Task #518 - Task Development Trials new

People

Assignee	**Accountable**

Estimates and progress

Work		**Remaining work**	
% Complete		**Spent time**	0h

Details

Priority	Normal	**Date**	-
Category		**Version**	

Costs

Spent units		**Labor costs**	
Unit costs		**Overall costs**	
Budget			

Description

Description

What we would like you to create is a "first-pass Traversal System" for the Survival FPS prototype. The goal is to support more expressive movement (beyond simple walk/sprint/jump) in a way that is cleanly implemented in C++, easy to extend, and friendly to designers. The system should live in a "new Unreal project" (UE5, C++ first). Please focus on code quality, separation of responsibilities, and how easily other programmers and designers could build on top of what you make.

The traversal system should be built as a combination of a "custom movement/character logic in C++" plus small Blueprint hooks for tuning and animation events. Below is the scope and breakdown for the trial.

Core Requirements

Implement a "C++ FPS character" that supports:

- Walking, sprinting, crouching, sliding, and at least one "advanced" traversal action such as vaulting or mantling over low obstacles.
- All movement state, speeds, and stamina logic should be implemented in C++ (no Blueprint logic for core rules), with key variables exposed to designers via UPROPERTY.

Use the "Enhanced Input system" for bindings and keep a clean separation between input code and movement logic (e.g., input in the character, traversal logic in a dedicated component, or custom movement mode).

———

Traversal Features

The system should support a “variation of movements” appropriate for a Survival/FPS:

Baseline traversal:
- Walk, Run (Sprint with stamina drain/regeneration and configurable speeds).
- Crouch/slide with appropriate collision and speed transitions.

Contextual traversal (choose at least one of these and implement it fully)
- “Vault/mantle” over low/medium obstacles using traces to detect ledges and decide between a short vault vs full climb.
- “Ledge grab” where the player can catch a ledge when jumping toward a wall and then pull up or drop.

This logic should be encapsulated (e.g., as a custom movement mode or a dedicated C++ component) to avoid filling the character class with special-case code.

———
———

System & Code Design Expectations

Architecture
- Use a “custom movement component” or a clearly separated C++ traversal component that plugs into the character.
- Define a small internal “state machine” for movement modes (Idle, Walk, Sprint, Crouch, Slide, Vault, etc.) with clear transitions and rules.
- Expose read-only getters for other systems (e.g., `IsVaulting()`, `GetStaminaNormalized()`) and avoid tight coupling to specific animations or camera logic.

Detection & environment interaction:
- Use traces to detect obstacles/ledges and decide which traversal action is valid (e.g., vault if obstacle is within a low height range, mantle if higher).
- Ensure the system behaves reasonably with different obstacle sizes and slopes (you can demonstrate this with a simple test level).

Blueprint & designer hooks:
- Expose configuration data (movement speeds, stamina costs, max vault height, etc.) via UPROPERTY so they are easily tweakable.

———
———

Deliverables (by the end of the trial period)

- A character that can walk.
- A character that can sprint (stamina drain and visible stamina bar)
- A character that can crouch under obstacles.
- A character that can do one ‘advanced’ traversal mechanic(vaulting/mantling, ledge grab and climb)

The project should be hosted somewhere where it can be downloaded, and the code can be viewed (i.e. gitLab, github).

The candidate should provide either a video demonstrating the mechanics working or be willing to stream and demonstrate them on Discord.