# The English as a Second Language flashcard system

# Project Overview

## Topic

The project is to create a flashcard system for teaching basic English nouns to children whose primary language is not English.

A flashcard is a combination of an image of an object with its name and a sound file demonstrating how the word is pronounced. The software will teach basic English nouns to children using a flashcard system. It will have mini games to keep the children interested and engaged through the process. Users will be able to customise the app by adding their own flashcards. The customisation will empower the end-user to improve the app for use in ways that fit their ESL curriculum.

We intend on building a small, focused application that provides a simple tool to help with this specific part of language acquisition. The product will be available as a standalone website and on app stores. We also hope to make the software offline functional to enable use by remote communities who don’t have high-end technology or qualified teachers to teach English.

## Motivation

People worldwide consider English to be the most valuable language to learn as a second language (Taylor, 2020). Research has predicted that the market for learning English as a second language (ESL) will grow 7.1% and is expecting to hit the $54.8 billion mark by 2025 (Research, 2020). Our motivation is to tap into this growing market and help children learn some English in areas of the world that do not have access to native English speakers to teach them. Vietnam, for example, where there is a documented lack of English teachers available to meet the demand (News 2020).

Completing this project would show that our team was able to work together to produce a viable product. It would demonstrate that we could identify a need and craft a solution that fits within it. We consider success to be a high adoption rate. If successful, our project will an example of us using our skills to help the global community.

## Landscape

There are many existing applications in the market for teaching children second languages, especially English, as it is the most popular second language. A popular ESL flashcard system aimed at ESL kids is called Lingo Kids ([www.lingokids.com](http://www.lingokids.com)). We see three significant differences between our application and Lingo Kids, being:

1. ours can be customised, by adding new flashcard decks
2. We will allow people to share their customisations
3. We will also provide our application one hundred per cent free

Many of the competitors in this market seek to generate income from selling in-app advertising. In adopting this strategy, our competitors inherently geared towards more affluent urban areas that can access the items advertised.

For our motivations, market penetration (numbers uptake) is more important than profitability – our motive is to demonstrate our IT skills. We hope our strategy of not charging, or including advertising, will make the application more favourable for kids in rural, less wealthy areas. Because of this, we are open ourselves to a significant market segment that other competitors don’t value highly.

# Aims and Goals

## Project Aim

#### 1.0 Build an English as a Second Language flashcard system.

We aim to develop a basic working ESL flashcard application that can:

1. dynamically load flashcards
2. Initiate a the basic operation of cycling through the flashcards

### Enhancement Aims

#### 1.1. Add mini-game – choose the correct card

The mini-game will make learning with the flashcards more engaging of children.

#### 1.2. Add mini-game – memory game

The mini-game will make learning with the flashcards more engaging of children.

#### 1.3. Add flashcard customisation

The customisation will allow users to add new flashcards by combining a word with an image. The user will then be able to store their new cards in custom decks.

#### 1.4. Add customisation sharing options

Users will be able to store custom decks on the cloud. The community of users may search and download any custom decks where the owner has made them to publicly accessible. We envision forceful censorship to ensure the appropriateness of the content is maintained.

## Project management aims

#### 1.4. Create a presentation video

The video will outline what the project is and why it is worthwhile to build. Our marketing video will showcase our software to people who would be interested in it (I.e. schools, parents, teachers or education companies).

#### 1.5. Create a web site for the project

The web site will act as an access point for the project and expand as the project matures. The site will be an access point where people can download the software and extra flashcard sets. People may also view documentation or marketing material related to the product.

## Goals To achieve project aim 1.0

#### 1.0.1 Outline the core functionality of the application

List the core functions the application will provide.

#### 1.0.2 Draft a menu workflow

Decide how the app will flow between functions and draw up a menu structure that will enable this.

#### 1.0.3 Create a workable main menu and .exe package

This goal would be to create a working main menu prototype for our ESL flashcard software that will run on a Windows operating system from an executable file.

##### 1.0.3.1 tested menu functionality

##### 1.0.3.2 test user experience (UX) to ensure the GUI menu is clear and easy to navigate

##### 1.0.3.3 document menu

##### 1.0.3.4 add a background image

##### 1.0.3.5 add some background music.

#### 1.0.4 Create a basic set of twenty flashcards image files.

This goal would require us to source royalty-free images or create our own. The goal would be to have at least twenty of these to showcase the software. Match the images files to the English noun.

##### 1.0.5 Create a matching set of audio recordings for the images.

The goal here would be to have an audio recording for each flashcard image file we have (demonstrating the pronunciation). The audio recordings are to be brief, clearly spoken and match the English noun for the flashcard.

##### 1.0.4.1 test the audio files satisfy the criteria (outlined above)

##### 1.0.4.2 test images, audio and word (noun) match, and are comprehensible to the target audience

#### 1.0.6 Create a dynamic flashcard loading system.

The application will have a dynamic flashcard loading system that uses the files stored in the flashcards folder. The app will cycle through the folder files and compile them into flashcards. The flashcards will be grouped into decks and arranged within that deck in a defined order. No matter how many cards are in the folder, the software would need to be able to add them dynamically. This feature would also need to be tested for bugs and usability as well as having an easy to access

##### 1.0.6.1 document possible errors and how they will be handling

##### 1.0.6.1 test - the flashcards components are matched correctly

##### 1.0.6.2 test - that the app handles missing components in an appropriate way

##### 1.0.6.3 test - UX that the app reports missing components to the user clearly

##### 1.0.6.3 test – 0, 1 or 1000 flashcards do not cause app failure

#### 1.0.7 Create the game-play engine

Upon the selection of the appropriate menu item, The application starts the game mode. Once entered the game will run as documented. The app will exit the game-mode upon request and return to the home screen.

##### 1.0.7.1 write the engine

##### 1.0.6.1 test – the app initiates the game correctly

##### 1.0.7.2 app exit the game correctly

##### 1.0.6.2 the game runs as documented

##### 1.0.6.3 test - UX that the game fits the screen correctly and the target clients can operate the controls

## Goals To achieve project enhancements

#### 1.1.1 Create a working mini-game (Choose the correct card).

Our goal here would be to create a single mini game that is working and tested. The first mini game to create would be the ‘choose the correct card’ game. The specifications for this game are detailed later in this document.

##### 1.1.1.1 review specifications and write a list of achievable goals

##### 1.1.1.2 implement the goals

#### 1.2.1 Create a second working mini-game (Memory).

Our goal here would be to create a second mini game that is working and tested. The second mini game to create would be the ‘memory’ game. The specifications for this game are detailed later in this document.

