

Documentation

Floating Words:

Floating Words is an AR translation app for Android. It allows users to point their smartphone camera at a real-world object, and the app automatically creates a 3D AR anchor at that position with the translated label of the object. This vocabulary is stored in a permanent dictionary that can be accessed for detailed information about each word. In addition, the app provides a "Flashcards" feature for revising and studying newly added vocabs. Finally, a screenshot of the real object is stored for each word, so that the user has a visual reference to the vocab in question.

Requirements:

Unity Engine: Floating Words was developed on the Unity Engine, which, in addition to the AR Foundation for creating augmented reality content, supports a number of other handy APIs for creating user interfaces and builds for Android and iOS. We chose Unity Engine version 2021.3.13f1, which was the latest stable release at the beginning of the development phase. You will also need to download the build support for Android platforms in the Unity Hub.

Android: All users who want to install and use this application must have an Android phone with the version: Android 7.0 'Nougat' API level 24 or higher. Other devices with iOS or Windows as operating systems are not supported by Floating Words at this time.

Internet connection: Furthermore, the software must call the Google Cloud Vision API for recognizing image features and the Google Translation API for translating words from the source language into the target language. So a good Internet connection is also a prerequisite for a comprehensive user experience.

Lighting conditions: Since the Vision API is solely reliant on RGB data from the smartphone camera, good lighting conditions are essential for the object recognition algorithm to produce accurate and precise results.

How to get started:

Setup Google Cloud Vision API:

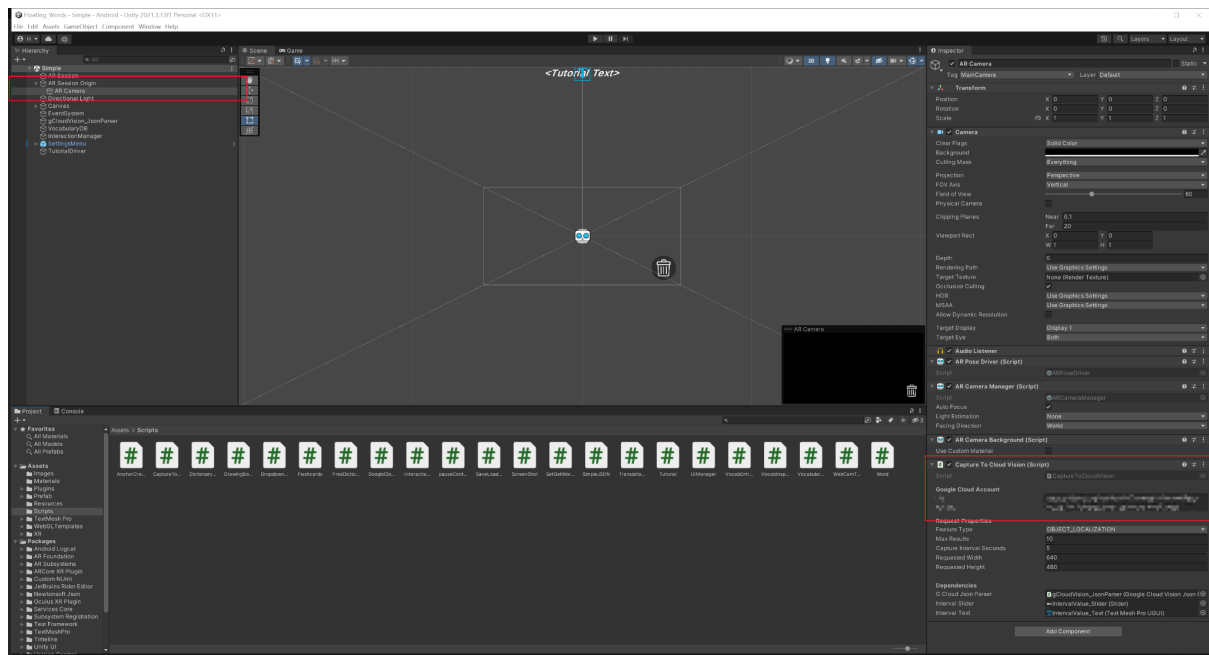
To use the Vision API, you need to create an API key in your Google Cloud account. To do this, you need to enter your credit card information at Google Cloud. This will automatically grant you a free \$300 credit that you can use for the machine learning API. ([Setup](#))

