The equipment system will be used in a few ways, the biggest impact being to character stats. When you equip a piece of equipment you get the stats of the equipment. In future builds we plan to incorporate the ability for equipment to imbue the character with more skills or give condition or elemental resistances. The equipment system can be used by going to the equipment menu in the menu UI where the menu will show the character and their three equipment slots. When a slot is interacted with, a list of items from your inventory will show up

with only the items that are accessories or weapons depending on what slot you are equipping to. When another piece of equipment is picked it swaps the equipment pieces.

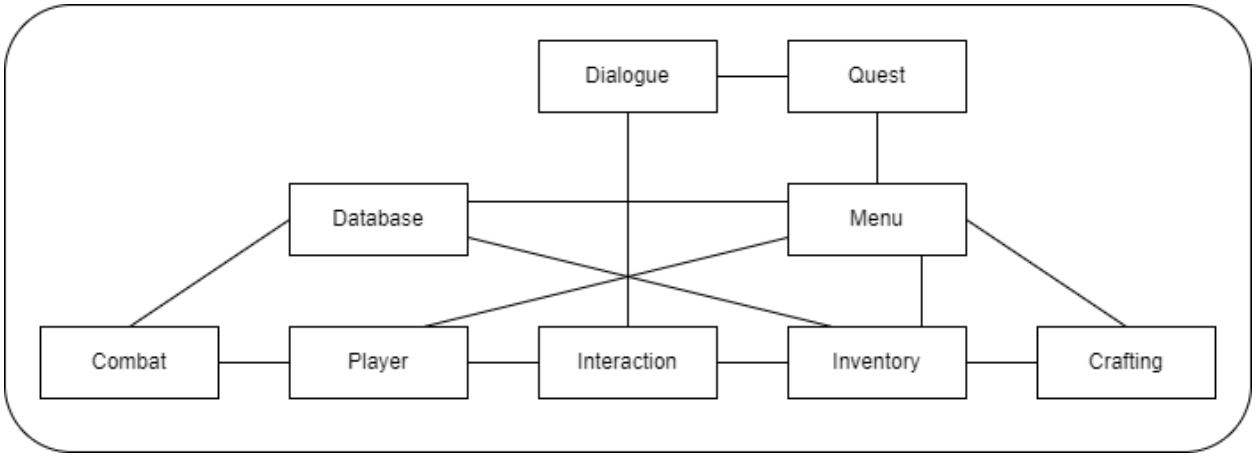
Logical View

The logical view describes the main functional components of the system. This includes modules, the static relationships between modules, and their dynamic patterns of interaction.

In this section the modules of the system are first expressed in terms of high level components (architecture) and progressively refined into more detailed components and eventually classes with specific attributes and operations

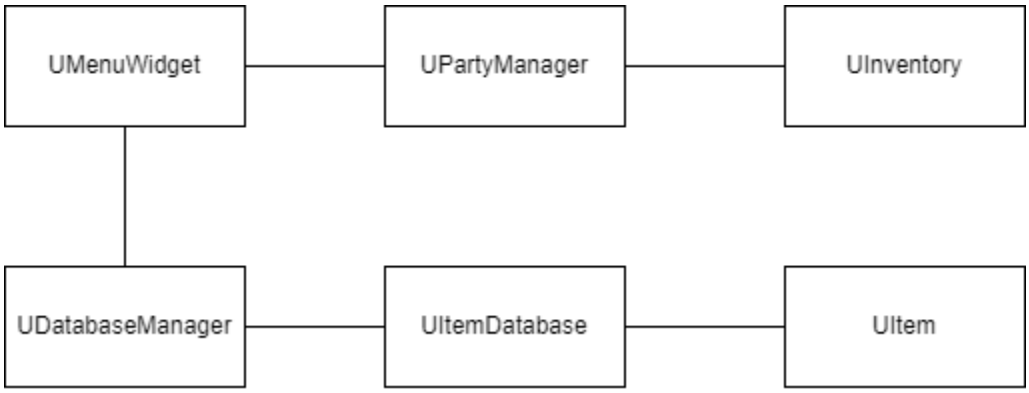
High-Level Design (Architecture of the Entire system)

The high-level view or architecture consists of 9 major components:



- **Player:** is the main control over the character in the world and allows the user to interact with the world.
- **Database:** stores the data needed for many features to work.
- **Interaction:** handles the objects the player can interact with like talking to characters, opening chests, opening doors, etc.
- **Dialogue:** is responsible for handling the flow of conversations and displaying dialogue to the user.
- **Inventory:** manages the items picked up by the player and money stored. Allowing the player to use them at a later point.
- **Menu:** allows interaction with some modules and features.
- **Combat:** allows interaction between enemies and player characters as they fight. Controls the flow of battle and which battler may act.
- **Quest:** allows the user to have stored data for quests and be able to receive rewards upon completion.
- **Crafting:** allows the player to combine multiple items together to create new items

