Project Bones -CMAdams

Background:

Project Bones is a proof of concept of an adventure puzzle game developed by "Undefined Studios". The proof of concept itself was created using the Bevy game engine, an experimental new game engine developed in the Rust programming language.

Structural Description:

Bevy's core design follows the ECS, or entity, component, system, design pattern which has been gaining popularity in games programming as a successful alternative and solution to the problems of traditional object oriented programming patterns. In ECS, the OOP concept of an array of structs (or in reality a gameworld of Classes) is replaced by the concept of a struct of arrays. Each entity can be thought of as an index associated with each array within our struct of arrays, and each component is merely a specific type for an array within our struct of arrays. Systems themselves are very straightforward; they are simply functions that operate upon the world by managing entities and components, or even triggering new systems. ECS has also been described as data-oriented design due to this flat structure when compared with OOP. When a value is queried within an ECS system, only the component itself is operated on, not the entire entity. This fundamental design change allows for gains in performance while simplifying some of the complex class hierarchies associated with game logic. Furthermore, it is also a wonderful fit in the Rust programming language as it is entirely friendly to the borrow checker; Bevy in particular has a wonderful ECS system allowing systems to be written entirely as normal functions in Rust.

The actual architecture of "Project Bones" is relatively straightforward focusing on a couple of key systems. Within the game world, players interact with a variety of different entities; the backend first determines what the player can interact with before processing any interaction targeted towards a given object. Depending on the type of object being interacted with, the backend processes the request accordingly. The dialogue system and inventory system are extensions of the interaction system, allowing players to further their interactions by picking up items or having conversations.

Movement itself is rather straightforward; the player may move in any direction unless they collide with anything. The project also integrates a JSON parser which handles settings for the game as well as hotkeys for the player. The most important feature is the integration of the Level Designer's Tool Kit or LDTK. LDTK allows our designers to work in an intuitive GUI environment to create the levels. Basic game logic can be stored in LDTK as well; this helps to streamline our creative process and make further levels and environments with ease.