

404 ERROR Project . . . . .	2
Client Needs . . . . .	3
UI Design . . . . .	5
Code Standard . . . . .	13
Architectural Designs . . . . .	16
User Story . . . . .	18
Deployment . . . . .	19
Sound effects & AI Voice generator . . . . .	21
Dialogue - VOICE GUIDING . . . . .	22
Game Overview: Vis-CAT . . . . .	24



## 404 ERROR Project

### Mission [↗](#)

*Not only develop a fully functional product for dot-connecting assessment that can be deployed on PC, MAC and ipad. But also make it easy to maintain and user friendly.*

### Team Members [↗](#)



Scrum Master

**Cheng Chen** [↗](#)

@Cheng Chen [↗](#)



Project Owner

**Steve Chen** [↗](#)

@STEVE CHEN [↗](#)



UI Designer

**Daxuan Yue** [↗](#)

@Daxuan Yue [↗](#)



UX Designer

**Jiaxing Fang**

@Tom



raithubaiti 

UI Designer

**Raad Althubaiti**

@Raad Khalid A Althubaiti

## ✓ Client Needs

### Functional Requirements:

- The circles have to be in an even gap
  - circles can be colorful, images, anything imaginary excellent
- **Dots need to be evenly spread and have constant shapes (such as radius and picture)**
- QR code scan
  - Before starting the test, the child needs to scan a QR code to get all students' names and IDs in that classroom and then they can select their name from a drop-down list.
- Dot interaction test:
  - There will be a fixed original picture shown on the left side of the screen and a space with 16 dots in a
  - 4X4 shape spread evenly on the right side
  - There will be 3 tasks.
    - Draw the original picture in the same way
    - Draw the vertically reversed picture
    - Draw the horizontally reversed picture
- Testing:
  - Two test choices, 1 short test, and 1 long test.
    - For the long test, there are 3 questions in total, after completing each of them, there is a choice to submit or retry.
    - For the short test, only the first question will be shown. The congratulation page will appear after finishing the 1 question no matter success or not.
  - If click submit and the answer is correct, go to the next test
    - When a player finishes the last test or fails one test, the process terminates
    - Shows the congratulation page after the child finishes the test
    - Send the collected data back to the server in a fixed form (need to communicate with the back end)
  - If they click retry, show the test again (need to confirm with client about should start at this ongoing test or should start at the first test)
    - When a player finishes the last test or fails one test, the process terminates
    - Shows the congratulation page after the child finishes the test
    - Send the collected data back to the server in a fixed form (need to communicate with the back end)
- Scoring:
  - For each test, if the child submits and succeeds on the first attempt, he gets 2 marks. If he clicks retry and succeeds on the second attempt, he only gets 1 mark from that test. If he fails the test, no mark will be given.
- Pass/Fail:
  - **Table A 4a: Minimum cut-off scores to pass the NDPA across various age groups**

Age (years)	NDPA
5	2
6	2
7	3
8	4
9	4
10	4
11	5
12	5
- Data sending
  - Send the collected data (including age, grade, score, pass/fail) together with the student's ID back to the server in a fixed form after the player completes their test (need to communicate with the back end to decide the details)

- Voice Guide
  - There should be a voice to guide the players on each step (and probably give them some positive feedback when they complete a task) as some children taking this test could be under reading age level. This voice is better to be tender and child-friendly.
- Animation
  - There should be some small animations such as the waving flower but there should not be too many of them that would interrupt the children doing the test.
  - There can be some instruction animations to show children what they need to do.

Non-functional requirements:

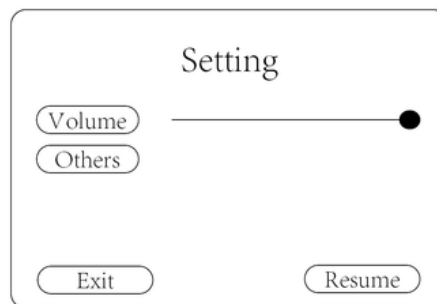
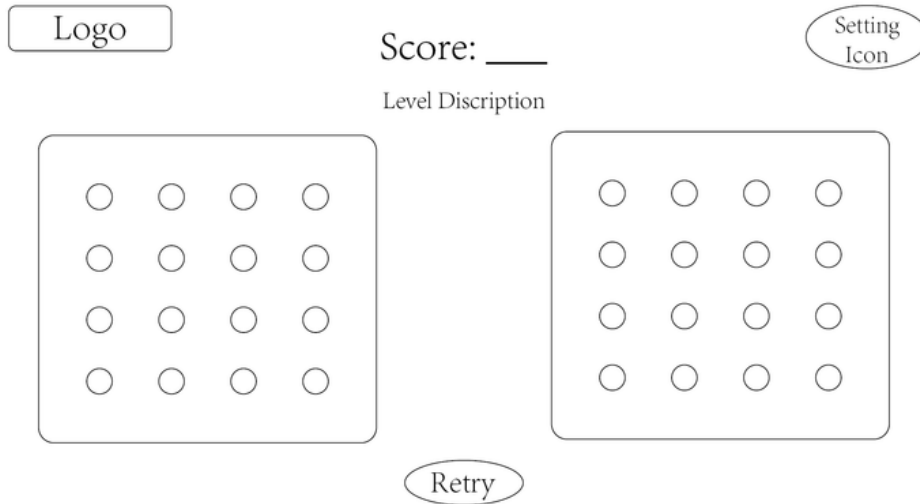
- This project will be a web app so it is easy to cross platforms

Date	Update History
31/07/2023	<ul style="list-style-type: none"> <li>• Mobile App or Web App Decision</li> <li>• Program Process Decision</li> <li>• Play Process logic decision</li> </ul>
03/08/2023	<ul style="list-style-type: none"> <li>• Detail decision for the program</li> <li>• Scoring Planing</li> <li>• How to send Data</li> <li>• Voice Guide decision</li> </ul>
01/09/2023	<ul style="list-style-type: none"> <li>• Add pass / fail table (age / NDPA)</li> <li>• Add Non-functional requirements</li> </ul>
22/09/2023	<ul style="list-style-type: none"> <li>• Add animation division</li> </ul>

## UI Design

For the initial UI template, UI Designers need to finish the following quests:

1. The style, and background of the window: cartoon-style
2. Where each UIs (picture, materials) are? Uploaded on GitHub
3. What does Logo look like? Vis-CAT
4. What is the font? Gamja Flower



Logo

Result (Fail / Success)

Score:

Some Other Texts

Exit Continue

This form is enclosed in a rounded rectangle. It contains a 'Logo' label at the top left, followed by 'Result (Fail / Success)' and 'Score:' on the next lines. Below these is 'Some Other Texts'. At the bottom, there are two buttons labeled 'Exit' and 'Continue'.

