and digital approaches to the cards however, these advantages are restricted to each approach respectively. For this reason, we believe that initially developing an improved version of previous works as well as encompassing a VR approach within could combine the benefits of both. A VR approach could also incorporate the look of physical cards as well as the ability to discuss and confer over the physical cards while at the same time allowing the analytical and relationship data to be overlaid.

There are many similarities and differences between the original physical cards and a potential digital approach to ideation cards. The first, obvious benefits of physical cards are the fact that they are real, tactile cards with the different functionalities that come with this. These functionalities include, moving cards, stacking cards, dealing cards, showing cards to other players, shuffling cards and randomly choosing cards. Along with the physical, 3D aspects to the cards, group work and collaboration is another key feature to ideation sessions with real cards. Being able to be in the same room and sharing the same experiences and space allows for users to better express their viewpoints on specific ideation scenarios. A digital approach, on the other hand, provides many similar but also different benefits. It may not provide the direct feel of physical cards and does not allow for certain functionalities like showing other players cards but it does provide other features which physical cards cannot. The first main feature a digital approach could provide is modularity, allowing for any deck to be used as long as they follow the same globalised format. This also means that testing new decks easily, becomes a reality. In order to do so, only a csv file containing card data has to be created. After doing so it can easily be imported into the system. This saves time as the creators do not have to go out of their way to physically print the cards, and instead they can simply import the CSV. This will save on both printing costs and allow the developers to recognise errors earlier on in the development process of the cards by allowing a medium where testing the decks is easily accessible. As well as this, saving ideation session data is also possible with a digital approach. Where it may be difficult to keep track of which cards have been rejected or accepted, or which state the session is in at which point during physical card sessions, during a digital card session these things are easy to achieve. This means that a loading function can also be implemented allowing for these sessions to be continued at a later time. Moreover, a digital approach can provide a very useful interface, maybe not as intuitive as a physical card ideation session, but an interface encompassing all the key features. The VR and physical approaches are both very similar with the exception of the physical touch aspect. It is for these reasons, with the few disadvantages of a digital or VR approach but the many benefits to be gained from them, that we aim to complete this project.

Previous work has been completed on this type of project, which has some of the functionality mentioned above, however, was created in an engine which does not directly support VR. We are looking to expand on the previous project's features by first porting the game over to Unity which could better utilise and play to the projects strengths as-well as create the application in a way which is more useful in the design process.

3 Related work

3.1 Lauren's Godot implementation

There are multiple sources we can draw conclusions and ideas from which are relevant to our project. The piece of work most related to our project however is Lauren's game build in Godot. This has all the core functionality we want our project to encase and build on and would have

been an amazing point to continue the work from, however upon further research we concluded that this would not be ideal due to the lack of support with VR features which are a core part of this project specification. Lauren's project also only works with the mixed-reality ideation card deck, as that is the deck which is hard programmed into the application. This is something we are aiming to improve in our version of the project by allowing for modular importation of the decks and therefore working with new decks as more can be imported at any given time. After reviewing Lauren's project we identified some of the key functionalities we will be keeping as we are making our own version of this project.

- **Key Functionalities:**
  - **Different start up conditions** - Allows the user to select how many cards can be dealt at the beginning of the round.
  - **Move cards** - Allows the user to drag the cards by clicking anywhere on their surface. Contains error if multiple cards are grouped on top of each other (will not select the card you are touching, but instead the top-most card).
  - **Write notes on cards** - A button on every card, which opens a plain text field in which the user can write down anything. This text field is attached to the card and will last through the entire game-length.
  - **Deal from different categories** - An option allowing the user to choose which category a card should be played from. Good implementation however it is hard coded and wouldn't work if any other decks were to be used. This is because the current deck only has 3 categories, while other decks may have upwards of 10.
  - **Uses just random card selection** - Works by choosing a random variable between 0 and the total number of cards stored in each folder.

3.2 Card Mapper [7]

Card Mapper was a research project based around ideation cards with two main functions, capturing "designs (expressed as either physical or digital concept sketches), into a structured repository" and providing "visualisations to explore the repository". They found that the outcomes of the project showed many potential uses for the data captured even if it was still considered an "initial technological prototype". As well as this, however, they found that capturing the basic data about the cards that they did was not enough and more must be done in future to help capture broader information. Our project will take inspiration from the conclusions of this project in how ideation session data is saved and when ideation session data is saved.