##### 1.2.1.1 review specifications and write a list of achievable goals

##### 1.2.1.2 implement the goals

#### 1.3.1 Add flashcard customisation

The goal is to allow users to add new cards, and store them into decks

##### 1.3.1.1 develop a workflow for importing the required components from the client

##### 1.3.1.2 implement the workflow

##### 1.3.1.3 test the implementation

#### 1.4.1 Add customisation sharing options

The goal is to have a method where the clients can load their customisations to the cloud, where other clients can download and use the new flashcards.

##### 1.3.1.1 develop a security and appropriateness strategy

##### 1.3.1.2 develop a workflow for cloud storage

##### 1.3.1.3 implement the workflow

##### 1.3.1.3 test the implementation

## Goals To achieve project management outcomes

#### 1.5.1 Create a presentation video

##### 1.5.1.1 generate video ideas and vote amongst the group

##### 1.5.1.2 expand the successful idea with storyboards and script outline

##### 1.5.1.3 shot the video

##### 1.5.1.4 edit the parts together

##### 

#### 1.6.1 Create a web site for the project

##### 1.6.1.1 generate a basic structure as a Github page

##### 1.6.1.2 plan future development

# Plans and Progress

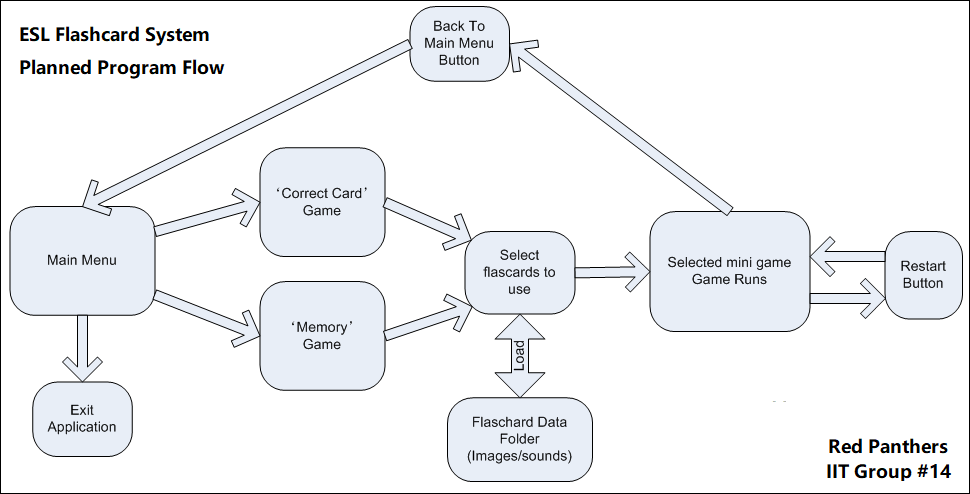
## The Plan

### Planned features and flow of the software

The ESL flashcard system will start with these features, with the possibility to expand and add more features later as needed:

* The main menu
* A collection of stock flashcards (showing a picture and the English noun below it)
* A flashcard selection screen to choose the appropriate flashcards
* A system which allows users to add flashcards
* A ‘Click the correct card’ mini game
* A ‘Memory’ mini game

Planned Program Flow Diagram:



### The plan for the Main Menu.

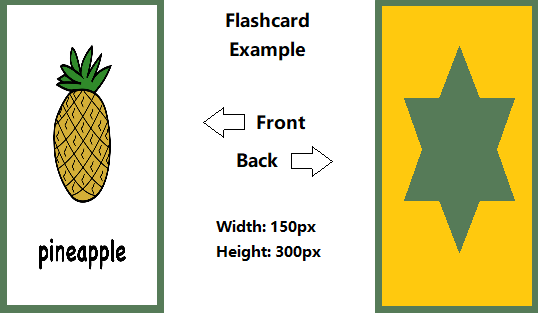
The main menu will consist of three buttons:

* ‘Correct Card Game’
* ‘Memory Game’
* ‘Exit Application’

Clicking either of the game buttons will proceed to load up the selected mini-game, whereas the exit button will exit the application. The application will have an appropriate background image and a simple music loop.

### The plan for the Flashcards

The team will make a basic set of twenty flashcards to the following specifications:



All flashcards will also have an accompanying audio file that reads the word on the flashcard in a .wav format.

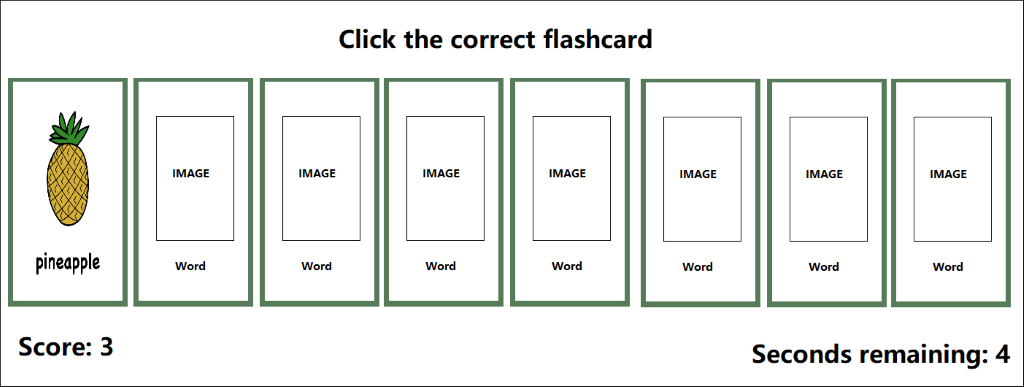
### The plan for the ‘Flashcard Selection’ system

All flashcard image files, and audio files will be placed in a single folder in the applications root directory in an images folder, underneath a flashcards folder: (flashcards/images). Each flashcard image will be named based on the card in the .PNG image format, say for example ‘pineapple.png’. A corresponding audio file in the .WAV format will also be placed in an audio folder in that same flashcards directory (flashcards/audio) and will be named to match the image name (I.e. pineapple.wav).

