03/02/2020 (Monday) (S1)

- Research and develop a really simple UI with UE4
- Implemented a simple UI with UMG
    - Changing on-screen widget
    - Handling button callback
    - Callback is implemented with C++ (inside custom game mode class)
    - Connect nodes with Blueprint (I have no idea how to make that with C++)

04/02/2020 (Tuesday) (S1)

- Research and develop UI with UE Slate (an old UI system)
    - Hard to do
    - Lack of documentation
    - Lack of tutorials
    - Community is small
- Implemented an extremely simple UI with Slate….
    - Text only, and align to top center
- Implemented a simple UI with UMG
    - Able to change to different levels
    - Still using Blueprint editor to connect nodes…. (How to exclude it from Blueprint?????)

05/02/2020 (Wednesday) (S1)

- Decided to use UMG instead of Slate
- Implemented level selection scene

06/02/2020 (Thursday) (S1)

- Made the shared perforce unreal engine project
    - P4IGNORE included
    - Added UMG, Slate, SlateCore dependencies to the project
- (**NEED CODE REVIEW**) Made the BasePlayerController and BaseGameMode C++ classes
    - Change widget
    - Input event handling, binding action and handle it

10/02/2020 (Monday) (S1)

- Controller support UI
    - Custom UserWidget version 1
    - Need to optimize

11/02/2020 (Tuesday) (S1)

- Tutorial for UI system
    - Watched (https://www.youtube.com/watch?v=G_kFa-V1rGs)
    - Repo Reference (https://github.com/iniside/ActionRPGGame/tree/master/Source/ActionRPGGame)
- Enhanced controller support UI structure
    - Moved all changing code to PlayerController

12/02/2020 (Wednesday) (S1)

- Made a demo for switching UI style by different button states
- R&D on UI resources manager, singleton pattern or HUD?

13/02/2020 (Thursday) (S1)

- Made singleton UI resources manager
- Inline review

17/02/2020 (Monday) (S2)

- Moved all UI codes to HUD class
- R&D UI animations

18/02/2020 (Tuesday) (S2)

- Moved default widget theme object to be stored in each HUD
    - Depends on the level usage
- R&D UE4 Timer
    - https://www.tomlooman.com/using-timers-in-ue4/
- Discuss with design team and asked for in-game HUD wireframe that showed in the previous inline review
- Implemented prototype in-game HUD wireframe
    - Health bar, charge bar, stone slots, character avatar, weapon display and timer

19/02/2020 (Wednesday) (S2)

- Designed the whole in-game HUD control interfaces for other developers to call/update player info box (in-game HUD)
    - Not sure it is the normal way to support multi-player, now I need player controller index to know which player infobox needs to be changed
- Made few animation clips for UI in Unreal Editor
- Integrated UI animation and controller support to LiveProject
    - Fade In and Slide-In animation @ start menu level

20/02/2020 (Thursday) (S2)

- Implemented interface methods for controlling the in-game player infobox
- Implemented Animated User Widget which able user widget to load all animation clips from itself and put it into a animation map and catch the “Show” and “Hide” animation callback for further usage
    - Such as AddToViewport and RemoveFromViewport

24/02/2020 (Monday) (S2)

- Finalised the in-game player info box interface
- Made a prototype level about the UGameplayStatics::LoadStreamLevel for making the loading screen
    - Failed, because it won’t use another game mode
    - It’s just for the in-game level transition more than loading screen
- Studied another way to make loading screen/loading asset by given name
    - Found LoadPackageAsync method

25/02/2020 (Tuesday) (S2)

- R&D LoadPackageAsync method to make a loading screen
- Integrated in-game HUD from prototype project to live project
    - Fixed camera issue
    - Fixed BaseButton bug, we can’t use a used NAME for delegate function

26/02/2020 (Wednesday) (S2)

- Integrated the in-game HUD

16/03/2020 Monday (S1)

- Setting up the StarStone project on the Perforce server

17/03/2020 Tuesday (S1)