Logo

Grade: Scroll Down Choices

Level: Scroll Down Choices

Continue

This form is enclosed in a rounded rectangle. It contains a 'Logo' label at the top left. Below it are two rows: 'Grade:' followed by a 'Scroll Down Choices' button, and 'Level:' followed by another 'Scroll Down Choices' button. At the bottom center is a 'Continue' button.

Cartoon version: **(FINAL DESIGN)**

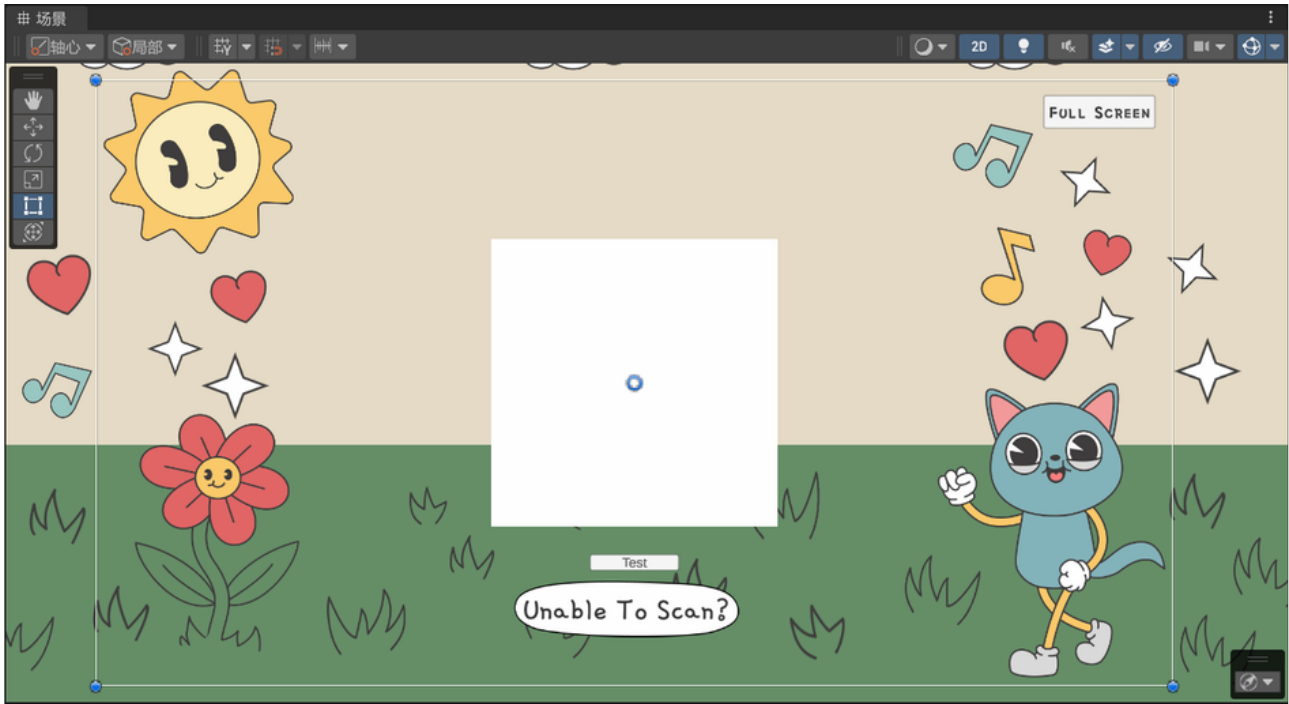
Suggested @Raad Khalid A Althubaiti



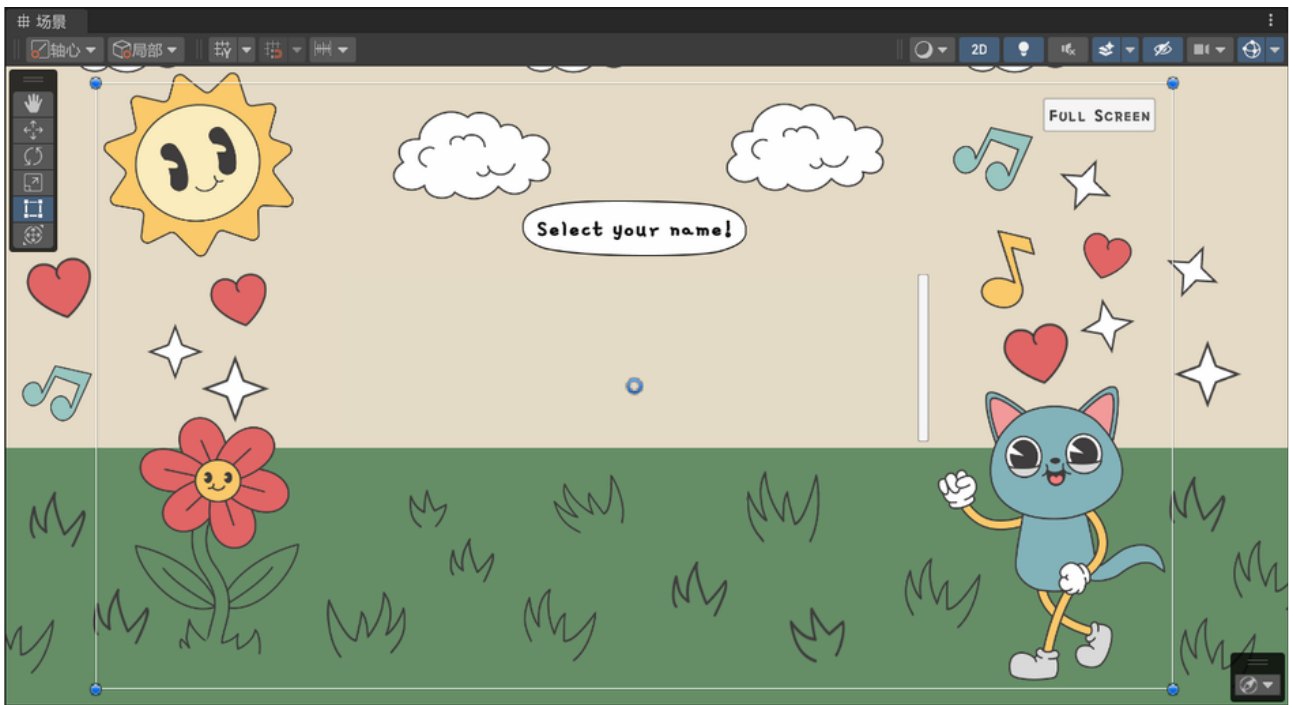
UI Designs after Sprint1 [↗](#)