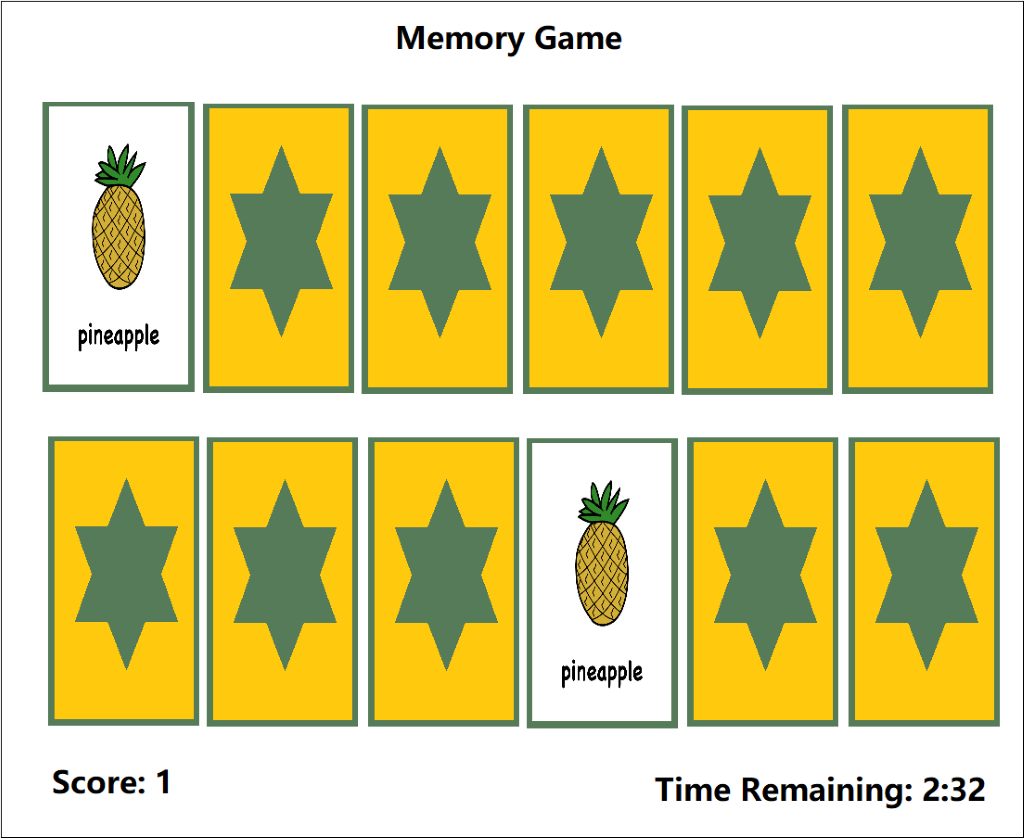
The application will then dynamically load the image and audio files to create the flashcards in the applications flashcard selection page. The user will then have to select six flashcards. As the user clicks flashcards, they will turn green. Once the user has chosen six flashcards, they will continue to the mini game they had previously selected.

### The plan for the mini game ‘Click the correct card.’

A row of six flashcards will be displayed face up. The application pronounces one of the cards. The user then has 10 seconds to click on the matching flashcard. The user gets one point for a correct answer, or one point deducted for a wrong answer. The round will end when the application has finished reading all the flashcards. The game will not repeat the same flashcard twice.



### The plan for the mini game ‘Memory’

A grid of flashcards is displayed, which is each of the six flashcards loaded twice, then distributed randomly on the screen. The flashcards are face down. The user will then click flashcards in pairs, looking for matches. As the card turns over, the audio plays. The user gets a point every time they correctly reveal a matching pair. The game has a timer that gives the user a time to beat for the round. The round ends when the user has matched all the pairs, or the timer runs out. A final score is given based on the time remaining as well as how many pairs the user successfully matched.

### The plan for the marketing presentation video

We will create a basic video advertisement for our flashcard system. The video will showcase all the features of our program as well as show a small sample video of a non-native child using this application.

### The plan for the website

We will create a simple website to show the product's features as well as allow the user to download the application onto their system. This website will show system requirements as well as documentation on how to use our software.

## Progress

### Programming Development

#### Setting up the Unity project

The project itself is created using the name ‘FlashCardSystem’. The settings were changed to 2D, rather than 3D as the program does not utilise the 3D engine of Unity.

**//more to add here, for example global objects etc, scenes list**

#### Global Project Assets

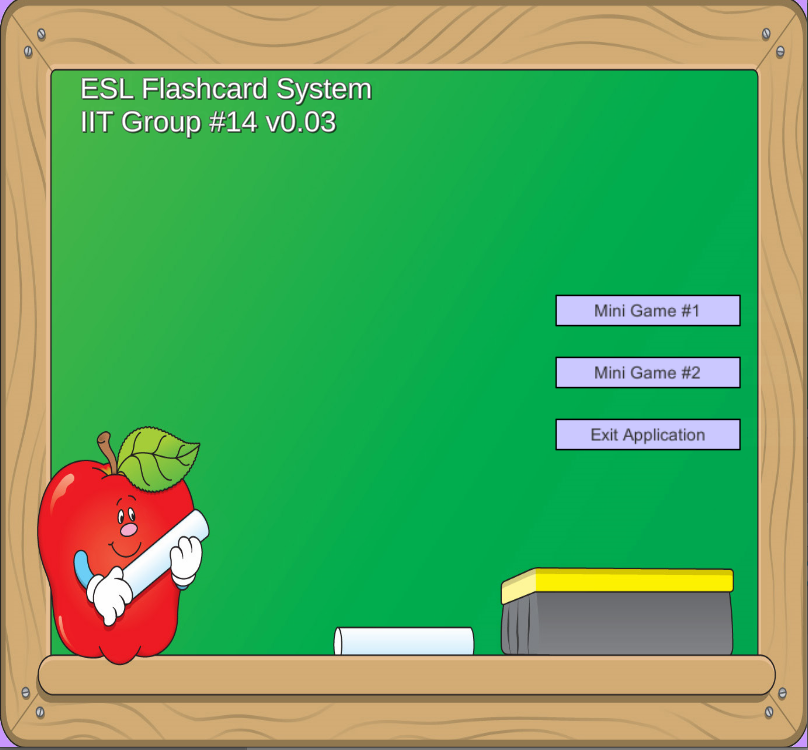
The project itself will consist of several unity scenes, which are containers which hold the game objects alongside scripts. As of this version (v0.03), there are four scenes currently setup:

* ‘Main Menu’: The main menu users see when they enter the application.
* ‘Flashcard Selection’: The scene that allows users to select which flashcards they will use.
* ‘Mini Game 1’: A scene that plays a simple ‘Choose the correct flashcard’ mini game.
* ‘Don’t Destroy on Load’: (Global scene that is always open)