Create a service account and download the private key file

1. In the Google Cloud console, go to the **Create service account** page.
[Go to Create service account](#)
2. Select your project.
3. In the **Service account name** field, enter a name. The Google Cloud console fills in the **Service account ID** field based on this name.
In the **Service account description** field, enter a description. For example, Service account for quickstart.
4. Click **Create and continue**.
5. To provide access to your project, grant the following role(s) to your service account: **Project > Owner**.
In the **Select a role** list, select a role.
For additional roles, click add **Add another role** and add each additional role.
Note: The **Role** field affects which resources your service account can access in your project. You can revoke these roles or grant additional roles later. In production environments, do not grant the Owner, Editor, or Viewer roles. Instead, grant a [predefined role](#) or [custom role](#) that meets your needs.
6. Click **Continue**.
7. Click **Done** to finish creating the service account.
Do not close your browser window. You will use it in the next step.

Create a service account key:

- In the Google Cloud console, click the email address for the service account that you created.
- Click **Keys**.
- Click **Add key**, and then click **Create new key**.
- Click **Create**. A JSON key file is downloaded to your computer.
- Click **Close**.
- Enter the API key in Unity GameObject "AR Camera" (AR Session Origin/AR Camera/CaptureToCloudVision.cs)



Set Vision API key in Unity

How to set up Google Cloud Translation API:

The Google Cloud Translation API can be used without an account or API key as long as a limited number of requests are made to the API. To unlock even more requests, you can create an account. For this app, the version without an account has proven to be sufficient, so you don't need to do anything for it.

Unity:

If the Google Cloud Vision API key is referenced in the CaptureToCloudVision component, you can now run the application in the editor. For the Japanese language to work you will need to unzip the file “Assets/Textmesh Pro/Resources/Fonts & Materials/NotoSansJP-Regular SDF.asset” and add it as a fallback font to the default font “LiberationSans SDF”.

Note: The camera will not open on PC since AR Foundation requires AR Core for the camera feed. To see the camera feed and test the AR functionalities you need to build the APK and run it on your Android device.

Setup Google APIs into Unity:

This was not so straightforward as directly importing the ready-made GIT project into Floating Words. In the Git project that we found and used as the base of implementation. There are no proper features that meet our project requirements. In this case we took a research on the official website and figured out that Cloud API provides a feature type called “OBJECT_LOCALIZATION”. And then we changed the type parameter in the input JSON file and sent it to the Cloud API. Additionally, we need to draw bounding boxes to show the objects on the screen. The coordinates of the bounding boxes are from the API response which is of JSON format with messages that include objects’ “name”, “BoundingPoly” and

the “score” indicating the possibility between the ground truth label and output label. So in this case we implement a class called “JsonParser” to extract the information from the response text and another class to draw the bounding box using OpenGL.

