Finalized Plan of Work

**Expected**:

* Early Plans:
  + Get the AnisoMPM code to work on WIndows
  + Get a simple version of the add-on working
    - Just for us to use. Not necessarily something that would be useful yet, but would just be something we could build off of.
  + Run the AnisoMPM code on some of our own objects to get a better feel for how it works
* Mid Plans:
  + Get everything to run completely within Blender
  + Get the add-on to run very simple simulations. Perhaps just being able to tear a single piece of cheese.
  + Get a working UI in Blender to run the simulations
    - This doesn’t need to be super fancy yet. Just something simple we know how to use.
* Late Plans:
  + Add more customizability to the simulations that you can run within Blender
    - Material Type
    - How it is damaged (Torn, cut, twisted, etc.)
    - Framerate
    - Simulation time
    - Etc.
  + Get Blender to send all of these new customizations to the AnisoMPM code.
  + Make the UI look as pretty as possible
  + Lots of testing, making sure that it is easy to use for everyone, not just us

**Actual:**

* Early Plans:
  + Get the AnisoMPM code to work on WIndows
  + Get a simple version of the add-on working
    - Just for us to use. Not necessarily something that would be useful yet, but would just be something we could build off of.
  + Run the AnisoMPM code on some of our own objects to get a better feel for how it works
* Early plans didn’t change. Since this was the very beginning, we had no reason to change our plans yet

* Mid Plans did change. We faced several roadblocks getting the AnisoMPM code to run in Windows, so we had to adjust for that. However, we did have a basic Blender add-on made.
* Mid Plans:
  + Improve the UI of the Blender add-on
  + Use TetWild and PyMesh to convert between files in Blender and the AnisoMPM code.
  + Continue trying to get AnisoMPM to work within Windows
  + Create Python scripts to generate headers for AnisoMPM
    - AnisoMPM uses header files to specify everything about the simulation.This includes things such as:
      * Framerate
      * Gravity
      * .mesh file location
      * Information about the physics of the object
      * Information about the damage to be inflicted
      * Information about other objects
      * Etc.
* Late plans had to change quite a bit. We did finally get AnisoMPM to run in Windows, but found that it used .bgeo files as output, which is owned by Houdini. Houdini is another premium 3D rendering software, which we did not want to use because our goal is to make something open-source. This also invalidated the work done with TetWild and PyMesh to work with .mesh files and convert them. We had to restart the conversion process from scratch.
* Late Plans:
  + Find a way to convert the .bgeo output to .obj without leaving Blender
    - Find a python package online to convert .bgeo to .obj so we can use it in Blender
    - Find another add-on for Blender that allows us to use some Houdini functionality. This would allow us to use gconvert, a function within houdini to convert .bgeo files into .obj files
    - Find the pattern in .bgeo files ourselves so we can convert them into .obj directly. Thankfully, we were able to use gconvert within Houdini to convert a few files so we could check our results
  + Add functionality to call conversion/animation scripts within our Blender add-on
    - Given this, add more material options pulled from other demos
  + Add more functionality to the Header Generation scripts. So that the end users would be able to specify more about the simulation.
  + Add as much to the UI as possible. Allow the user to easily insert whatever information they need to.