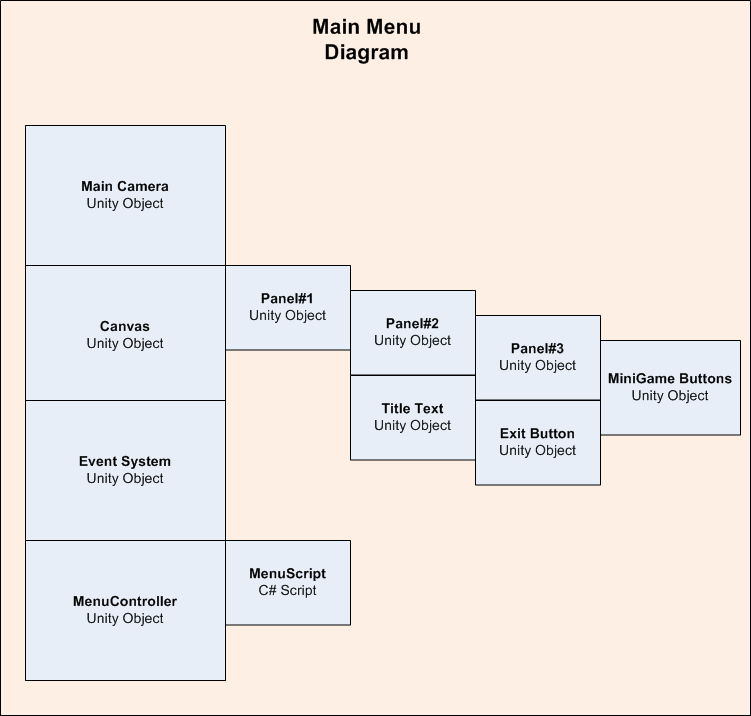
#### The main menu Scene

The main menu consists of a background image, an exit button, and a list of buttons for each mini game.

As of version v0.03, this is the current display that the user will see when they load the application:



##### Scene Diagram:



##### Main Camera

This is the main camera that focuses the screen, the settings were configured to use an orthographic 2D camera set to five units away to fit the main menu on the screen.

##### Canvas UI

This was created by using a unity ‘canvas UI’ object, and some smaller panels inside it. One panel is the overall large panel that displays the background image, the next panel is a smaller one to hold in the title text as well as the buttons. The reason UI is used is to ensure that no matter the screen resolution all objects will fit on the screen. Inside the Canvas UI is Panel #1. This panel keeps all the UI objects stretched onto the screen as well as holds a background image, sourced from the royalty free site (http://clipart-library.com). Inside Panel#1, is Panel#2. This panel holds the Title Text object which displays the title of our application.

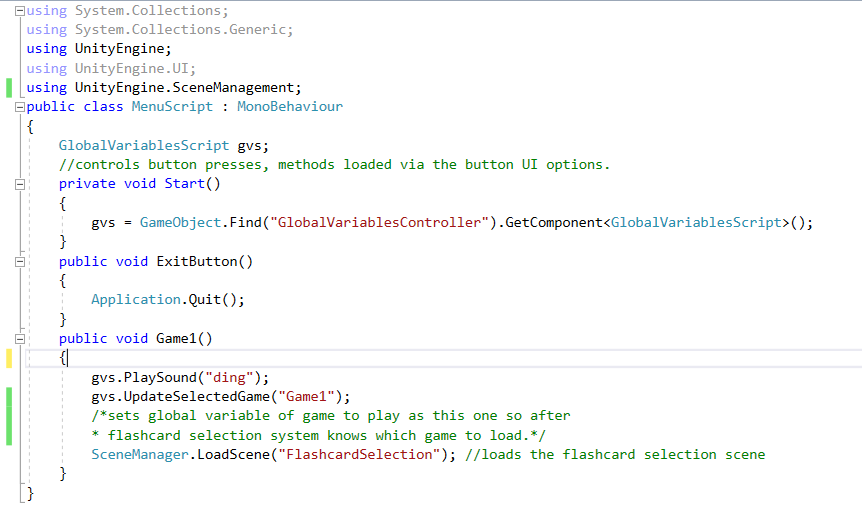
Inside Panel#2, is Panel#3, which holds all mini-game buttons. These buttons are used to launch the minigame. They run a unity UI button touch function to launch a function on the ‘Menu Script’ C# script, depending on the button pressed. As of this version (v0.03), only one mini game is working so only the ‘Mini Game #1’ button is active. The exit button is simply there to load the ‘Exit Application’ function and quit the application.

##### Event System

This is a default unity object that is created to register clicks on the ‘Canvas UI’ objects and buttons, it is created by default and can be ignored for now.

##### Menu Controller/Menu Script

This is a unity object that is created to hold the Menu Script. The menu script controls button presses and sets up the system based on these presses to register the correct button pressed, saved the game selected in the global variables controller and then load the flash card selection scene.