Start Page

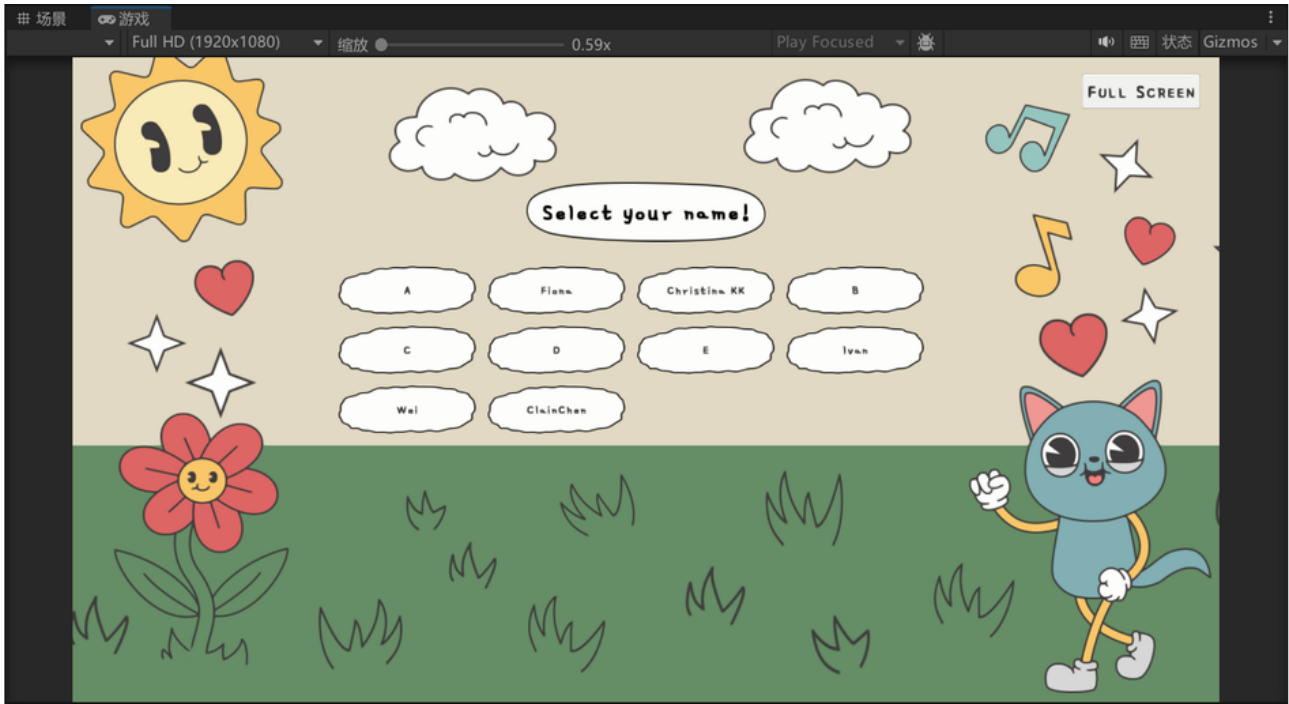


Scan Page



Select Page (Without names)