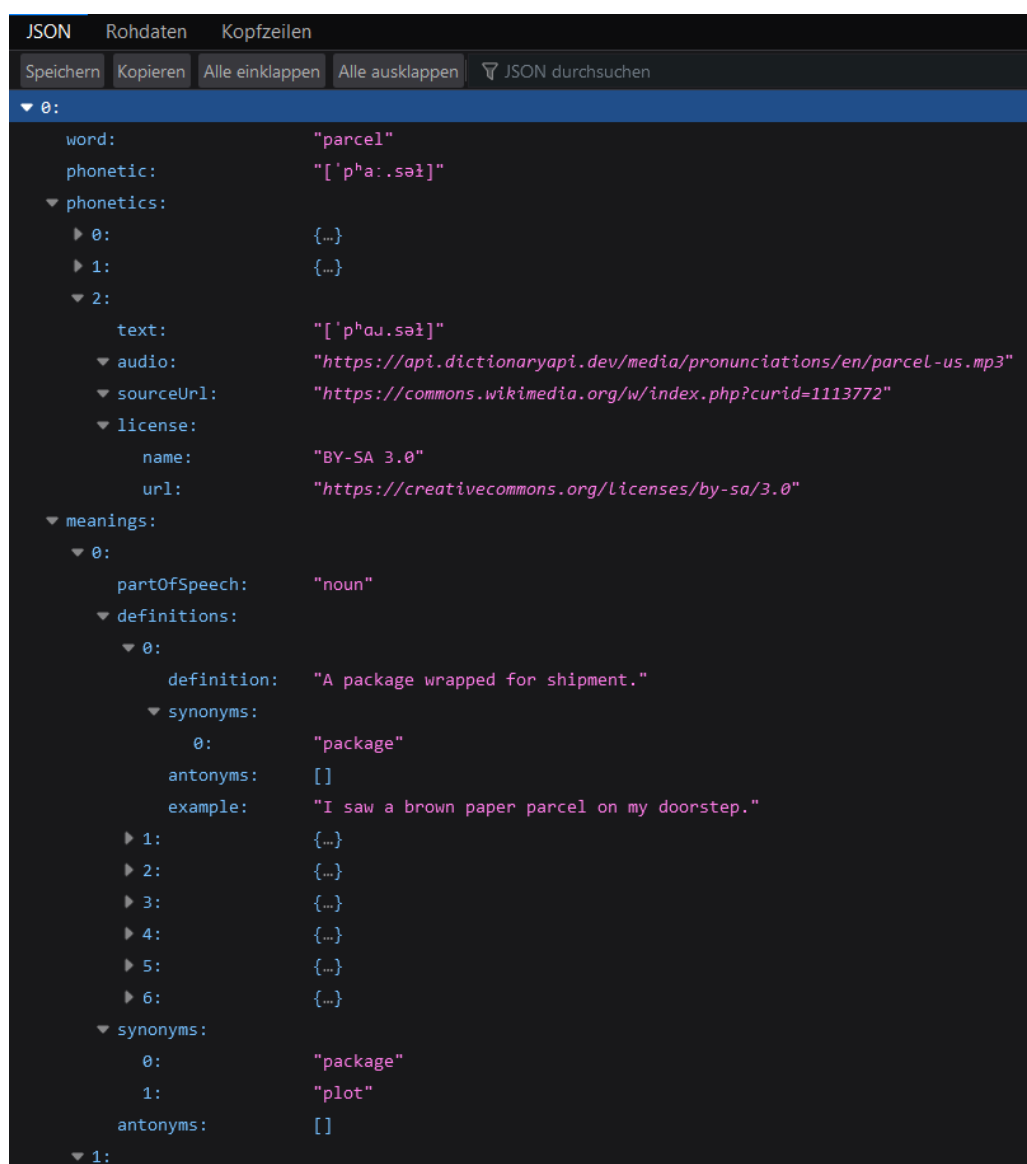
Camera orientation problem: The image screenshot on the phone is a part of the input that we should send to the Google Cloud API. However the way that system saves the image and the picture is displayed are different. We should vertically flip the picture manually to correct the mirror flip wrong orientation that is intended by the system.

Challenges with Google Cloud API: The model that Google Cloud API used for object localization and detection is not so optimized. Some objects or shapes are very prominent, e.g. Person, packaged goods, etc. Which means lots of objects will be categorized as a general class as packaged goods even if the objects in the bounding box are common enough. But still it is hard to make the behavior better since besides the default model provided by Google Cloud API, we can not find other models that are used for object classification and localization.