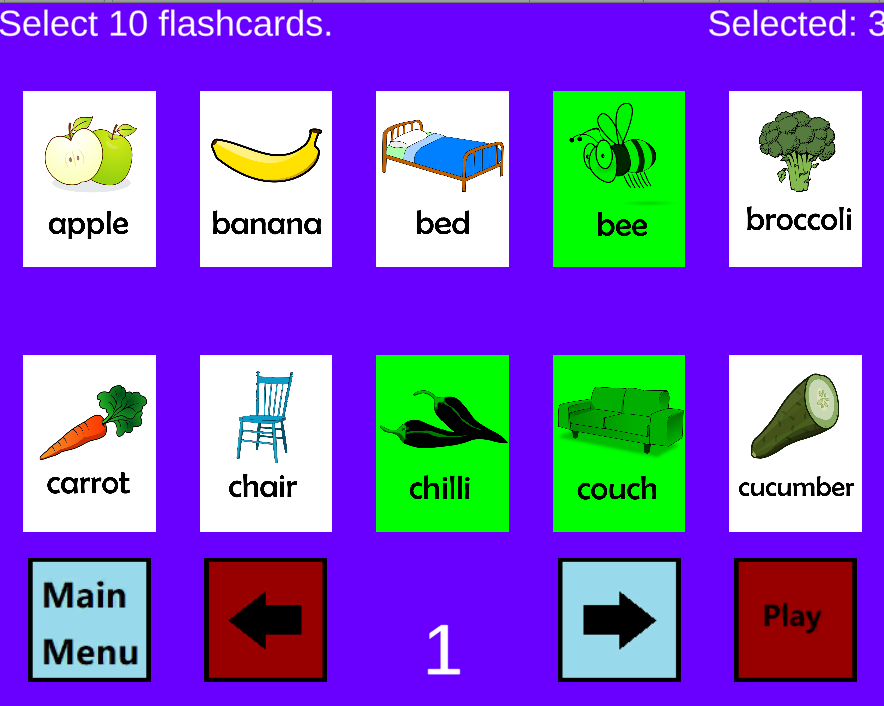


#### The Flashcard Selection Scene

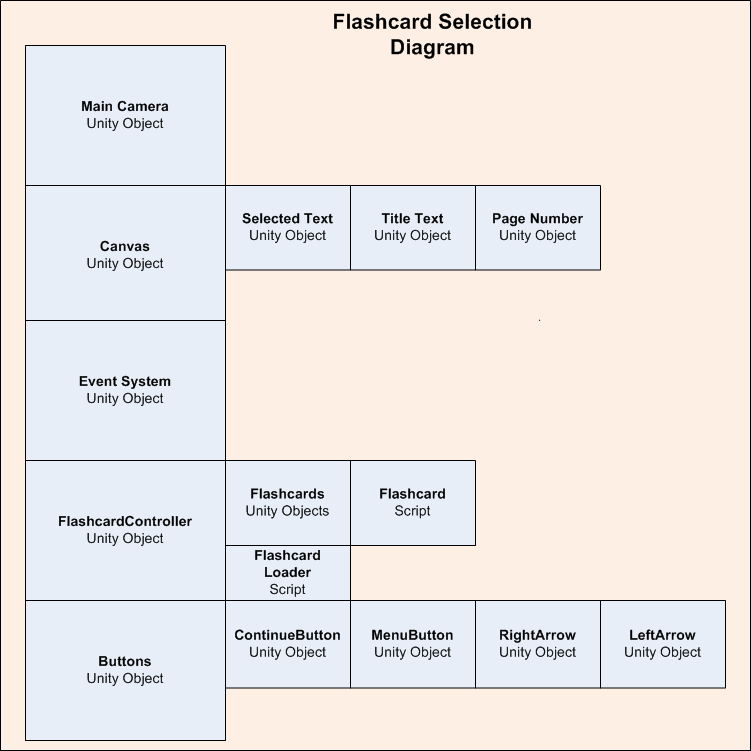
The flashcard selection scene is loaded after a user clicks one of the main menu buttons. This scene has three general tasks:

1. Dynamically loads flashcards by pulling both images and audio from the ‘flashcards’ directory in the application root.
2. Displays the flashcards on a table of cards, with two rows, five cards per row.
3. Allows a user to select the ten flashcards they will use in the mini game they have selected.

As of version v0.03, this is the current display that the user will see when they load the flashcard selection area of the application:



##### Scene Diagram:



##### Main Camera

This is the main camera that focuses the screen, the settings were configured to use an orthographic 2D camera set to twenty units away to fit the main menu on the screen. This is a difference in distance as opposed to the main menu

##### Canvas UI

This was created by using a unity ‘canvas UI’ object, in the flashcard selection scene it is simply used to display the three different text objects, ‘Selected Text’ which tells the user how many flashcards they have already selected, ‘Title Text’ which tells the user to select ten flashcards and ‘Page Number’ which shows the current page of the flashcards you are viewing.

##### Event System

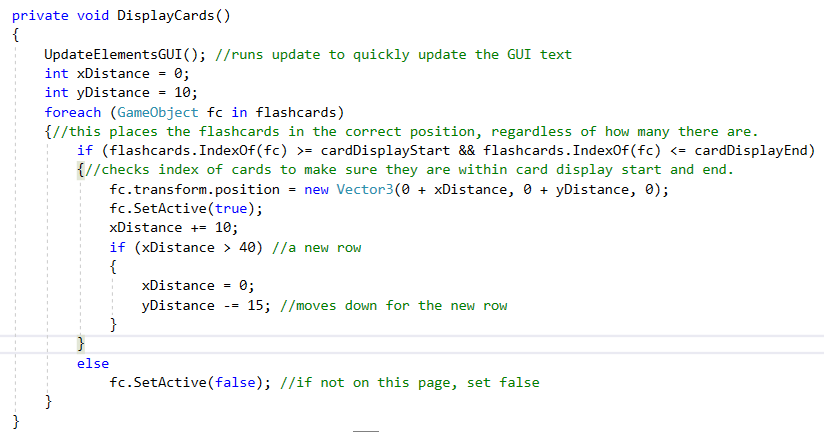
This is a default unity object that is created to register clicks on the ‘Canvas UI’ objects and buttons, it is created by default and can be ignored for now.

##### Flashcard Controller / Flashcard Loader Script

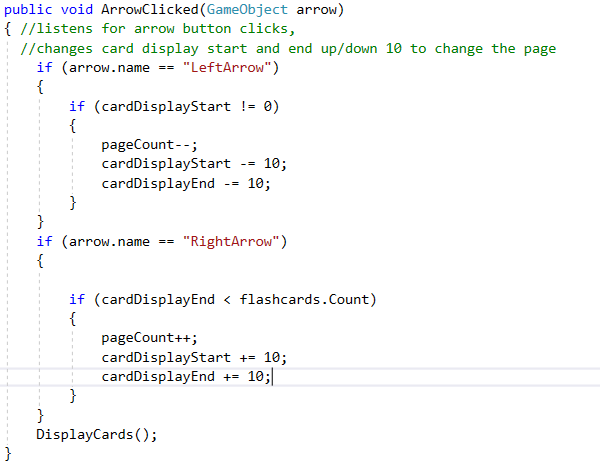
This is a unity object that is created to hold the Flashcard Loader Script. The flashcard loader script is responsible for reading all files in the applications root directory, flashcards folder and creating the unity game objects for these flashcards.



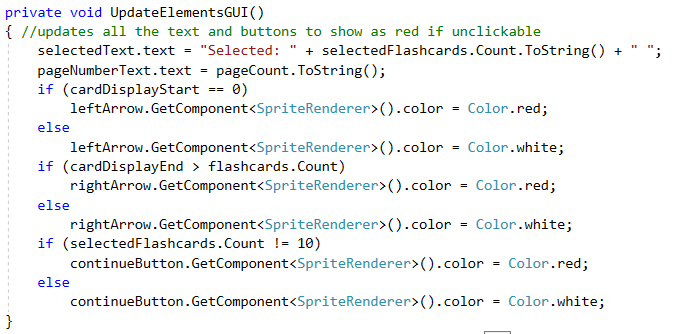
1. Loads all images in the /application directory/flashcards/images/ folder.
2. Loads all images in the /application directory/flashcards/audio/ folder.
3. Matches these files together and creates flashcard objects that will have both the image and the audio files.
4. Moves on to the display cards method.



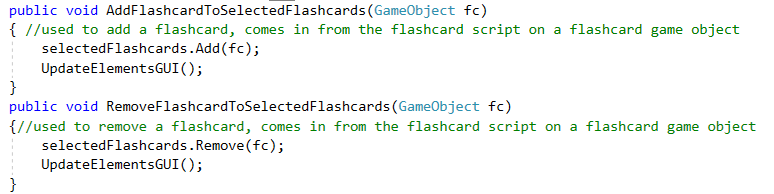
Here the application displays all the created cards in two rows, with five cards per row. If there are more than this, they will not be loaded unless the user changes the page.