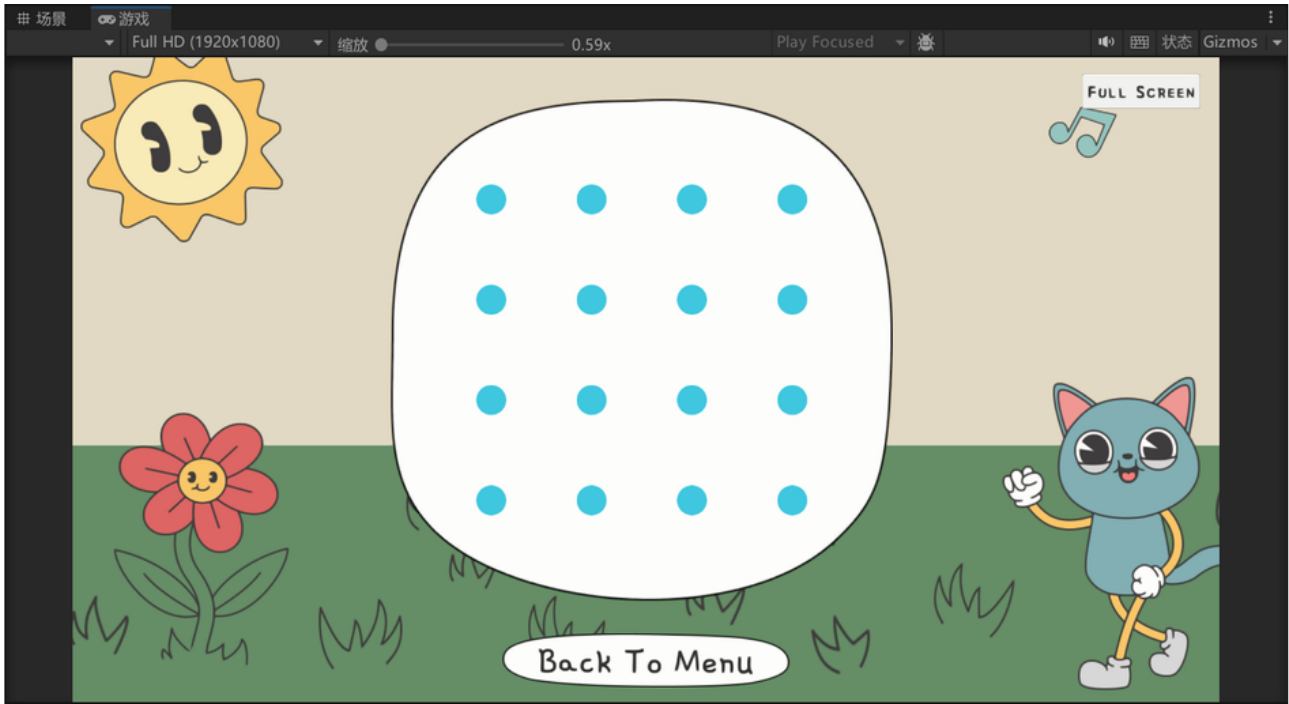




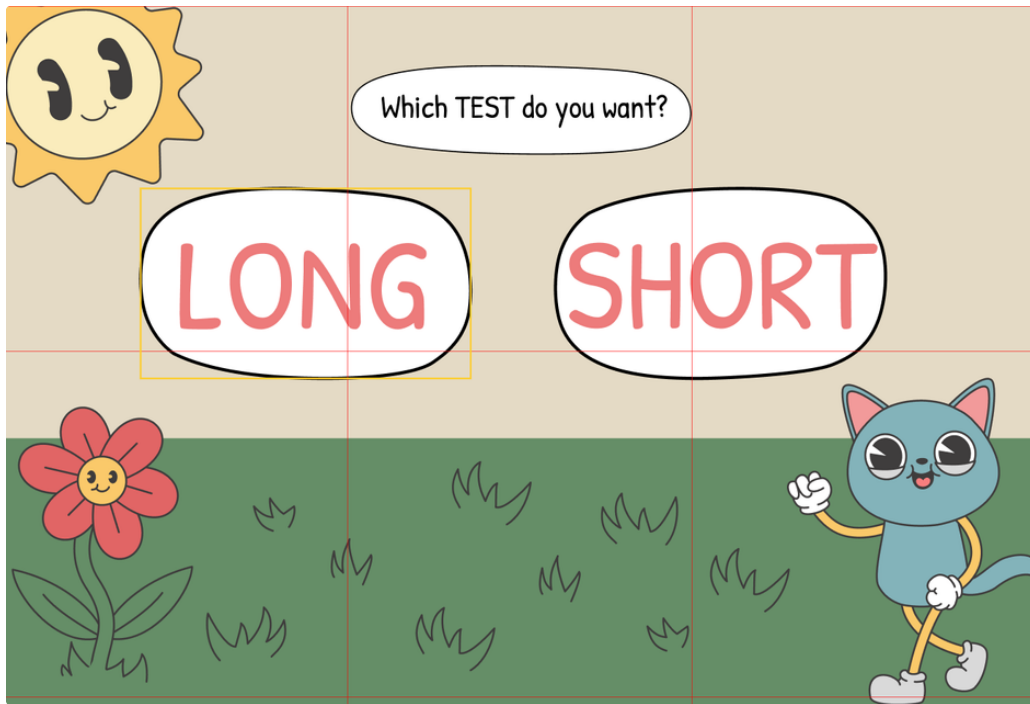
Select Page (With Names)



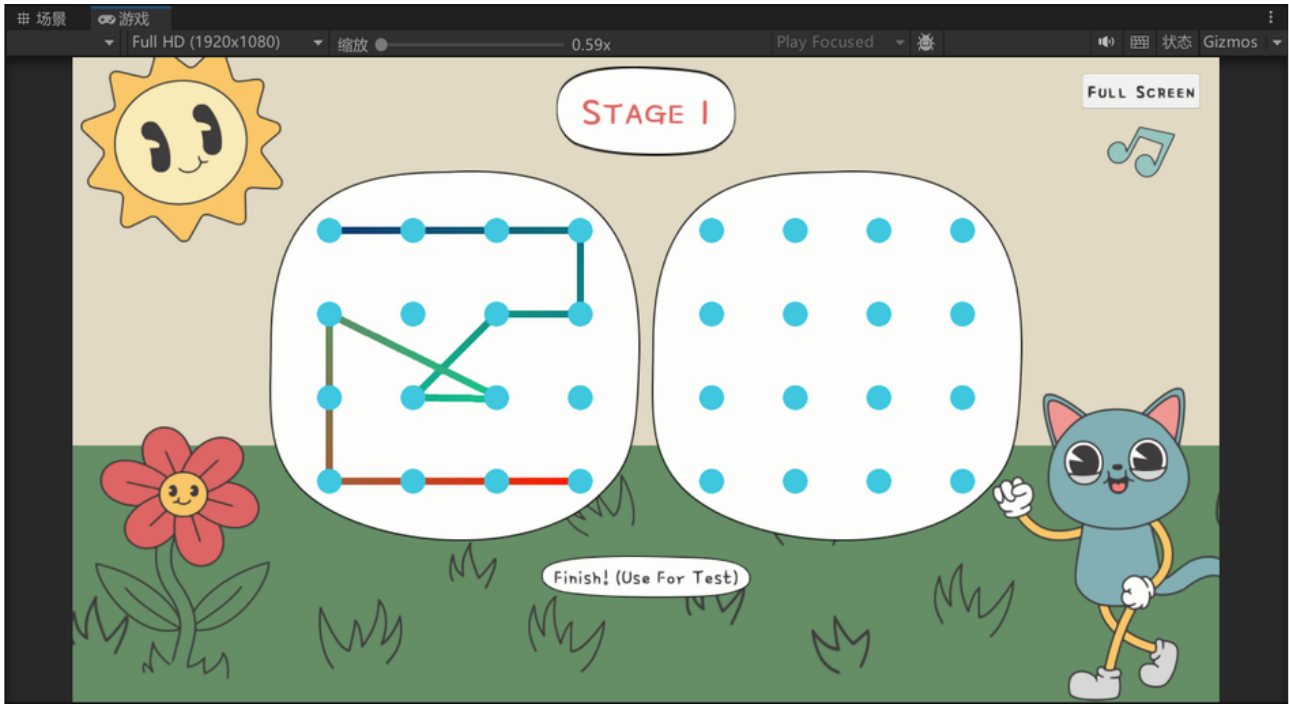
Intro Menu



Play Ground



Mode Choose Page



Actual Playing Page



Result Page

Date	Update History
14/08/2023	<ul style="list-style-type: none"> <li>Plan version 1 UI outline</li> <li>A UI Design come from Raad</li> </ul>

