## Game Concept Developer

## Summary

I decided to build a runner around the **Influencer** theme.

I could not think of something around *fashion* that would not require a lot of art, and also I discarded *body transformation* for the same reason.  
So why *Influencer*. Being an *Influencer* and getting followers, subscriptions and likes, it revolves a lot around *fashion* too and everyone can relate to it, so it would be *marketable*.  
  
I used crowd growth for the stacking/collecting mechanic.

**Core Gameplay**

Player moves forward collecting followers.  
Must avoid obstacles and take into consideration the line of followers.

Movement is with swype control.

At the end of the level, player get coins based on how many followers he collected.

**Thinks that could have been there**

There are a lot of things I would like to add to make it more juicy, but time is tight and I didn’t want to risk it.  
  
- Gameplay  
Instead of having only obstacles and followers, I tought about adding the popular opposite narratives gates. So if you chose a good thing, you get more followers (or multiply them), if you choose something bad, you lose them.  
This could have made the gameplay a little better.  
  
- End of level.

At the end of the level it would be a vertical bar with a lot of multipliers (x1, x2, x3, etc). When player get there, all the followers start showing like or heart emojis and the multiplier goes up based on the number of followers. It would be something like “With this many followers, you will get this many likes”.

- Game feel

Because this is about influencer and getting followers and likes. I wanted to add emojis to the followers. So every time you get one, it shows heart or happy emoji, and when you lose one and it falls, it shows a sad or funny emoji.

This will make it look way more fun and more relatable to instagram or tiktok.

## Technical

I built the whole project around **ScriptableObjects**.

It’s important for the project to be able to be maintained and extended for the rest of the team, and **ScriptableObjects** are perfect for this.

Anyone can see and update the properties of all objects or create new ones.

They could create different obstacles or persons or whole levels, without the need of touching any code.

Also, with SOs, the data could be stored on the cloud in a json or xml file, and restored into the SO directly when loading the game. So we could tweak values from our server without the needing to update the app itself.

## Reusability

**ScriptableObjects**

All SOs are very easy to change or use just as they are, so they could be used in any project.

**Common Scripts or Managers**Basic Enums for game state or configurations.

UI Manager.

Game Manager.

Level Manager.

Feedback Manager.

Player Preferences.  
Level Manager.  
All these are pretty generic so they can be used in any game.

**Scene Structure, Canvas and UI**

You can have the same scene with all the managers and a basic UI. And that could serve as the base for almost any HyperCasual game.  
When I create a game for myself, I always use the same scene as a base, so I don’t have to create a new UI or any setup.   
Then you make some basic tweaks for the level end UI or something specific you need and you’re ready to go.

**Player Movement**

This could have been more modular. But player movement code can be used in any runner.

Because the time was tight, I did not write modular components for everything and I used more singletons than I should.

Finally:

I know you’re evaluating not only code, but also creativity and game feel, and I feel that I failed in that part.  
I panicked a little when reading the theme about fashion or body improvement, I went blank and started thinking about all the art I don’t have and ideas that won’t work.

So I ended up doing something that I thought it could work, but I didn’t have the time to add more particles and animations to make it feel better, and also I didn’t make the end of level minigame. And a nice end of level makes a huge difference.  
  
Because of this, I tried to focus in architecture and maintainable code.

It felt kind of weird to be honest, but well, you can see in my portfolio that I can put together some nice art and some gamefeel. So with a nice artist in the time that would have been amazing!

Also, just so you know, one of the reasons I want to work with you, is so I can get better and become a better dev. Working in a team I can learn to write better code, but also get better at quick ideation and game design.

So, if you get me the chance, I promise you won’t regret it. I love to learn and I love making games, so in just a few months I will improve a lot!  
  
Thanks for this opportunity!