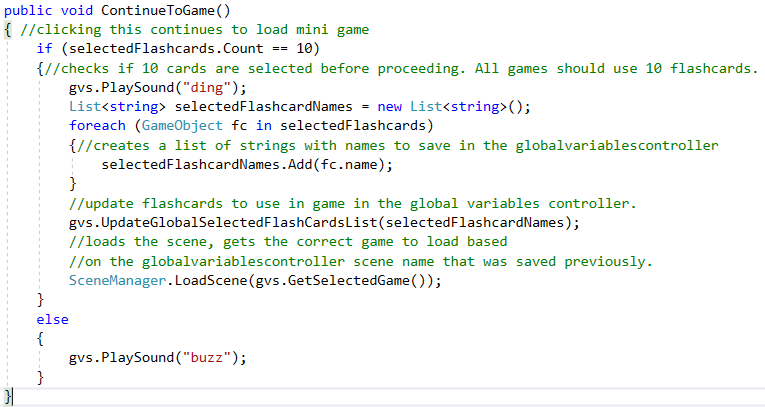
Here the application awaits either ‘Left Arrow’ or ‘Right Arrow’ buttons to be pressed, it then changes the page of the flashcards the user is viewing. If the user is already at page 1, it will not allow the user to continue going back, as well as if the user is at the end of the flashcards, the user will not be allowed to continue forward.



This Update Elements GUI function is run every every time a button is pressed to update the graphics user interface (GUI) of the user. It changes arrows to red if they are unavailable to be pressed, as well as the continue button to red if the user has not selected enough flashcards to proceed. It also increments the selected flashcards text so the user can see how many flashcards they have already selected.



These functions come in from the flashcard scripts on the flashcard objects and add/remove them from the selected flashcards list. The selected flashcards list is used to be passed into the next scene, the selected game from the main menu, so that the users correct flashcards are used when they play their selected mini game.



This Continue to Game function is used when the user clicks the ‘continue’ button the GUI. It first checks if the user has selected 10 flashcards, then proceeds to save the name of all the selected flashcards and saves them in the global variables controller to be carried onto the next scene. If the user has not selected 10 flashcards, the user will hear a buzz and not be able to continue.

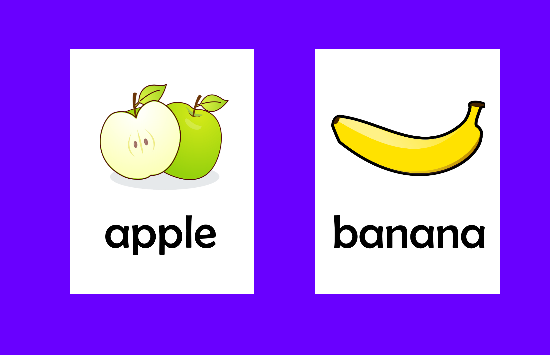


This last function is a generic function used to load files from the computer system. It is used by the flashcard loader to pull both images and audio for the flashcards.

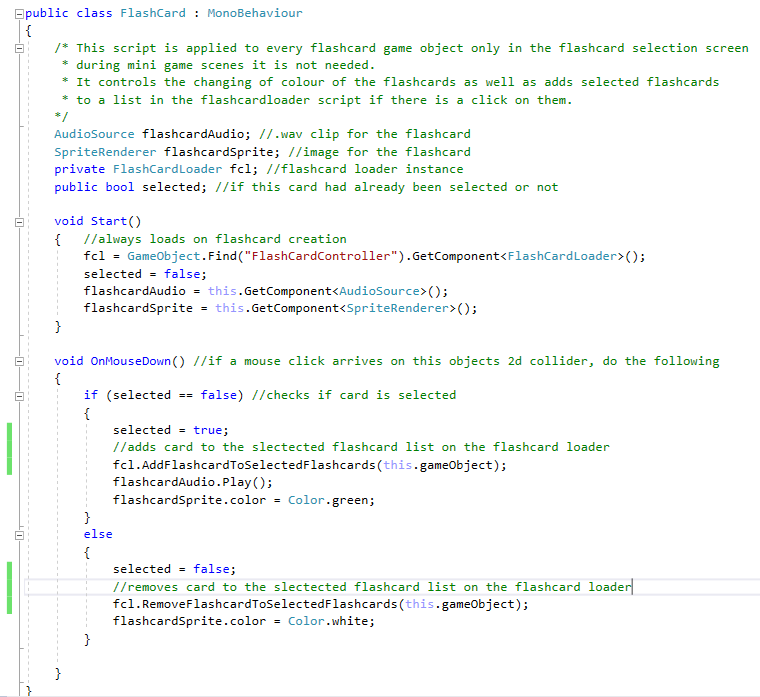
##### Flashcard Game Objects / Flashcard Script:

The flashcard game object has both an image an audio attached to it. It is created by the flash card loader script. This is the game object that physically shows up as card in the game itself.

Here is an example of two of these objects appearing in the flashcard selection screen.



Attached to these flashcard objects is ‘flashcard’ script. This handles users' interactions with the flashcard's game objects.

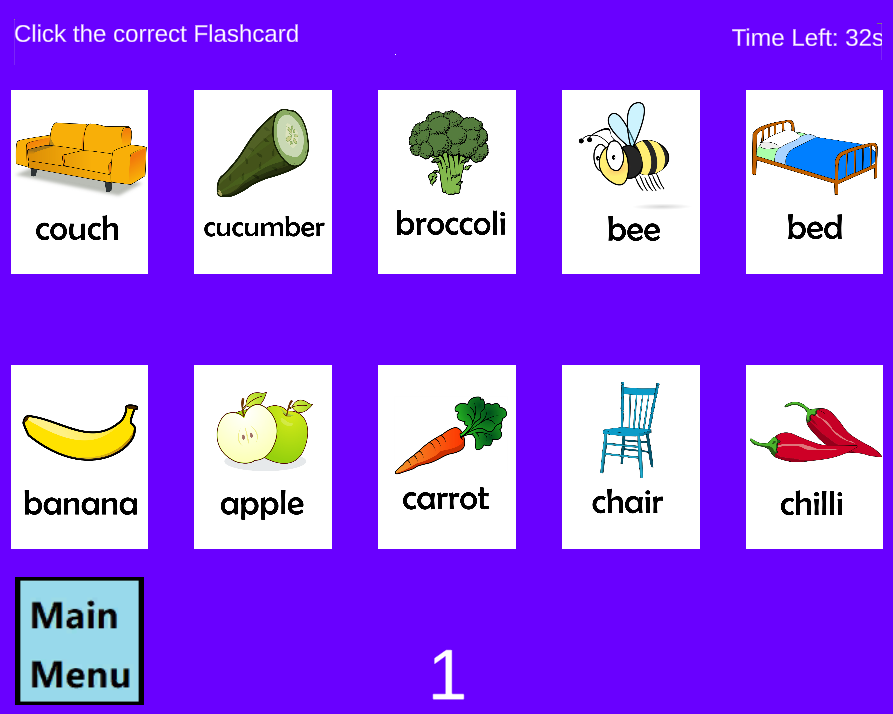


The flashcard script simply listens for a click on the flashcard game object, then sends passes itself to the flashcard loader script when pressed. If it has already been pressed, it sends the request to deselect it.