3.3 Physical Decks

As well as existing digital ideation card examples, a lot can be taken from the original, physical cards and how they are used. Ideation card decks, their rules and overall use in different design sessions shows the correct layout and way of use which should be considered carefully when implementing our VR version. There are multiple ideation card decks already released and multiple also in development. Here is a list of some of them:-

- Mixed-Reality Games Deck [4] - Deck designed to help in creating AR and VR games. Contains three different categories, Opportunity, Question and Challenge, each designed to push

the design forward in a productive way. Cards in this deck include many different things to think about when designing a mixed reality game and the three different categories help to define these different things.

- VisitorBox Deck [5] - Deck designed to help "enable heritage organisations to rapidly generate and share ideas for new visitor experiences" . Includes twelve different categories, each benefiting the VisitorBox design process in their own way.
- Other decks - Multiple other decks which deal with different themes and scensarios.

3.4 VR tabletop card games

- Tabletop simulator - Tabletop simulator is an independent video game which allows players to play and create their own tabletop games in its own multiplayer physics sandbox. Similar to the ideation card design, it has no victory or failure conditions. After choosing a game-mode they want to play (either through the community mods or through the official game modes) players can interact with the objects on screen using physics simulations. The user can spawn in and move pieces, and the game has all the common mechanics of what you can do with physical objects such as move, flip, throw, rotate, etc... The game also has features for automatic dice rolling and hiding player's pieces from one another. As well as this, there are further mechanics implemented such as saving the game or keeping a history of moves so the user can undo. Furthermore, this game has an asset store which can be searched for textures or other community game-modes.

4 Description of the work

The main aim of this project is to further the implementation of ideation cards into a digital medium by aid of virtual reality technologies. Moreover, a digital environment for completing ideation card design sessions which allows for the use of any decks and includes the ability to gather data about the sessions is another key aim. In order to reach these aims, certain objectives have already been reached and others are still currently in development. Here we will outline the main objectives we want to reach by the end of the project and which objectives have been reached thus far.

Key aims/objectives reached thus far:

- Tabletop environment - A basic tabletop environment with a deck which can be used to create further cards.
- Feed in deck from external sources - Decks can be imported via CSV file as long as they use universal format.
- Have cards and decks as objects on tabletop
- Move cards around table - Cards can be moved by clicking and dragging using the mouse.

Key objectives for the project as a whole:

- Create a fully working Unity port of Lauren's project with added functionality.

- Ability to manipulate cards as if they were real
  - Flip cards
  - Stack/group cards
  - Deal cards
  - Randomly select cards from a group
  - Show others cards
  - Move stacks as one
- Implement fully modular systems - Potentially allowing for card importation from an online database meaning the software can work even if cards are not stored locally.
- Have different startup conditions to enable different kinds of ideation sessions e.g. start with cards from set categories, start with random cards, start with different number of cards - All these can be dependent on the type of ideation session currently in use.
- Produce a VR implementation of Lauren's project to allow ideation sessions within a VR environment.
- Allow for data saving
  - Save at different states of ideation session - User should be able to close the session, and load it back in the same state at a later date.
  - Save data from session i.e. Which cards are accepted/rejected - Be able to upload this data to the document, so it can be analysed in future interactions/implementations.
- Analyse use of VR ideation session with physical ideation session - Compare outputs from each session and gather data from each.

5 Methodology

To develop this project, we have chosen to use Unity as our IDE. As previously mentioned this is because Unity contains a multitude of features which are beneficial to us when developing the project.

5.1 Unity Features

- Unity's XR Support - "XR is an umbrella term which encompassess virtual reality (VR), augmented reality (AR), and Mixed Reality (MR)". This inbuilt support allows us to branch out in multiple directions if necessary and is further extended by compatibility with multiple devices as well as forward compatibility for future devices and software which is provided for free by Unity.
- Unity Collaborate [6] - Unity also has a collaborate feature which is aimed at allowing "small teams to save, share, and sync a Unity Project in a cloud-hosted environment" Once a project has been updated, any changes made are then uploaded to the cloud. This then notifies any other users about the update, allowing them to synchronise their work with the latest version.

This includes an inbuilt merge system which lets you choose which files to keep and which files to discard. This is also further supported by an online version control system.