Mid-Level Design of Equipment Module

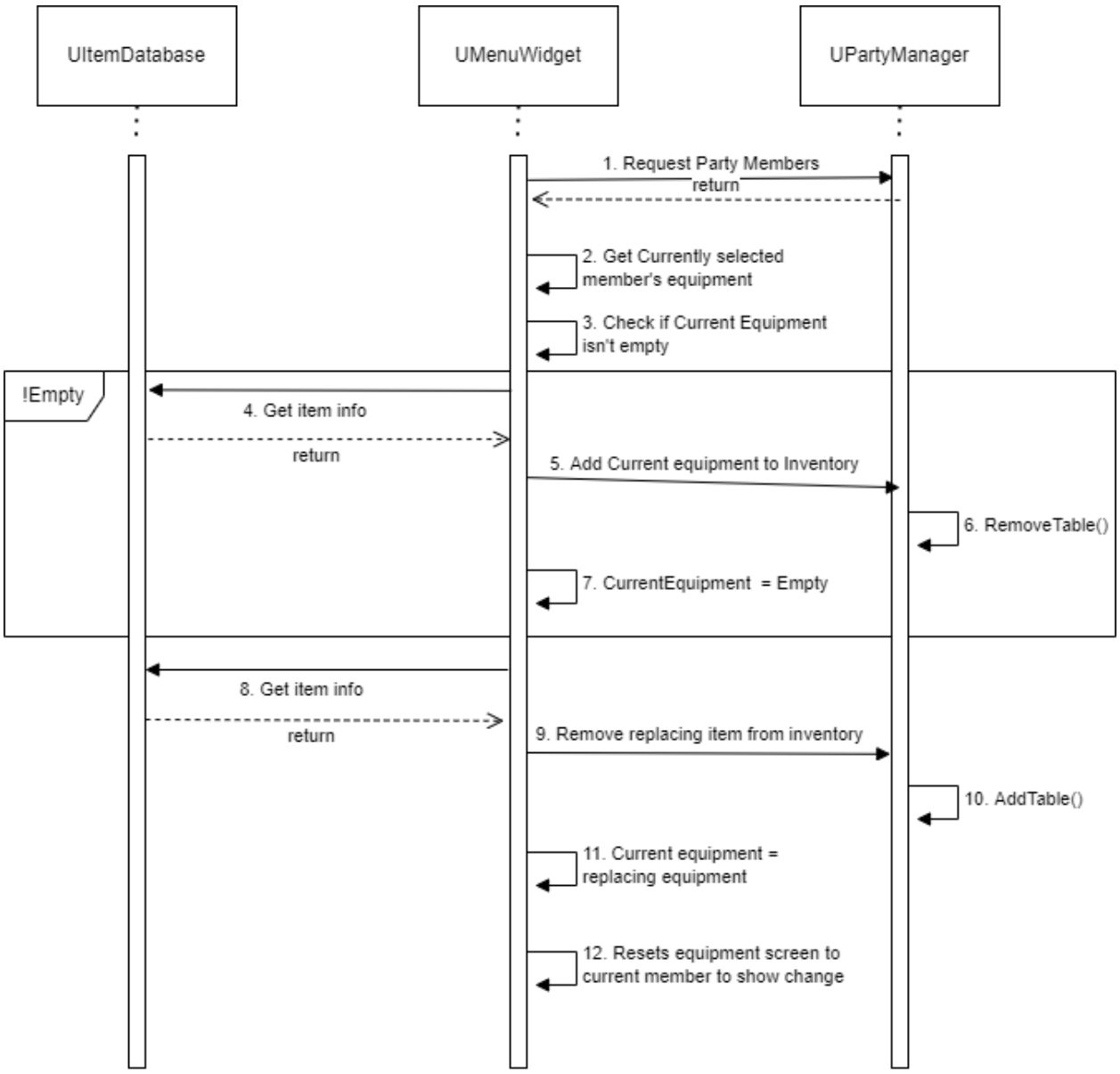


The Equipment Module interacts mainly with the two Managers(Party and Database) from the menu UI to change character equipment and stats.

Detailed Class Design of the Equipment Module

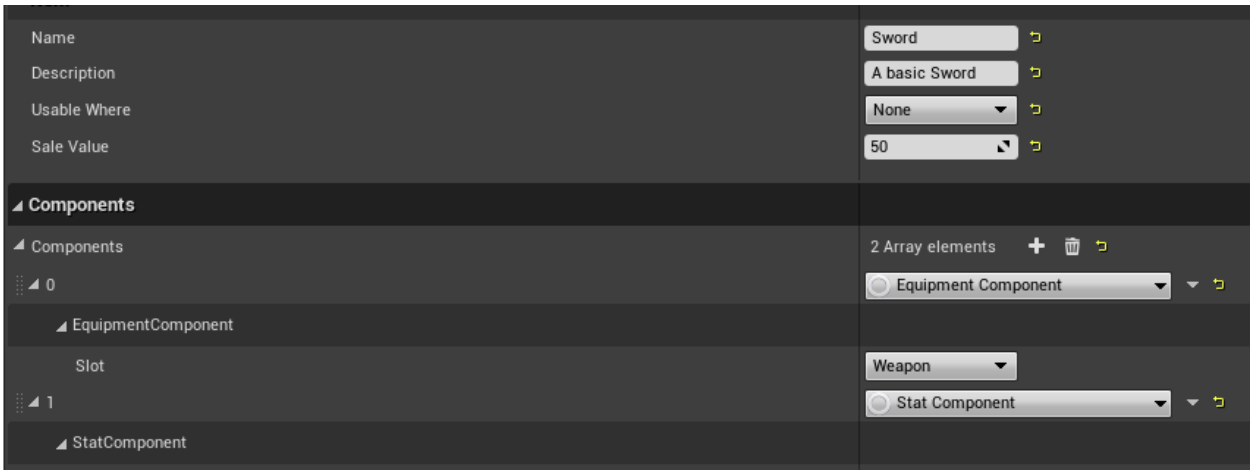
See UMenuWidgetRelationsUML.png

Process View of the Equipment Module



Above is the process of changing a piece of equipment to a different one. If there is a piece of equipment equipped in that slot, it will place it in the inventory. And then it will remove the replacing item from the inventory and equip it in the slot.

Use Case View



Above is a Sword, a weapon in the game. The sword has an EquipmentComponent and a StatComponent.

When in the equipment menu, it will check if items have the EquipmentComponent and which type of slot it fits in. and when equipping it checks if the equipment has a StatComponent and removes the stats of the old one and adds stats for the new one to the current character.