21/08/2023	<ul style="list-style-type: none"> <li>• More UI Designs come from Raad and Daxuan Yue</li> </ul>
01/09/2023	<ul style="list-style-type: none"> <li>• Update the old UI Designs to have a better looking</li> <li>• Add more UI Designs come from Raad and Daxuan Yue</li> </ul>
18/09/2023	<ul style="list-style-type: none"> <li>• Decide the final UI Designs and put all pages into document</li> <li>• Delete the old versions of other UI Designs</li> </ul>
22/09/2023	<ul style="list-style-type: none"> <li>• Organize the format of the document</li> <li>• Some change come from Unity</li> </ul>
18/10/2023	<ul style="list-style-type: none"> <li>• Add a page to let children decide the test mode</li> </ul>

## Code Standard

Language: C#

### Variables Naming Standard [↗](#)

- **Little hump nomenclature**, which starts from lower case word and follows with upper case first

```
1 String name = "Bob";
2 int countNumber = 8;
3 bool isMax = false;
```

- The length of the naming should not be too long, okay to use some abbreviations

```
1 // Don't do this!
2 int lengthOfDocument.....;
3
4 // Okay to do this
5 int lenghtDoc;
```

- **DO NOT** abbreviate the words that are hard to understand! If you do, please put a comment above it to describe what it is.

```
1 // Don't do this!
2 int titv;
3 // Means this is the variable
```

- The temporary variable should be as simple as possible, but it is easy to let people know their meaning

```
1 int i = 0;
2 String s = "hello";
3 int count = 0;
4
5 // Such as this circumstance
6 int a = 0;
7 while (...){
8     a += 1;
9 }
10 Debug.Log(a);
```

### Function & Class & Namespace Naming Standard [↗](#)

- **Upper hump nomenclature**, start from the upper case word and follow with the upper case first

```
1 void ThisIsTheFunction(){}
```

- **DO NOT include an abbreviation in the function name as possible as you can!**
- Enum should start from Upper letter E and follow with Upper hump nomenclature naming.

```
1 enum ESeason {Spring, Summer, Autumn, Winter};
```

- The interface should start from the Upper letter I and follow with the Upper hump nomenclature naming

```
1 interface IMyInterface{}
```

- The class should be in upper hump nomenclature naming

```
1 Class MyClass{}
```

- Struct should be in upper hump nomenclature naming

```
1 struct MyStruct{};
```

- The namespace should start from DOT and follows the direction in the Scripts folder

```
1 // The script is in directory: Asset/Scripts/Utilities/  
2 namespace DOT.Utilities{...}  
3  
4 // The script is in directory: Assest/Scripts/Line/LineBehaviour/  
5 namespace DOT.Line.LineBehaviour{...}
```

## Constants Naming Standard [↗](#)

- **Have to be complete upper size, words split by '\_'**

```
1 const String CONSTANT_STRING = "constant string";
```

## Modularization Code [↗](#)

- Put the different parts of codes that play different roles (where finish their task) into blocks by using empty lines to separate them apart

```
1 void Function1{  
2     // code block one  
3     ...  
4  
5     // code block two  
6     ...  
7 }  
8  
9 void Function2{  
10     ...  
11 }
```

## Comment Standard [↗](#)

- Comments should be placed where needed.
- All Functions (other than getter, and setter), interface, class, and beginning of the script file, should have comments to summarize what is this part of the code doing.

```
1 // Description of MyClass ...  
2 class MyClass{}
```

- The comment for utility functions should be detailed, here is the standard format

```
1 ///<summary>  
2 /// ...  
3 ///</summary>  
4 ///<returns> ... </returns>  
5 void Tool1{}
```

- The comment should be in double languages: Chinese and English

```
1 // This is the comment  
2 // 这是注释
```

```
3 void MyFunction()
```

## Others [↗](#)

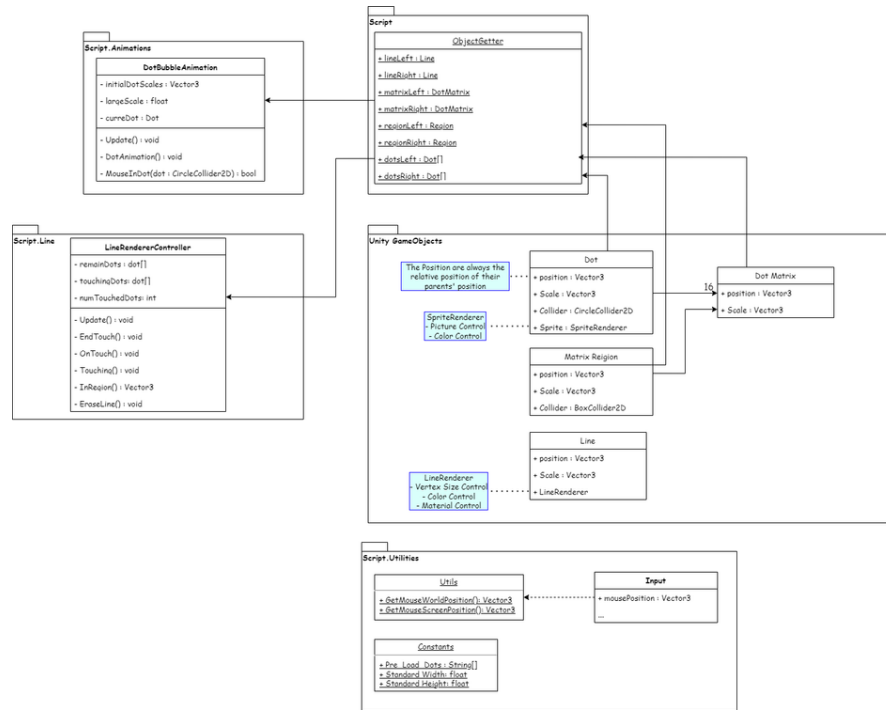
- All utils (tools) functions that will be used in different classes should be located in Utilities.