- Asset Store - The unity asset store is a library of free and commercial assets which are created either by Unity Technologies or other members of the community. This covers anything from textures, models and animations to tutorials and editor extensions. The interface allows us to easily manage which of these are installed or in use, and lets us import them into the project through one click. We use extensions such as Unity Debugger and VSCode extensions to further improve the experience we get with Unity. And we could also potentially use textures and the tutorials to help us with our work.
- 2D and 3D Scene support - Unity is available for both 2D and 3D design work, allowing us to customise our project to the style we need. Depending on this style we have to import textures in different ways as well as change the way our camera sees the content.
- Cross/multi platform - With very little work unity can allow for the project to be exported to match many different platforms and versions. Exporting to a webGL version is as simple as a couple of clicks away.
- Large support base - As previously mentioned, Unity has a large community which releases content for free and forums where anyone is happy to help on encountered issues.

5.2 Further software used

- Github - Although we are already using Unity's collaborate feature as a version control system, we are also keeping the main version updates backed up on Github aswell.
- Trello - We are also using Trello to keep track of all the different features that need completing as well as what is currently being worked on, what has been finished and what needs finishing.

5.3 Agile Methodology

Through the creation of this project we have chosen to use an Agile methodology with a focus on extreme programming as our main developmental approach. We thought this would best suit the project due to the ability to divide all the tasks into smaller parts. Each part will represent a different feature which needs to be completed in order to fill our requirements for the project. We can prioritize each of these tasks in an order of importance as well as group similar tasks together and give them an estimated deadline which has to be kept in order to ensure we are at the right place in our project. These individual "parts" can also function as a "sprint" (which is a development cycle within the agile methodology). We have chosen to represent these sprints through the use of a Gantt Chart which can be seen later in this document.

By choosing an agile approach, it allows for constant feedback to be gathered after each stage of production. Once a sprint cycle has been completed and the software is within working order, a meeting with our supervisor takes place in which we can gather feedback on the functionality and appearance of the software. This feedback is then used to recognise any improvements which need to be made as well as gather information on any features which currently may not be implemented in the software. Furthermore, as the project is already split into smaller parts, it allows for individual requirements to be addressed at each meeting rather than addressing them as a complete package

once the entire product is complete. These feedback gathering stages allow us to create a software which better suits user needs.

An example of where this has already affected us in the current development process is during the learning process of Unity. Initially a plan was created which listed all the features for completion using the Godot engine, but once further information was gathered and we realised that using Godot was not in our best interest, we had to change the plan and therefore our working timeline. This allowed us to easily change the requirements and the order that certain parts had to be executed in, in order to match our learning curve using the unity software. We managed to successfully allocate our self a sprint dedicated to learning and planning out the project in the different software.

Overall, we felt that the Agile principles outlined in the "Agile Manifesto" [8] were a correct fit for this project and allowed us to dictate our time better. These principles stood out to us:

- "Early delivery, creating shippable software in two-week 'sprints'"
- "Responding to changing requirements"
- "Measuring progress with working software as the metric"
- "Simplification - not making software overly complex"
- "Small, self-organising teams that regularly reflect on best practices". This has been the approach that has been followed for the developmental cycle of our application.

5.4 Research/Data Analysis

Unfortunately as the first semester for this project largely consists of making sure we have a functioning version of Lauren's project with extra functionality, most of the research/data gathering will have to wait until the second semester when we have a fully functional version with more features. We are currently planning to finish implementation of the base version of the game (without the addition of VR) by January. At which point several small groups of people will be gathered to test the program and complete some trial runs. We are going to use a qualitative approach to gather feedback from users and use a quantitative approach to gather card data.

6 Design

For this project, we initially considered making it using a 2D environment. Following is a list of conditions we considered:

- Different working conditions :
  - Group work - Would users be sitting next to each other? Or would users be sitting opposite each other? How would the group interact with the cards?
  - Individual work - Can a single user make use of the software to host his own ideation design sessions?
  - Paired work - Would the layout change if there are multiple users around the screen?
- Different viewing angles - How would users, stood at opposite sides of the table see the cards being presented to them? Will the screen have a wide viewing range to allow for users at the side to accurately see card descriptions?

- Different number of people - as above.
- How the cards are manipulated - How will the cards be moved in a 2D environment? Will it support multi-touch for multiple users to move objects at the same time?

