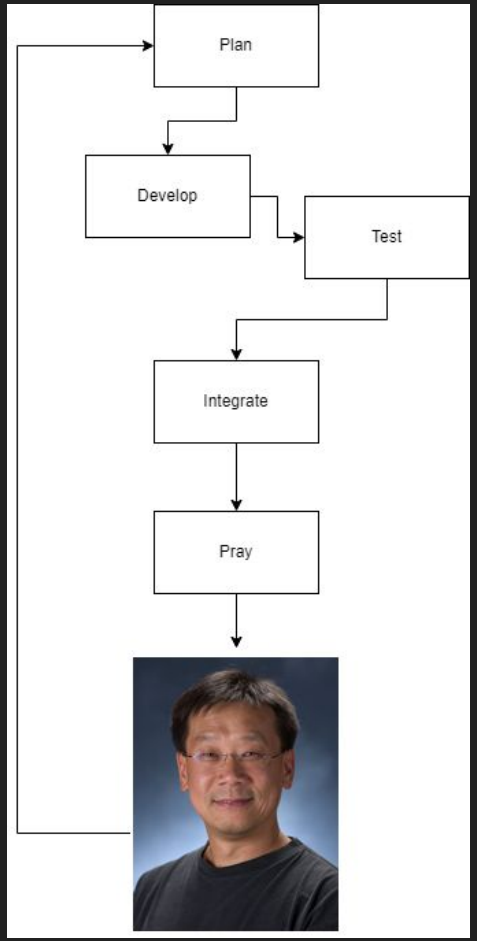
Retro Presentation

Team Origin

Shakeel Khan, Jeongwook Oh, Jino Chai, Gary Yuen

Process

- Start the week off identifying what tasks need to be done.
- Divy up the tasks.
- Communicate via Discord to post updates/ask for help.
- Submit a PR on GitHub when something is ready.
    - Integration.
- Pray.

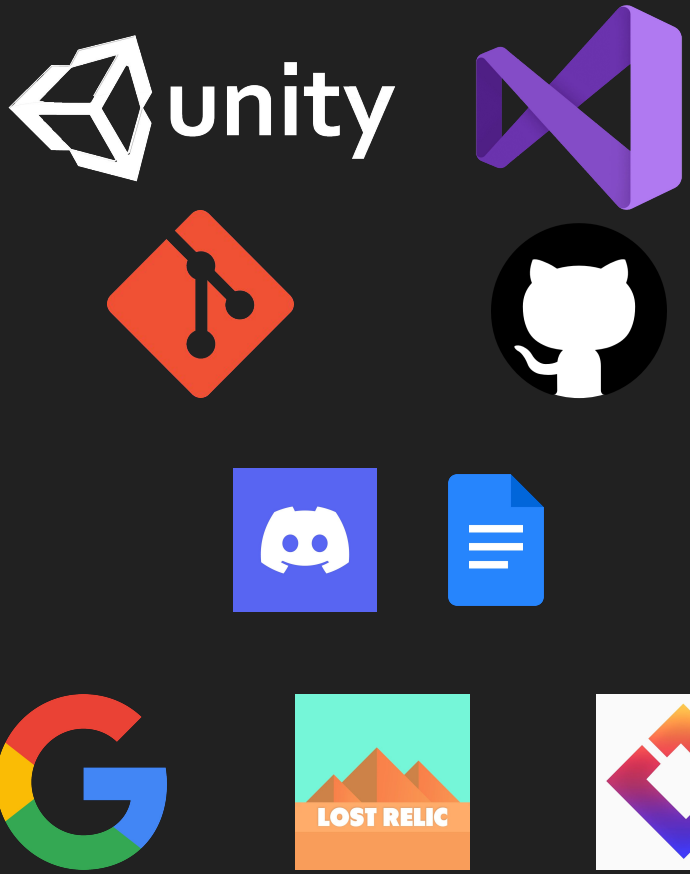


Tools

Development:
- Unity w/ Visual Studio
- Git
- GitHub
  - Issues
  - Integration (branches and PRs)

Communication/Organization:
- Discord
- Google Docs

Misc. Resources:
- Google
- YouTube
  - Brackeys.
  - Lost Relic Games.



Responsibilities

Shakeel:

- Players (the architecture, also did most of the Lancer).
- Inventory.
- Item selection (both back and frontend).
- Initial main menu & some other UI stuff.
- Facilitated the merging of branches on Git.

Jeongwook:

- Arts (*everything*).
- HUD & character selection screens.

Jino:

- Did some of the enemies.
- Worked on the level manager.
- Music & sound effects.
- Currently working on the boss!

Gary:

- Did most of the enemy behaviors.
- Vanguard and Trailblazer Abilities.
- Player Movement adjustment.

Likes

- Special abilities in a bullet hell are not new, but they are cool to use.
- “Inventory system is the greatest thing that I have ever laid my eyes on, and I cannot hope to see anything else when blinded by its glorious radiance.” - *Gary Yuen c. just now*
- The overall polish of the UI and arts is very nice.
- The music is great, unless you’ve listened to it 2,361 times.

Dislikes

- Unity (it’s a love-hate relationship).
- Git (powerful, though difficult).
  - Nasty merge conflicts when the same scene has been modified.
- The amount of effort put into the project is not reflected in the gameplay.
  - Inventory system.
  - Getting the characters going.
  - Making the HUD as beautiful as it is.

Next Time

Next time? There's a next time?

As for our game...

- Bullet Hells are incredibly resource intensive.

Game design is hard…

- Need to flush out the details of the game earlier in the process.
  - Better documentation overall.
  - This puts everyone on the same page.
- Documenting code is important!
  - Helps your team members understand your code if they need to work with it.
- Integrate earlier to iron out kinks (as opposed to building it for the first time 1 min before class).
- Better distribution of responsibilities.