Date	Update History
14/08/2023	<ul style="list-style-type: none"><li>• Add Code Review</li></ul>

## Architectural Designs

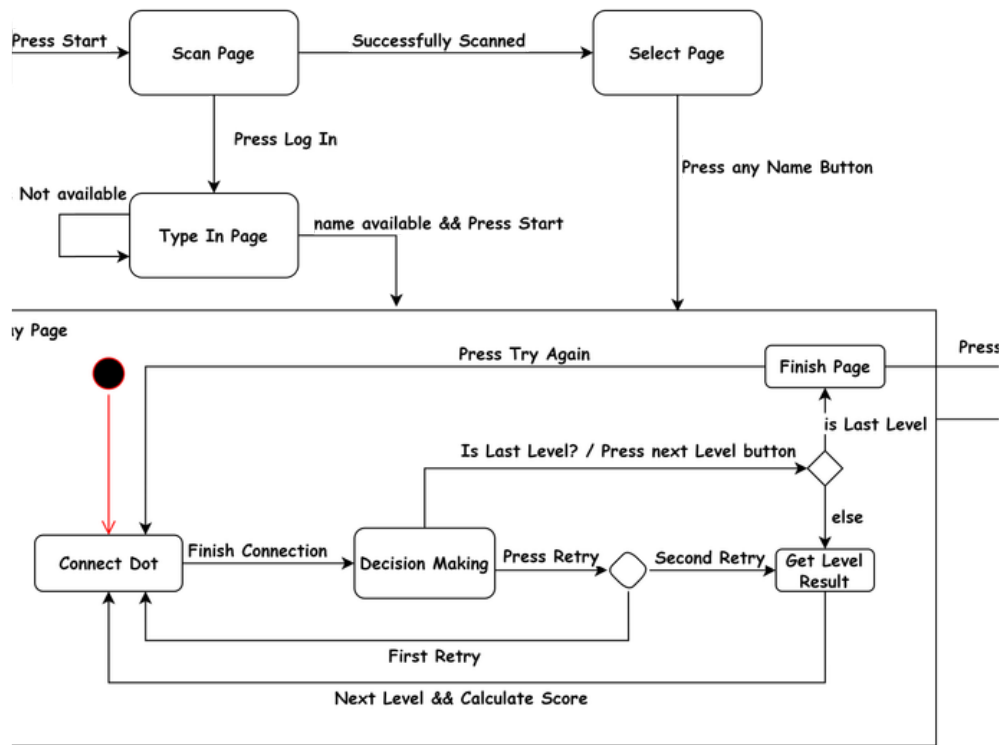
The whole software development is done in Unity, which is basically a game development engine. So the Relation between each class (object) will not only be decided by the scripts but also be decided by the game objects in the game scene.

### Domain & Design Package Model



### State Diagram of Game Processes





Date	Update History
01/09/2023	Change the title of the page
02/09/2023	Update the Domain & Design Package Model & State Diagram of Game Processes
02/09/2023	Add the notes of the page

## User Story

User	Action	Priority/Size
Children	be able to prepare by scanning a QR code and selecting their name from a list so the preparation part before the test is easy for them to do	high/high
Children	be able to do the copy, vertical flip, and horizontal flip test in a funny way	high/medium
Cristine	deploy the product on a server	high/medium
Children	be able to see how lines are connected before playing the game	medium/high
Children	be able to try the drawing before the actual test	medium/small
Children	have a chance to retry a test	medium/small
Children	proudly see the congratulatory message on the screen after finishing the test	medium/small
Cristine	the web app sends the data of children's id, score, and pass/fail back to the database after they finish their tests	medium/medium
Children	be able to listen to voice instructions and watch animation instructions	low/medium
Children	be able to click on the cat for voice instructions	low/medium
Children	be able to mute the audio when it is not needed	low/small
Children	be able to select the hardness of test	low/medium

Date	Update History
18/9/2023	Add a user story table
22/9/2023	Delete the previous User story out the table.

