States:

After menu, game starts in idle mode, where it waits for player input. From here it can transition to pause menu mode, action mode, and event mode.

Idle Mode: Game waits for player input. All units and AI are paused, and do not move. Changes into action mode when the player walks. Player can initiate and cancel actions, but such actions will not automatically play out. There will be a separate button for actively passing time, which is used to either wait if no actions are queued up, or to execute queued actions.

Action Mode: Game advances time. All queued actions will play out first, and then AI controlled characters will follow their AI

AI:

AI characters will have “types” with a priority list. Their type defines the emphasis of that character’s AI, and their priority list is a list of actions and conditions they must check to fulfill their type. Most types will have survival at top priority, and will first check if there are any threats that severely threaten their hp. Afterwards they will seek to fulfill their role in battle. Healers will seek out targets with the lowest hp to heal. Dps will seek out weak targets to destroy. Debuffers will attempt to neutralize dangerous targets, and Tanks will attempt to cover targets with lower survivability or higher priority in a given context. Buffers will go through a decision tree of conditions based on the enemy composition, number, and characteristics to decide which other role needs buffs first to do their job, and will have an AI that analyzes battles on a higher level, in terms of “offense”, “defense”, “flanking”, etc., since their buffs place emphasis on a certain role in a battle, and thus on a certain strategy.

AIs will decide whether or not a priority for their “role” is fulfilled if specific parameters are met. This will involve checking all relevant members of a battle, as well as possible characters not involved in battle.

Game Loop:

-Start at floor beginning

-Travel through floor

-Fight monsters

-Reach exit and go to next floor

Pseudocode:

StartGame:

-Generate map

-Spawn character

-Spawn enemies

MapGen:

IdleState:

-Listen for player input for movement or menu

-Listen for pause menu

-Be able to switch to ActionState

ActionState:

-Be able to switch to IdleState

-Be able to pause

Character Controller:

-Listen to movement input

-Store a command for an attack

-Start an attack when action phase starts

Outside of combat:

-Monsters have preset path marked out by destination transforms, and move in a set patrol path between them. When the player isn’t moving, the enemies don’t move.

Inside of combat:

-Moving causes the game to change to action mode where everything can move

-Whenever something initiates an action such as attacking or casting a spell, there is a delay before damage is applied in an area. On the frame that that delay ends then the character either applies their effect or casts a shape on the area and applies their effect on whatever was hit by that cast. All characters will thus require a charge-up animation.

-When the player performs an action, the game doesn’t pause until the action is over, or if the action is canceled by the player or by being interrupted.

Skill System:

-change target stats (by flat amounts and percentage)

-change target position

-change user stats (by flat amounts and percentage)

-change user position

-apply status effects with a timer

-be able to lay obstacles

-be able to create areas of effect that persist for a certain time and apply their effects continuously

-be able to target multiple people, preferably by either reusing single target code in a multi-target function or have a system that works with either one or many targets, and can specify a limit for max number of targets

-Skills should be able to call other skills to allow combinations of a few basic skill types to be combined into more complex skills

-extensibility for new effects

Item System:

-Be able to do virtually everything the skill system would do

-Have items be either consumable or not

-Items would be a collection of item information, and using them is done by an external function by the character, which reads the items’ details and adds them to the list.

Stuff that needs putting somewhere:

-item system

-character action queue

-skill system