- Implemented/ported the animated user widget from the previous project
    - Reviewed and submitted
- Implemented the UI base classes
    - BaseHUD for all HUD classes
    - IngameHUD for in-game UI control interface
    - TitleHUD for title screen control interface
- Implemented the UI assets container for applying styles to those UI elements which are sharing the same style
- Ported the in-game HUD layouts from the previous project

18/03/2020 Wednesday (S1)

- Added the title level and blueprints
    - Title HUD class and blueprint
- Implemented loading screen
    - Simple demo for now, need to wait for more information from the designer
- Found a way to change the game logo, launch image and splash screens/movie on the PS4 platform
    - I asked on the UE4 PS4 Development Forum
    - Tried to build a package and installed it on the PS4

19/03/2020 Thursday (S1)

- Review from Stakeholders on MS Team
- Submitted the TitleLevel to the Perforce
- Researched another way to make a Perforce-like versioning tools
    - Git LFS + Amazon EC2 + Amazon S3
    - EC2 for git server, I used Gitea because it supports Git-LFS
    - S3 for large file storage

20/03/2020 Friday (S1)

- Research an alternative git solution for Coders

Server Setup Instruction
Pre-installation
1. Download PuTTY
- For connecting to the EC2 server

EC2 Setup
2. Create EC2 instance
- Create a private key that we need to gain access to our EC2 server
- Download the private key during the initialization
- t2.micro is the free tier, no fee for the first 12 months

S3 Storage initialization
3. Goto AWS Control Panel -> S3
4. Create a bucket and name to `git-bucket`, and set the region to `eu-west-2` (London)

IAM Role/Permission Group initialization
5. Goto AWS Control Panel -> IAM (Identity and Access Management)
6. Create a new user under Access Management/Users
7. Set the user has `AmazonS3FullAccess`
8. Copy the `Access Key ID` and `Secret access key` during the initialization

Start mounting S3 Storage to the EC2 server
9. Follow this instruction to connect to the EC2 server with PuTTY
- https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html
10. Install Gitea to the EC2 server
- includes MariaDB, git, and Nginx
- Follow this instruction: https://www.vultr.com/docs/how-to-install-gitea-on-ubuntu-18-04
- You will create a user when you install gitea, for example `gituser`
- You might encounter an error when you try to install Gitea(Error 1709), this should help you to solve the problem: https://github.com/go-gitea/gitea/issues/2979
11. Follow this instruction to mount S3 bucket to the EC2
- https://cloudkul.com/blog/mounting-s3-bucket-linux-ec2-instance/
- You will create a folder that should automatically sync file data with the S3 bucket that we created before
12. Follow the instruction to gain access right for another user to access the mounted folder
- https://stackoverflow.com/a/30119806

Repo Configuration
13. Create a repo on the Gitea, and pull
14. Open cmd/terminal and navigate to the repo
15. Run `git lfs install`
16. Run `git lfs track --help` to see how to set up what type of file we want to be able to lock
- For example, we want to lock PSD files
- Run `git lfs track "*.psd" --lockable`
- It will update/generate a .gitattributes and you will find the configuration in it

Lock/Unlock Features
17. If you want to lock a file, use `git lfs lock "filename.extension_name"`
18. If you want to unlock a file, use `git lfs unlock "filename.extension_name"`
- You can force to unlock a file while someone is locking it, but be sure that what you are going to do
19. If you want to show all locks, use `git lfs locks`

Push and Pull
20. We can NOT use `git push` or `git pull` command to update the source because it will break the lock feature
- Use `git lfs push` or `git lfs pull` to replace the `git push` or `git pull`
- It will protect the locked file for you if someone is trying to push code to the git, which means if you locked a file, everybody can not push their commit if their commit includes the file that you locked.