We soon realised that using a 2D environment was not compatible with the goals we had set and neither was it compatible with VR. Therefore we decided to create the project using a 3D environment, this is largely because we wanted to give the user different options based on the way they are running the software (VR or normally). By choosing to use a 3D method we also gave ourselves a larger arsenal of tools to use, by allowing us to give the game more graphical depth through the use of shadows, animations and different camera/viewing angles. We have attempted to make the interface as clear and easy to use as possible in order to represent what a physical ideation card design session would look like.

As can be seen by the example of a physical ideation card session in Figure 1, the layout is simple including a tabletop to work on, cards laid out in front of the player, a deck to select cards from (seen on the right of Figure 1) and cards in the hand of the player which they can select from. We have tried to replicate this layout in our designs as much as possible as this is what our project is based on, so should strongly resemble. Moreover, the layout and organisation of physical ideation sessions has been carefully researched and thought about in order to work well, because of this, it is vital we stick closely its standard layout.

Figure 1: Snapshot of a physical ideation card session in play

In Figure 2, a top down design of our project can be seen. This design shows how cards will be dealt in front of players in a line with the deck on the left side (or right depending on player orientation to the screen). Along with these key aspects, other features are clearly visible e.g. the ability to change category (top left), the ability to reject cards (right), the ability to save your session or get help (bottom left) and the ability to close the session. In this top down view, the design can almost be seen as a 2D implementation which in some cases is useful to have however, due to our projects 3D nature and the ability to use different camera angles, other representations of the table will be available. These other representations will be useful for the different scenarios that the project may be used in.

Figure 3 shows a concept design of a VR ideation session. This design is effectively the same as the top down design shown in Figure 2 but "zoomed out", as to show the full 3D nature of the table and its workings. VR headsets worn by users will mean that they can walk around (and

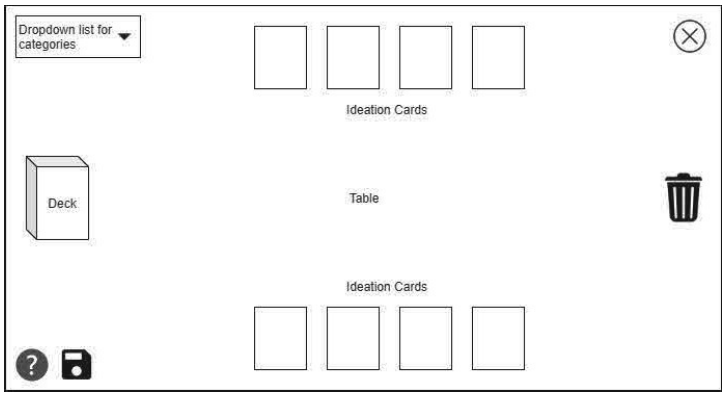

Figure 2: Top down design view of the project

through) a VR table interacting with the VR cards on it. Ideally the VR experience will be multi-user/multiplayer however the complexity of this is still unclear to us so whether our implementation is singleplayer or multiplayer is an issue which will be decided later on in the project once initial VR aims have been met.

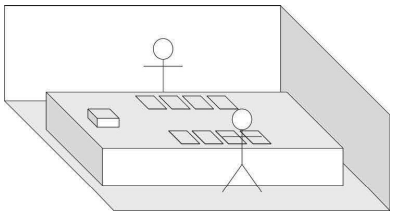

Figure 3: Design idea for a VR ideation design session

As explained in the Methodology section, the final deliverables for the project will be implemented and achieved through use of Unity. All scripts and code thus far, have been written in C# due to its connection and compatibility with the Unity engine. The majority of the rest of the projects code will be written using C# as well with the exception of a potential online database used for further modularity. This will probably require a small amount of work in PHP, JavaScript and mySQL (or similar) which we are familiar with.

7 Implementation

The current implementation of the project includes a tabletop environment within Unity where ideation cards can be created (randomly spawned) and moved around on the table. Information about the cards is stored within the objects themselves and is imported via a CSV file containing the data. Any CSV file in the correct format can be imported meaning any deck can be used as long as it is saved in this format. With all data attached to the cards, further implementation of saving states is an easy next step once the structure of data to be saved has been decided. The current

system explained above can be seen in Figure 4 which shows the workings of the implementation in more detail. Key implementation features that can be taken from Figure 4 include:

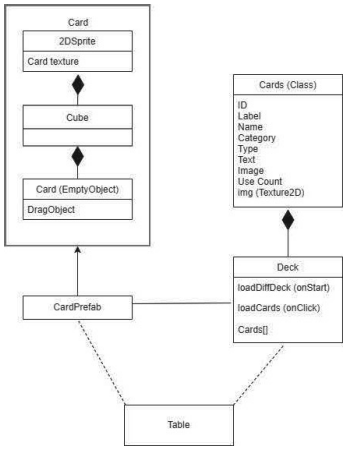

Figure 4: System architecture diagram for current implementation

- Table - Not programatically connected to other elements but used within the world space to run ideation sessions on.
- CardPrefab - A prefab object of unity which contains a "template" for card architecture. The template is made of an EmptyObject, a cube and a sprite (2d/UI)
  - EmptyObject - Contains a dragObject script which allows the cards to moved around the table and also contains the data about the card e.g ID, Label, Name, etc...
  - Cube - Provides the 3D presentation/aesthetic for the card.
  - Sprite (2D/UI) - Holds the card image which is displayed to the user. Originally the card was a single cube, textured with the card image however, the resolution was poor and the cards were difficult to read. Using a card split into different components with a sprite holding the image allows for a much higher resolution and ease of card reading which is essential for ideation card sessions.
- Deck - Contains the array of all cards imported into the session and has a script which creates a new random card from the array when clicked.
- Cards - Class for holding information about the specific cards. When cards are imported multiple instances of this class are made containing the data relevant to each card imported from the CSV. The card texture is also stored here and is created using the filepath of the card image from the Image field in the CSV.

As well as Unity implementation, a large amount of work this semester has gone into CSV importation and making it as modular as possible. CSV importation works as follows:

- A CSV file containing all the card data must be made containing the fields ID, Label, Name, Category, Type, Text, Image and Use Count in that order.

- When the user runs the current system, they will prompted on start to select a CSV file to import.
- Once the CSV file is selected, the script loadDiffDecks utilises the onStart method of the Deck to load in all card data into an array of Cards arrays. It is an array of Cards arrays because each array within the main array contains all cards from each specific category. This array is then stored within the Deck object.
  - The importation works by firstly locating and reading the file.
  - Looping through each line of the file, splitting up the fields by ','.
  - Saving each line into a new Cards object.
  - New Cards objects are stored in an array of all cards.
  - While looping through the file, storing lines in Cards objects, the code is also checking and saving the different categories into an array.
  - Once all categories are known, the array of Cards objects is looped through to split them into separate arrays by categories.
- Once all data has been imported correctly, card data is stored in specific card objects when they are "spawned" by the Deck object.
  - This works by randomly selecting a category
  - Then randomly selecting a card within that category
  - Retrieving the respective Cards object from the array and then storing it in the Unity card object.
  - The texture of the object is then loaded from the texture saved in the Cards object.

8 Progress

So far in the project, for the most part we are keeping on track with our Gantt chart, however there are certain features which we have fallen behind on. This is because as the weeks have gone on we have realised there is better ways to complete certain features so we have had to go back and alter them. This has caused us to spend time, in excess, on those features and therefore have ran over-time on the other features. The Gantt chart shown in Figure 5 is the original Gantt Chart and in accordance to it, the tasks we have completed are :-

- Learn Unity and get familiar with Lauren's project
- 2D tabletop environment
- Modular (Be able to import decks) - This task took us longer than expected to complete as we have had to change the way decks are imported into the game completely.
- Decide on direction of project(AR/VR or Intelligent dealer) - We have decided to follow on the Virtual Reality route for this project as well as further increasing the modularity of the software to work with any deck input into it.

And these are the features we have failed to implement on time:

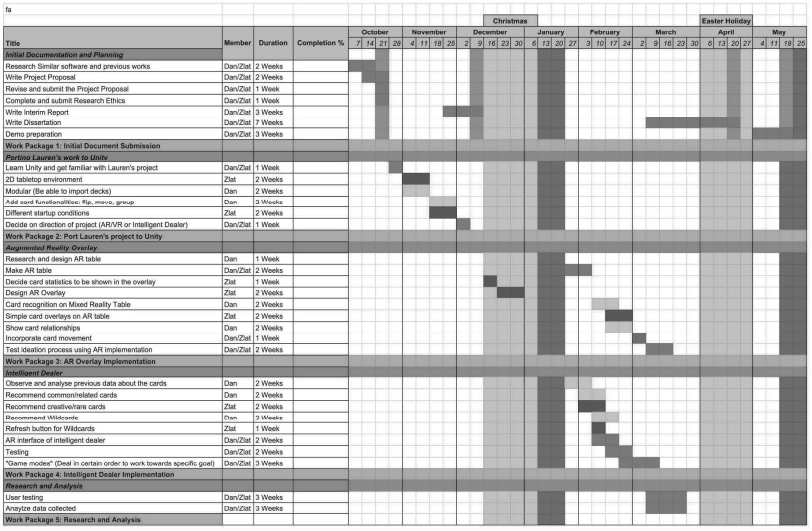

Figure 5: Original Gantt Chart

- Add card functionalities: flip, move, group
- Different startup conditions

The reason modularity as a whole has taken us longer than expected is that we needed to create an importation system which works for any deck input into the software no matter the number of categories. Initially we created a version which was similar to Lauren's Godot implementation, where we hard-coded the decks and the cards needed as well as their respective path on the disk. This worked perfectly, however as per the reason we are changing it; it would only work for a single deck. To deal with this, we had to completely modify the importation system. Initially we created a system where card data is loaded into a class through a CSV file to ensure the importation was handling the values correctly. After this was confirmed, we started working on dynamically allocating the number of categories when the CSV file is loaded, into a different array List. This would automatically scan through the file and each time a category which is not present in our secondary array is located, it would simply add this category to the array. This means that if a deck consists of five unique categories, those five unique categories would be found on importation and added to the deck. This would work in a similar manner if the deck had thirteen unique categories. Programming the modularity in this way is extremely important to the software as we cannot guarantee how many categories will exist in any future decks, therefore not setting a limit to it allows it to work no matter the category number. Once this was confirmed to be working, we now needed to scan through each category and check the number of cards connected to it. So we then further refined the importation so once it scans all the categories, it would once again look through the list and find all cards belonging to that category and link them together. This is important as it

will benefit another feature we are to create in the future, which is the feature to allow the user to choose which category they would like to deal cards from. Figure 6

For future implementation, the following is a list of features we would like to implement:

- Simple VR table environment
- Card movement in VR
- Card Mechanics in VR - As mentioned previously, features such as flip, rotate, throw away, pick up cards, potentially show other people cards. General mechanics which would work on physical cards.
- Potential VR Multiplayer - We are currently not sure to the complexity of creating this therefore we will dedice in the second semester if this is possible.
- Save and Load functionality - The user should be able to save the game at any time and at any state and resume that state later on.
- Help Screen for VR.
- Online database implementation of Decks - Move all cards to an online source, rather than having them stored locally on a machine. The CSV file would simply need to contain a URL to the location where the card is saved and it would be fetched at run-time when the card is called or the deck is chosen. This allows for the program to be easily used on any computer even if the cards are not stored locally.
- Data Overlay - Potential data overlay to show connection between cards based on previous game design sessions to show overall compatibility.
- General makeover of the software - This is required to make it look more professional and usable in an industry standard.

9 Project management

In practice, as a group we work well together. We have both stuck to the tasks we set for ourselves well overall. Some setbacks have occurred during the process which have been explained previously in the Progress section but have been handled well by the both of us. We have assigned time during each week where we have gaps between lectures to meet up and complete project work, this accumulates to approximately three/four meetups a week. We feel the is sufficient to get the work needed, done. Along with this, individual work is key as many tasks set have been assigned to either of us specifically so for this reason we both spend a large amount of time seperately on individual work as well. In terms of meetings, we have successfully met up with our supervisor, Steve Benford every two weeks after sprints have been completed. Some meetings have needed to be delayed, once for an uncontrollable reason and another where our work had been delayed by other deadlines. We decided to delay as we believed it more beneficial to have a meeting with more substantial work to discuss than go to our allotted session with uncompleted work. Meetings, on the whole, have been successful. Our first official meeting was fairly unorganised and unstructured due to lack of