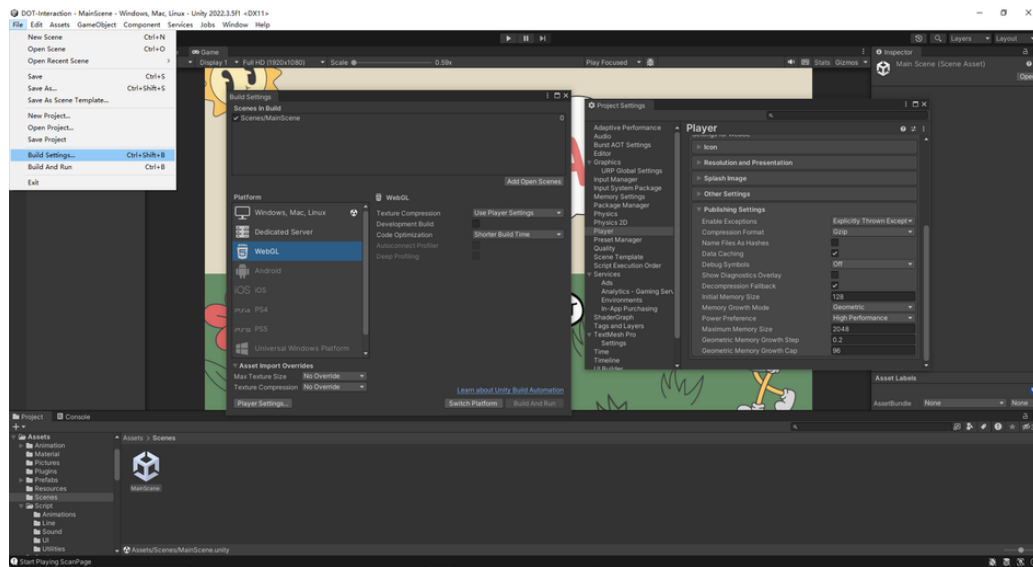
## Deployment

The public website for the project: <https://test.d1wbd8xyp2ly9d.amplifyapp.com>

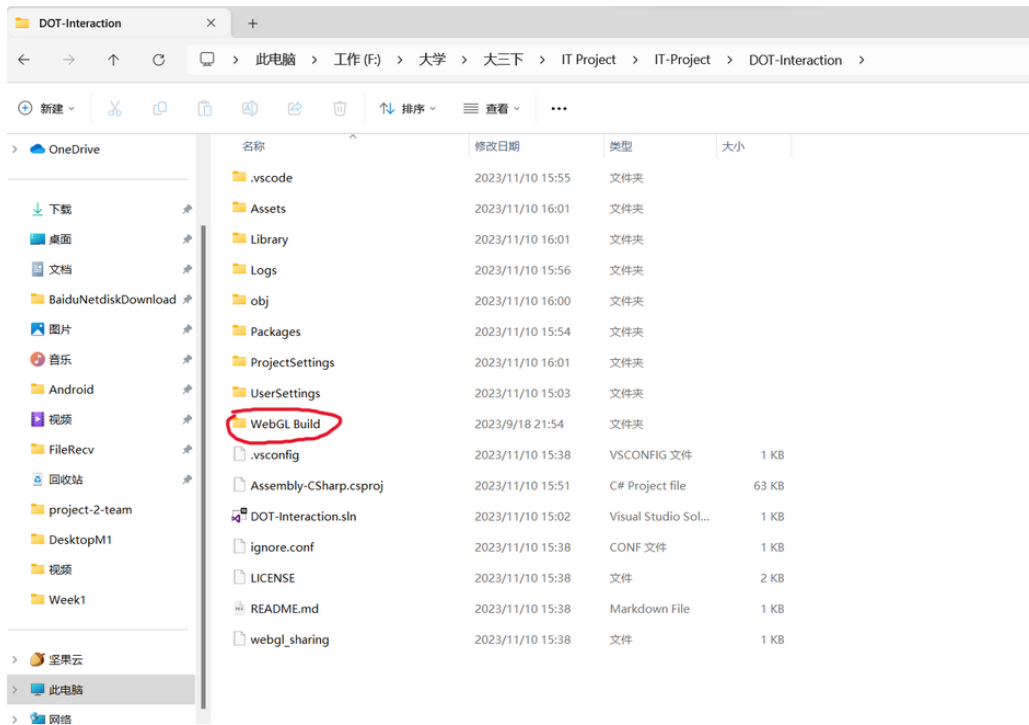
The product will be deployed through a cloud server, we choose to use the AWS(Amazon Web services) Amplify and upload the WebGL build file to the server. No FTP (file transfer protocol) platform is needed during this process as we can directly upload files to the AWS Amplify. It will be assigned to a URL so the users can access the test by entering the link. As this is a web app, it will be easy to cross different platforms.

GitHub repository: [GitHub - ClainChen/IT-Project](#)

This is a Unity project. You need to open the DOT-Interaction file inside the main repository with Unity editor version 2022.3.5f1(LTS) if want to make any changes to this program. We provide a WebGL build inside the (...) file. However, if you want to build the project again, select file->Build settings, and make sure you switch to the WebGL platform before build. The player settings are already set for you.








The WebGL build file is what you want to put in your cloud server.



Date	Update History
01/09/2023	Add the explanation of the page
15/09/2023	Add a link to the public website for the project
10/11/2023	Add the GitHub link to the code repository and introduction

## Sound effects & AI Voice generator

The Cats sounds

 Lets have some fun.mp3  You found me again.mp3  Remeber curiosity is the cat's best friend.mp3  You are the best player ever  
r.mp3  whats over there.mp3

Choosing the Easy mode and Hard mode:

 Choosing Test.mp3

Reminder of the rules of Game

 Reminder Finger.mp3


Sound effects:

 bubbles-14830.mp3

 uiclick2-79817.mp3

 level-passed-142971.mp3

 toy-button-105724.mp3

 interface-1-126517.mp3

 fantasy\_ui\_button\_6-102219.mp3

 pick-92276.mp3

 pop-up-something-160353.mp3

 stop-13692.mp3

 shooting-sound-fx-159024.mp3

AI Voice Generator

[🔗 English Text-to-speech software | Ondoku](#)

Free bubble sound effects and other kinds of sound effects.

 [Free Bubble Sound Effects Download - Pixabay](#)

Background music

 [Best SFX \(Sound Effects\) Catalog to Download | Artist](#)

Date	Update History
21/08/2023	Add bubble sound an background music choices
22/09/2023	Add specific sound effect file
18/10/2023	Add every sound effect file with labels and organized format of page.

## Dialogue - VOICE GUIDING

### **In StartPage Play :**

"Welcome to our exciting pattern-matching game Click Button to get started!"

### **In ScanPage :**

"Please scan the QR code to get your name."

### **In SelectPage play :**

"Please select your name by pressing the name button. If you cannot find your name here, you may click the "I can't find my name" button, and you can input your name by yourself! "

### **In TypeInPage play :**

"Please Type In your name, School, and other details information ..."

### **In MenuPage play :**

"Now, we are going to start the game! Are you ready? You can press "I want to try first" to get into the playground so you can freely draw the line by yourself!

### **In PlayPage play ( in Stage1) :**

It's time for a fun pattern-matching test. You'll see a set of

dots on your paper, and your job is to draw lines to connect the dots in the correct order to match the pattern shown. Take your time, and if you make a mistake, don't worry – just try again. But you only have one more chance. Good luck!

### **Enter Stage1 play :**

...Repeat the pattern shown on the left.

### **Enter Stage2 play :**

Great job! Now, we're taking our pattern-matching test to the next level. In stage 2, the pattern you see will have undergone a vertical flip. It means that if there was a dot on the left side in the original pattern, it would now be on the right side, and vice versa.

...

### **When in Stage2 play :**

... Repeat the vertical flip pattern

...

**In stage 3 play:**

We're now moving on to the third stage of our test. In this stage, the pattern you see will be a horizontal flip of the pattern you encountered in stage 1. This means that dots that were at the top in the original pattern will now be at the bottom, and dots at the bottom will be at the top. This is the final stage.

**Enter stage 3 we play:**

Repeat the horizontal flip pattern

**Finish the test play:**

Congratulations! You have done all the test!

Date	Update History
22/09/2023	Add dialog content of each stage.

---

## 👏 Game Overview: Vis-CAT

### Core Concept: [🔗](#)

This game use to test children's visual cognition and space imagination ability. Children need to copy the actual or some changing line pattern to show their abilities.

### Genre: [🔗](#)

The App should be a simple line connection application, which included the test stage and the stage that play need to play the game to test their ability. This game do not want to give children any pression and hope to let children feel enjoy when they doing the test.

### Target Audience: [🔗](#)

The students between 5 - 12 years old, and needs teacher to play with them.

### Progress [🔗](#)

### Control: [🔗](#)

The game is facing to Tablet users, so basically using the touch-screen pattern, but also designed to able to run in computer by using mouse.

