# Pipeline for my final year project

Blender > Game engine, be that Unity, Unreal, or Godot.   
  
Blender projects are stored in a .blend file. This type of file stores every bit of information about the mesh, rig, and animations. Any tool you use in blender will be stored here.

Game engines only accept assets like these in a .fbx file. These files don’t hold as much data as the .blend files, as Game Engines prioritise performance over Artistic Freedom

Generally, a Game Engine will only accept the following thing:

* Meshes
* Deformation bones
* Vertex Weights
* ShapeKeys (depends on engine, may only be specific ones.)

(Everything else gets lost in the conversion from .blend to .fbx)

This poses limitations, as sometimes the rigs, animations and models will break upon conversion because they relied on tools that .fbx doesn’t support. This may mean that an animation will be incomplete, a rig will get changed to only use basic bones, messing up the mesh, and the model won’t update the same way when moved manually in an engine.

Animations have to be baked in and exported separately, and any models and rigs have to be carefully made so that the shape of the mesh doesn’t rely on unsupported tools. Rig need to be “Game-Ready”.

Unity and Godot have their own export scripts for .blend files, so I can simply keep a .blend file in my asset folder, and they will read it just fine.

## Game Engine > Blender

Sometimes, some adjustments to animations will be necessary for an object that is already in the game engine. If we’re using Blender as our main modelling and rigging tool, it might be a better idea to use the .blender file that then got converted into .fbx. But, if that isn’t an option, or we want to see a specific situation we set up, we can open the .fbx file in Blender

The bones of the rig will have to be retargeted, as .fbx uses a slightly different format for its coordinate data. They might be in the correct place but facing a wildly different direction by default.

# Existing tools and solutions

These problems have been known for a longer time, and some people have developed add-ons for the open source engines and Blender.

**Game Rig Tools**  
Extension for blender that generates a Deform Rig and Control Rig for you to work in separately. A Control Rig is one that uses Blenders many tools, while a Deform Rig copies the position of the Control, and is Game-Ready.   
Game Rig Tools also simplifies the baking of animations, which would be the most time-consuming process while using this kind of setup

**Rigify**  
Extension for Blender, Has many different features, like improved pre-built rigs for basic common models, like quadruped, humanoid, shark, horse, etc. These prebuilt rigs are made of sub-rigs, like Arms, Legs, Head, and so on, which you can add on to your own rigs.  
Rigify can also be used to retarget .fbx rigs automatically.

**Send To Unreal**  
addon for blender, fully automates exporting .fbx files for Unreal Engine 5, in a similar manner to Unity and Godot.

# Potential Automation

I can make my own Blender add-on that combines the features of addons listed above. That would create a tool that can use .blend files in any game engine   
 -> would that be overkill? Potential for deployment online on its own, have to learn how to make an add-on

I can write my own scripts to automate certain tasks, like retargeting or   
 -> not consolidated for re-use, but easy to make.

I can make a Blender add-on and only adress issues I come up with in my project

-> focuses solely on what my project needs, + have to learn how to make an add-on