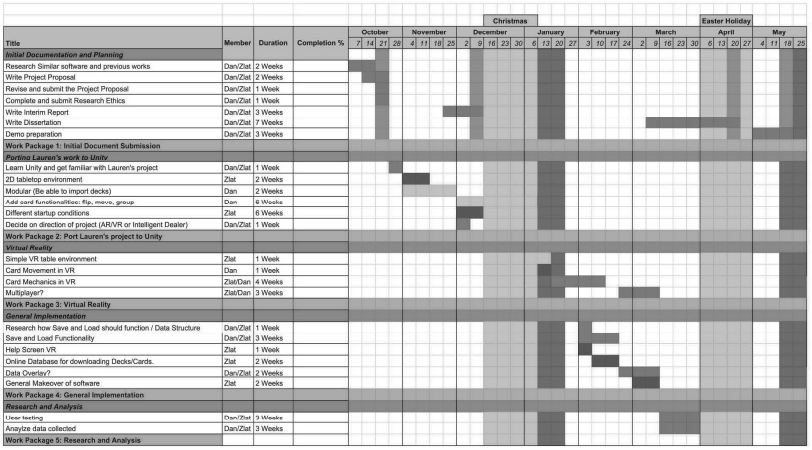

Figure 6: Updated Gantt Chart

planning however, after that we went into each following meeting with talking points to discuss, updates on progress and any questions ready at hand. As well as this, notes from most meetings have been made to ensure maximum content and information from meeting is remembered. We have also had multiple meetings with Dimitri, who is currently working as our lead technical advisor as well as acting as an external sponsor due his needs for further implementation of the ideation cards. Meetings with Dimitri, usually happen in between our scheduled meetings with Steve and are more informal in their planning. If we have problems or queries about the ideation cards, Dimitri is usually our first port of call. Some meetings with Steve have also involved Dimitri, these group meetings are very worthwhile as we are all able to discuss every aspect of the project and each person's view on the topics. This is useful because everyone is coming at the project with different assumptions and aims, so meeting with everyone allows us to find a middle ground to work with. All meetings have included equal involvement from both of us and we have both contributed valuable ideas to the project. In terms of coordinating the work, we have evenly split tasks depending on how long we think they should take as well as taking into account our individual strengths and weaknesses to determine who would be better.

10 Contributions and reflections

The scope of this project has increased to support features we had not thought of in the original design process. More work related to the modularity of the project has been done and further has been allocated for the second semester. This is especially important as the software needs to

be easily scalable in the future if necessary, therefore laying down a stable foundation is crucial. With the groundwork for deck importation already being laid down allowing decks of any size to be imported and used, future work such as an online database to store the cards as well as download from could be implemented. These additions have increased the difficulty of the project, but at the same time we now believe they are a part of the core requirements. We are hoping that these extra features will make this a more fulfilling project which has gone above and beyond the original specification.

Overall, we are happy with the stage we are currently at in regards to work completed. However, we believe that more work needs to be completed in the second semester in order for this software to meet both ours and our supervisors expectations. Whilst we have fallen behind on a couple of the features such as the card mechanics, this is because we have had to change features to better work in any scenario with any deck. We believe the features we have missed out on can be implemented before the start of the second semester as to not impede with our projected Gantt chart. In the new semester if our work done is up to date with the Gantt chart, we should try and proceed further rather than getting complacent. This is especially true knowing that we may encounter unpredictable issues to do with the VR implementation.

References

[1] Michael Golembewski, Mark Selby. *Ideation Decks: A Card-Based Design Ideation Tool.* DIS'10, pages 89-92, Aarhus, Denmark.

[2] Christiane Wölfel, and Timothy Merritt. *Method card design dimensions: a survey of cardbased design tools.* IFIP Conference on Human-Computer Interaction, Springer Berlin Heidelberg, 479-486, 2013. `https://doi.org/10.1007/978-3-642-40483-2_34`

[3] Lauren Robinson, Steve Benford. *Digital Ideation Card Decks: A Study of the Importance of Tangible Artifacts in Design.* 2018.

[4] Richard Wetzel, Tom Rodden, Steve Benford. *Developing Ideation Cards for Mixed Reality Game Design.* 2016.

[5] Ben Bedwell, Katharina Lorenz, Steve Benford. *VisitorBox.* 2018.

[6] *Unity Collaborate.* `https://docs.unity3d.com/Manual/UnityCollaborate.html`

[7] Dimitrios Darzentas, Raphael Velt, Richard Wetzel, Peter J. Craigon, Hanne G. Wagner, Lachlan D. Urquhart, Steve Benford. *Card Mapper: Enabling Data-Driven Reflections on Ideation Cards.* CHI '19 Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. Glasgow, Scotland.

[8] *What is agile development?* `https://www.itpro.co.uk/agile-development/28040/what-is-agile-development`. 2019.