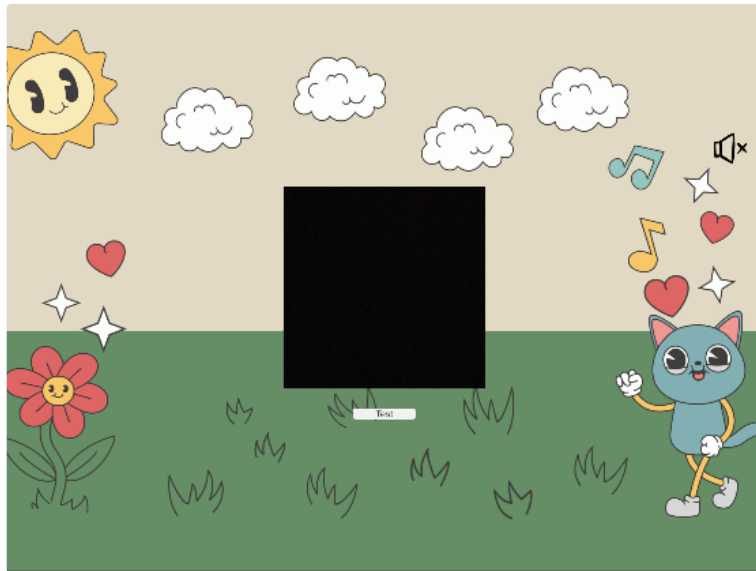
### Pages: [🔗](#)

Start page:

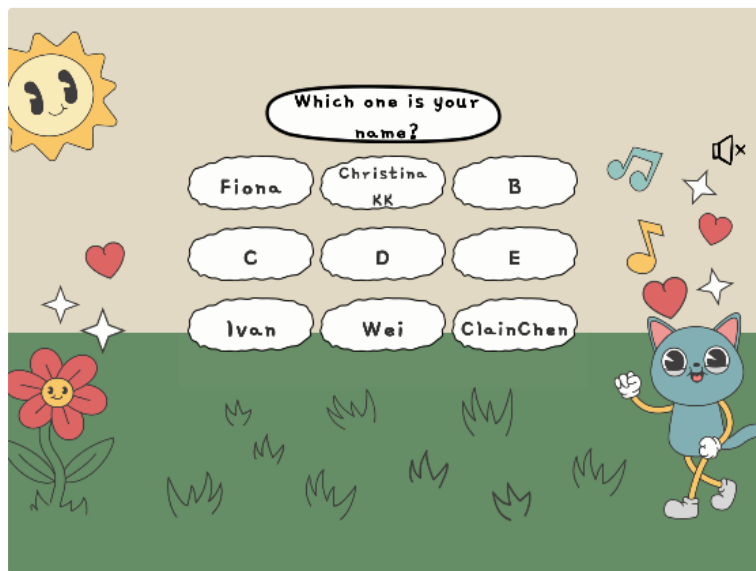


QR code scan page:





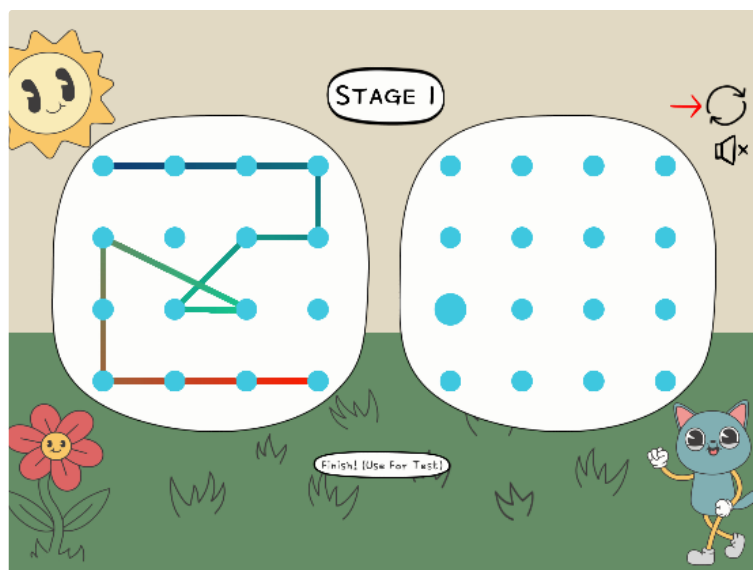
Name selection page:



Menu page:



Game page:



Finish page:



### Gameplay Mechanics: [↗](#)

This will be a 2D flat plane game just like a normal browser, which run like Kahoot. This app will fulfill the following functions:

- Player is able to interact with the buttons and buttons are able to do their works
- Player is able to play freely in Try Stage.
- Player is able to experience the normal gaming process in Play Stage
- Player has only one retry chance in each level
- Player will directly finish the game after they fulfill the condition
- Player are able to do the interaction with vis-cat
- The APP is able to scan the QR Code and receive the data in QR Code
- The APP is able to send the final result of player to the backend

### Game Design [↗](#)

#### Art and Audio: [↗](#)

##### Art Style: [↗](#)

Responsible by Raad and Marco

The art style should be fully cartoon, and create a atmosphere with please and funny. The element in background should be simple to avoid distract children's attention. The game will include some easy animations, such as the cat is able to wave it's arm, the sun is able to rotate, and the cloud can move around, etc.

##### Sound & Music: [↗](#)

Responsible by Tom

As customer's requirement, the game do not have background music, but still has the sound effect when play connecting lines.

##### Assets: [↗](#)

To mask the game as simple as possible, we will make the background of game to pure color with different level, and add more elements gradually. We found dialog box, cloud, sun, cat, music score, stars, etc. Some simple icons, use for setting and page changes. All assets and fonts are found in the free website, and input them in Unity for further creation.

The animation will entirely made in Unity, which using Unity's Animator and Animation Controller Components.

### User Interface (UI): [↗](#)

The UI was quite brief, the button is round side with white background, and the dot matrixes are using the default material in Unity.

### References:

UI:

[🎨 Retro cartoon animation | Figma Community](#)

[🎨 Doodle icons | Figma Community](#)

[🎨 Comics in Figma | Figma Community](#)

Fonts:

[🎨 Gamja Flower - Google Fonts](#)

AI Voice:

[↗ English Text-to-speech software | Ondoku](#)

### Technology and Tools: [↗](#)

Development Software : Unity

UI Design Software : Figma

Animation Design: Adobe Effect

AI Voice Creator: [↗ English Text-to-speech software | Ondoku](#)

Sound: StudioOne5

### Team Communication, Timelines and Task Assignment: [↗](#)

Scrum Master: Clain Chen

Project Owner: Steve Chen

Programmer & Animation Creator: Clain Chen

API Implementation: Steve Chen

UI Design In Figma & Animation Design: Raad, Marco Yue

UX Design: Tom Fang

Date	Update History
10/10/2023	Add this page to the confluence
10/11/2023	Update this page to match the most recent readme file in GitHub