23/03/2020 Monday (S1)

- Submitted in-game HUD based on the 2nd pass in-game hud wireframe
    - Include interface for a gameplay programmer
- Update title screen which is based on the 2nd pass title screen wireframe
- Made Loading Screen Hint DataTable
    - Unreal Engine C++ Fundamentals – DataTables
    - Mapping Hint Title and Hint Content
    - Loading Screen Widget will pick one of the records from HintDataTable randomly

24/03/2020 Tuesday (S1)

- Fixing the show widget issue, invalid address bug
- Submitted Loading Screen & Loading System to Perforce
- Tried to build a PC executable for Rich to test the functionality
- Requested a new splash screens video from designer
    - In case, we switch our target device to PC (No launch image for PC platform)

25/03/2020 Wednesday (S1)

- Gathered more information and requirement for MainMenuScreen from Designer
- Designed the architecture for implementing the main menu screen
    - What functions I need to call
    - What features are going to implement
- Updated BaseHUD class and AnimatedUserWidget
    - Added variables for the loading screen logic
    - Added access right for child classes to access the current on-screen widget

26/03/2020 Thursday (S1)

- Implemented the prototype version main menu screen
    - Controller support
    - Binding correct keys
    - Animations
    - Change focused item when the submenu appears/disappears
    - Tested on PS4
- Review meeting

30/03/2020 Monday (S1)

- Implemented the options menu
    - Foundation for option menu items, such as toggle group
    - Added new variables to the default theme assets
        - For setting up UI brash style for options menu items
- Integrated the main menu and some bug fixes with the Perforce project

31/03/2020 Tuesday (S1)

- Implemented the options menu items
    - Toggle Group for brightness settings
        - Toggle group logic
        -
    - Progress Bar for BGM and SFX volume settings

01/04/2020 Wednesday (S1)

- Revisit and refine in-game HUD
    - Add victory screen
    - Create a testing in-game HUD level for debug and QA

02/04/2020 Thursday (S1)

- Inline review
- Sprint planning

20/04/2020 Monday (S2)

- Self-planning for Sprint 2
    - How to integrate on the live version
        - Discuss with the combat team who are responsible for the character properties (Will A and Toyan G)
    - How to apply UI assets
    - How to test the functionality
- Confirmed who will export the UI assets from artists team
    - It will be outsourced
- Updated the in-game HUD class that connects timer widget to the HUD class
- Created demo for Tom to export UI assets
    - Used Lunacy to mockup images for him to know how to separate the UI assets in different parts

21/04/2020 Tuesday (S2)

- Updated in-game HUD to support taking pickup enum to change the styles
    - NO WEAPON ICONS NOW
- Updated In-game HUD UI Assets
    - Healthbar
    - **BLOCKERS: The player properties don't have enough information for me to get current player statuses, such as the selected character, controller index for each player and active weapon type. I asked Will Anderson and Deniss but they said there are no such tasks in their backlog. I raised this issue to Christina yesterday and hope this issue could be**
- Meeting & Tutorial
- Updated my tasks with Christina
    - Added missing tasks to Trello board

22/04/2020 Wednesday (S2)

- Updated In-game HUD UI Assets
    - Starstones
    - Player Tag
    - Player Info Box Background

23/04/2020 Thursday (S2)

- Updated in-game HUD UI assets
    - Added character icons
    - Added chargebar frame and filled image
    - To do that, I adjusted and refined some part of the layout in player info box

27/04/2020 Monday (S2)

- Updated player info box to take enum to change weapon icon
- Updated UI to use latest UI assets
    - In-game HUD - chargebar, character icons
    - Pause menu
- Credit screen

28/04/2020 Tuesday (S2)

- Created a HUD class for tutorial level to show loading screen and load level function
- Updated the pause menu UI with the latest images that are provided by Tom
- Added multiplay logic
    - PlayerControllers now will be created at the StarstoneGameMode::BeginPlay
    - By default, there will be 3 players and 1 AI character
- In-game HUD integration
    - Integrated with the function that Toyan provided to me yesterday

29/04/2020 Wednesday (S2)

- Fixed bugs
- Testing