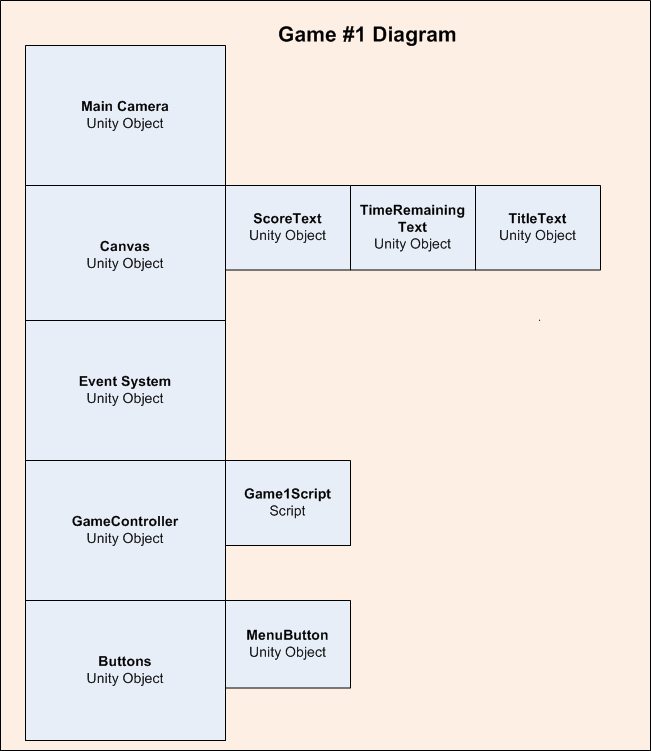
#### Game #1 Scene

This game is created with the following configuration, first the user hears audio from one of the cards at random, then they must click on the cards. If they click the incorrect card, it turns red and one point is deducted. If they hit the correct card, a point is added, and a new cards audio is played. The user has 60 seconds to get as many points as possible. When the time runs out, the user can no longer click on anymore cards.

Here is an example of the Game # 1 scene as of v0.03.



Here is the diagram for game #1.



//will finish writing this later – JW

#### Application version log:

V0.01

--Creation of Unity Project.

--Creation of Main Menu Scene.

--Creation of Buttons for Main Menu.

--Creation of Main Menu UI.

--Creation of Main Menu Controller.

--Basic Main Menu Scene completed. Will need to revise and update in a later version.

V0.02

--Creation of flashcard selection scene.

--Creation of flashcards selection buttons.

--Creation of flashcard loader script.

--Creation of the flashcard script.

--Flashcard Selection scene complete.

V0.03

--Creation of Game #1 Scene.

--Creation of GameController.

--Creation of Game1Script.

--Game #1 Scene complete.

V0.04

--Resolved issue with flashcards not being able to be dynamically loaded once the application is built.

--Updated and removed placeholder assets that were used during initial creation. Now only loads flashcards via dynamic loading from the flashcards folder.

--Marked main menu button game # 2 as ‘under development’.

#### Programming still to do:

##### Game # 2-?

The project still needs the addition of more mini games, starting firstly with the planned-out memory game that will need to be created. This can be created following the similar pattern of how game # 1 was created.

##### Cloud integration and downloading and sharing of flashcard packs

Another planned feature is the idea of sharing flashcard packs. An upload and download function would need to be created on a cloud repository with unity3D functions set to be able to load these at will.

### Assets Acquisition and Creation

### Testing and QA

# Roles

For our group we have decided not to go with individual roles, but with section the work into categories of work that need to be done. We have identified six categories that will need to be completed. The reason we have chosen this approach as opposed to singular roles is so that people may take up multiple tasks and roles if they wish to do so. Here are the categories we have chosen to go with, all work in our timeframe as well as project progress will also be separated into these roles if they are applicable.

## Admin

This will consist of all organisation on our platforms, including Trello/slack/discord/GitHub, meetings, the document report as well as all other admin related tasks. Another task falling under this category is the compiling and quality control of our reports and documents. This will also include the work on the rest of the document less related to our project, for example the team profile section on our main document.

## Programming

This will consist of all work related to programming and development and will mainly focus on the use of Unity3D to create the actual application as well as creating logs. Programming will also be responsible for documenting, screenshotting and explaining our development process so it is easy for another team to pick up our project.

## Assets acquisition/creation

This will be all asset acquisition and creation. This involves creating the flashcards, recording audio, finding background images and any other graphics/audio we might need for our project. This will also encompass the creating of a progress report for this process and have we have managed so far.

## Testing/QA

Testing will be all testing and QA on our application project, as well as the creation of reports showing testing and QA information.

## Website Creation/Updating

This will be work related to the updating of our GitHub page to reflect all the new content generated by Assignment 3.

## Presentation Video

All work related to the creation and storyboarding of our presentation video for our application.

# Scope and Limits

The general scope of this project does seem to be possible to do in the allotted time, as the project itself is reasonable for six people to create. We have, however, placed some limits on creating this application to deal with scope creep and help us meet and manage our time and work commitments on this project.

One of these limits has been the number of mini games we will produce, as it stands now, we only plan to add two. Ideally, if time and scope constraints weren’t an issue, we would want to add many more mini games to the application.

Another limit on our project is the number of flashcards we will create and supply with the application. A set of twenty is a rather modest number. Ideally, we would have many more. An ESL flashcard application such as this should have at least 200 or so cards in a stock version. We plan an enhancement where the users may create their own flashcards, but we would prefer more flashcards bundled with the stock application.

# Tools and Technologies

Our project will require a small collection of different hardware and software to achieve our goals. We have listed below the hardware and software that we need and the experience within the group using these tools.

## Software required:

* Unity3D (v2019.2.17f1) - License is free for non-commercial use and educational use. Used for creating the actual application using the Unity3D framework.
* Microsoft Visual Studio (v15.9.17)– License is free for non-commercial use and educational use. Used for the programming aspects of the application development in conjunction with Unity3D.
* Gimp (v2.10.8) – License is free for non-commercial use and educational use. Used for flashcard image creation.
* GitHub – used to help us collaborate our work and host our website under a GitHub page.
* Trello – used to help us organise our workload.
* Slack – used for all text correspondence between the team
* Discord – used for our weekly voice meetings.
* **INCOMPLETE: WHATEVER VIDEO EDITING SOFTWARE WE USE TO CREATE THE PRESENTATION VIDEO?**
* **INCOMPLETE: WHATEVER SOFTWARE WE USE TO CREATE THE WEBSITE AS WELL AS HOSTING ETC (PROBALY GITHUB?)**

## Hardware required:

* A computer capable of running Unity3D and Microsoft Visual Studio.
* A microphone for recording sound for the flashcards.
* An internet connection to allow online collaboration between the group.

