# Android Game Using an Entity Component System

## Project Plan

## Aims and Objectives

### Aims

Create a finished skeleton of an android game using both LibGdx and Artemis-odb frameworks. The reasons I say ‘skeleton’ is because during this project I would first focus on the mechanical and backend aspects of gave development as opposed to the frontend. As my art skills would not demonstrate any technical achievement on my part.

Correctly utilise an Entity Component System when creating this game. In short, focus on creating most characters and interactions on screen by breaking down entities into a different set of components. The goal with this is to easily add and remove different functionality from different entities within the game.

Use intuitive design to make the game easy to be used by those who use phones. An issue I’ve had before is that I created controls there were too complex for the platform I created a game for. I do not wish for this project to suffer a similar fate.

### Objectives

Decide on the core gameplay loop prior to the finished prototypes. This is a very important part of the project as the core gameplay is what makes people want to continue playing your game. In my first game, the core gameplay was swiping arrows based on their colours. In my next game, the core gameplay wasn’t decided upon until later in the project, to its detriment. This isn’t something I can already have pre-planned since it requires other players to try out different controls schemes and gameplay loops to see which they prefer.

Although a basic objective, finish with an actual working functional product. The reason why this is an objective is because games are incredibly easy to over scope. On top of that bugs are common as testing, at least the front end, is a bit more manual in games currently.

Create solid tests for parts of the game that cannot be tested manually. An example of this would be in a project where I created a randomized map. I created a test that generated 20000 maps of varying sizes and checked their attributes to ensure everything was generated correctly. I need to make sure the parts of this is the game I’m going to create are also thoroughly tested

Implement a randomized game element. This is an incredibly broad statement, as this is the original plan. But to boil it down a bit further, there are two types of games that I’m interested in, investigating can be replayed a lot times. Ones with thousands of levels and ones where the game is different every time. In both situations, there is an algorithm behind it’s creation. As an example, you can create a sudoku generator that can generator thousands of puzzles and based on different variables the puzzles can be easier or harder to solve. I want to create a system where from a small set of variables you can create thousands of combinations for a player to explore. In the coming weeks, I may come back and refine this statement.

### Challenges and Originality

The challenges of this project are the use of an Entity Component System. The tools I have chosen to use have good documentation, but are not incredibly common in comparison to the standard way of creating games. The resources online for game development are already comparatively niche when compared to web design. But this system is a niche of a niche so it’s certainly going to be challenging.

Another major challenge is scope creep, as mentioned earlier, when developing a game, it’s easy to want things that are well beyond your means or underestimating how long it takes to figure out something you thought was simple.

Performance is something that can often be forgotten. I’d like the game to be able to be run on a range of different phones with a smooth framerate. In game development managing your resources is very important as well as when to re-use and dispose of assets.

The originality in this project are the tools I am using. Many people are creating a game or app, but I’m not sure if any are trying to use this relatively new way of creating a game. The initial setup makes starting the game a bit slower but the payoff is when the project gets incredibly large. I know the Unity Engine takes aspects of the Entity Component System, but I’ve opted not to use it as using two frameworks, means you need to program more of the game interactions yourself, instead of using plugins

## Requirements

### Functional Requirements

#### Basics

* The Player can select different modes of game to play
* The Player can view their statistics
* The Player can pause the game
* The Player can exit the game

#### Dancer

* A player can control a single dancer
* The player interacts with enemies by either tapping, swiping or dragging them.
* If the player’s dancer dies (loses fans) the game is over
* A dancer animation responds based on player input

#### Map

* The map will display different rooms to players
* The map will display special rooms in different colours to players
* The map will highlight the current room the player is in

### Non-Functional Requirements

* The game utilizes both LibGDX and Artemis ODB.
* The game utilizes both sound and music
* The game utilizes a range of touch controls, (swiping, tapping, dragging)
* The game can be run on minimum Android SDK version of 15 (changeable)
* The game can save and load user data
* Mobile processors should be able to render the game without any significant drops to frames
* The game controls should be intuitive to users, easy to pick up and use