## Group experience:

Jason Walstab – I have experience at a hobbyist level for Unity3D and Gimp and experience at a professional level for Microsoft Visual Studio (primarily in .NET and C#).

**INCOMPLETE: The rest? No idea you guys fill this is in if you want, maybe we should have someone have experience in video editing software?**

# Testing

Our group plans to do testing each week during development using QA (Quality Assurance) sessions. Each week a new feature will be added to the application. The programmer will initially do basic testing.

A dedicated project member will do a follow-up QA session. In this session, they will test the program and write up any errors, issues or problems they find. The programmer will then be able to follow up. If the dedicated QA tester encounters a serious problem, the programmer will schedule in another testing session before moving on to add the next applications feature.

The development team will maintain a testing log to ensure that all issues found by QA will be followed up by the programmer.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Monday** | **Tue** | **Wed** | **Thu** | **Fri** | **Sat** | **Sun** |
| **1** | Project Planning: Overview | Project Planning:  Ideas | Project Planning:  Roles | Project Planning:  Execution | Project Planning:  Planning | Project Planning:  Tools | Project Planning:  Review |
| **2** | Discord Meeting (8PM) | S: Main Menu | S: Demo Video Draft |  | E: Main Menu | S: Flashcard images set (+20 cards) | S: QA Session #1 |
| **3** | Discord Meeting (8PM) | S: QA Session #1 | E: Flashcard images set (+20 cards) | S: Flashcard Audio (2) | E: Demo Video Draft | E: Flashcard Audio (2) | S: Demo Video Final |
| **4** | Discord Meeting (8PM) | S: Flashcard Selection System | S: Basic Website Creation |  |  | E: Flashcard Selection System | S: QA Session #2 |
| **5** | Discord Meeting (8PM) | E: QA Session #2 | S: Mini-Game #1 |  |  | E: Mini-Game #1 | S: QA Session #3 |
| **6** | Discord Meeting (8PM) | E: QA Session #3 | E: Basic Website Creation | E: Demo Video Final | S: Finalise Project Report |  | E: Finalise Project Report |
|  | | | | | | | |
| **7** | Discord Meeting (8PM) | S: Mini-Game #2 |  |  |  | E: Mini-Game #2 | S: QA Session #4 |
| **8** | Discord Meeting (8PM) | E: QA Session #4 | S: Flashcard images set (+80) |  |  |  |  |
| **9** | Discord Meeting (8PM) |  |  |  | E: Flashcard images set (+80) |  |  |
| **10** | Discord Meeting (8PM) |  |  |  |  |  |  |
| **11** | Discord Meeting (8PM) |  |  |  |  |  |  |
| **12** | Discord Meeting (8PM) |  |  |  |  |  |  |
| **13** | Discord Meeting (8PM) |  |  |  |  |  |  |
| **14** | Discord Meeting (8PM) |  |  |  |  |  |  |
| **15** | Discord Meeting (8PM) |  |  |  |  |  |  |
| **Task Assignment** | | | | | | | |
|  | Everyone | Admin | Programming | Assets acquisition | Testing/QA | Presentation Video | Website |

## Week 1

## Week 2

## Week 3

## Week 4

## Week 5

## Week 6

## Week 7

## Week 8

## Week 9

## Week 10

## Week 11

## Week 12

## Week 13

## Week 14

## Week 15

# Risks

## Scope too large

There is a risk that the scope of the project is too large for us to manage. While we will diligently try to pursue this task of creating an ESL flashcard system, the project might be too big, and we will not be able to finish or meet time constraints for our project. We should try to minimise this risk by sticking to and meeting deadlines as well as following a well-organised structure that will come from good project planning and not procrastinating on tasks we individually need to complete.

## Programming issues

The programming skills for the collective group will not be enough to complete the project.

We may not have the expertise to complete essential components, such as the dynamic flashcard loading system. This case would be a critical failure, as we could not then achieve any of the functionality that relies on the essential component.

We may not have the expertise to complete additional features. Once we have completed the core functionality, we may find we cannot complete the enhancements we have documented. This case would be less critical, but the impact on the overall value of the final product may be substantial.

## Application Quality

The end product may not be of high enough quality to be palatable to potential clients. We may complete a functional app by following our plan and meeting the technical and logistical challenges but have a product that is unfavourable to the potential clients. If the final product does not present well and is easy to use, then it will face poor takeup rates with clients.

## Market Saturation

We may not get a good take-up rate in the marketplace because of the number of competing applications. The market for ESL learning tools is already quite large. Although we envision a target section within the market, our app may fail to be recognised.

## Reaching the end-users

Our application may not reach the intended users. Our intended users will primarily be non-native English speakers living in remote locations in foreign countries. We can devise an advertising strategy, but it is a complex market segment to target. Our advertising approach may not work.

## Too difficult to use

Our application may not be useful to the end user because it is not easy to use. We may find the app intuitive, but our clients are different from us and they might not feel the same way.

## System requirements too high

Our app will not be able to be run on systems our clients currently have. Our application is primarily for people in areas that lack modern technology and they may be running technology generations behind ours. We might find we are accessing technologies that their systems cannot support.

# Group processes and communications

## **//not done yet - will finish later**

## References

eLearning Industry. (2020). *10 Best Language Learning Apps For Kids - eLearning Industry*. [online] Available at: https://elearningindustry.com/10-best-language-learning-apps-for-kids [Accessed 31 Jan. 2020].

News 2020, "Vietnam lacks teachers to achieve language targets", *Thepienews.com*, 2020. [Online]. Available: <https://thepienews.com/news/vietnam-teachers-language-targets/>. [Accessed: 08- Jan- 2020].

Research, 2020, "English Language Learning Market to grow at 7.1% to hit $54.8 billion by 2025 – Insights on Recent Trends, Size, Share, Growth Opportunities, Key Developments and Future Outlook: Adroit Market Research", *GlobeNewswire News Room*, 2020. [Online]. Available: <https://www.globenewswire.com/news-release/2019/07/26/1892347/0/en/English-Language-Learning-Market-to-grow-at-7-1-to-hit-54-8-billion-by-2025-Insights-on-Recent-Trends-Size-Share-Growth-Opportunities-Key-Developments-and-Future-Outlook-Adroit-Mar.html>. [Accessed: 08- Jan- 2020].

Taylor, 2020, "People around the world still think English is the most valuable language to learn, study shows", *CNBC*, 2020. [Online]. Available: <https://www.cnbc.com/2019/06/28/people-still-think-english-is-the-most-valuable-language-to-learn.html>. [Accessed: 08- Jan- 2020].