Components:

Free Dictionary API:

In addition to the Google Translation API we use the free and open source dictionary API [“Free Dictionary API”](#). Google's Translation API is only used for translating the English word into multiple languages but does not provide more information about a word. For this, we use the dictionary API to fetch additional information about a vocab from the internet by creating a web request. This includes details like the definition, part of speech (noun, vocab, adjective), synonyms, example sentences, etc. For some words, it will also provide a URL to an audio clip of the English pronunciation of the word. The API does not require an account or key to access this information.



```
JSON Rohdaten Kopfzeilen
Speichern Kopieren Alle einklappen Alle ausklappen 🔍 JSON durchsuchen

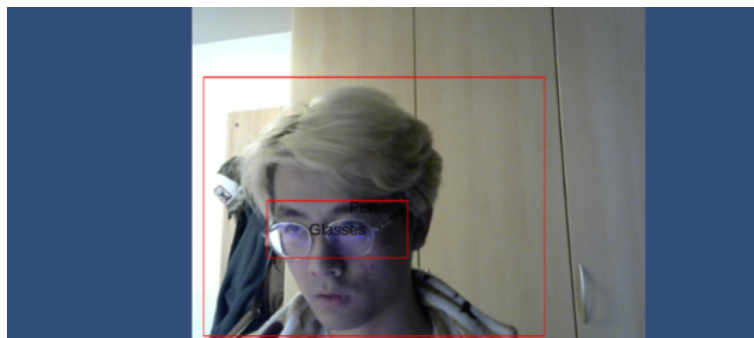
▼ 0:
  word: "parcel"
  phonetic: ["'p'hɑ:.səl"]
  ▼ phonetics:
    ▶ 0: {...}
    ▶ 1: {...}
    ▼ 2:
      text: ["'p'hɑu.səl"]
      ▼ audio: "https://api.dictionaryapi.dev/media/pronunciations/en/parcel-us.mp3"
      ▼ sourceUrl: "https://commons.wikimedia.org/w/index.php?curid=1113772"
      ▼ license:
        name: "BY-SA 3.0"
        url: "https://creativecommons.org/licenses/by-sa/3.0"
  ▼ meanings:
    ▼ 0:
      partOfSpeech: "noun"
      ▼ definitions:
        ▼ 0:
          definition: "A package wrapped for shipment."
          ▼ synonyms:
            0: "package"
            antonyms: []
          example: "I saw a brown paper parcel on my doorstep."
          ▶ 1: {...}
          ▶ 2: {...}
          ▶ 3: {...}
          ▶ 4: {...}
          ▶ 5: {...}
          ▶ 6: {...}
        ▼ synonyms:
          0: "package"
          1: "plot"
          antonyms: []
      ▼ 1:
```

Sample JSON response for the word “parcel”

AR Anchors :

To avoid having to permanently relabel the current camera view and the objects it contains every time an object is outside of the camera view, we remember the world position of each identified object using a 3D anchor. This allows us to greatly reduce the number of API requests (which are limited in a free Google Cloud account) and still identify a large number of objects simultaneously.

As the GCloud Vision API returns a 2D bounding box of the labeled object we use this information to calculate its center and send a ray from this xy-position in camera space into the 3D world space. When this ray hits a feature point or surface polygon we attach an anchor to its xyz-position in the 3D world.



2D Label



3D anchor

Logcat :

Normally, without Logcat, developers cannot detect errors and see debug log messages when the project is built and run on mobile devices. There is nothing that tells the developer where the errors are or why the software crashes. In this case, we recommend Logcat to those developing software for mobile phones as a target platform. You can find it in the package manager and install it with a single click on the "Install" button.

023/01/13 08:45:17.412	19133	19320	Error	Unity	System.NullReferenceException: Object reference not set to an instance of an object.
023/01/13 08:45:17.769	19133	19320	Error	Unity	at InteractionManager.Update () [0x00000] in <00000000000000000000000000000000>.o
023/01/13 08:45:17.769	19133	19320	Error	Unity	System.NullReferenceException: Object reference not set to an instance of an object.
023/01/13 08:45:17.789	19133	19320	Error	Unity	at InteractionManager.Update () [0x00000] in <00000000000000000000000000000000>.o
023/01/13 08:45:17.789	19133	19320	Error	Unity	

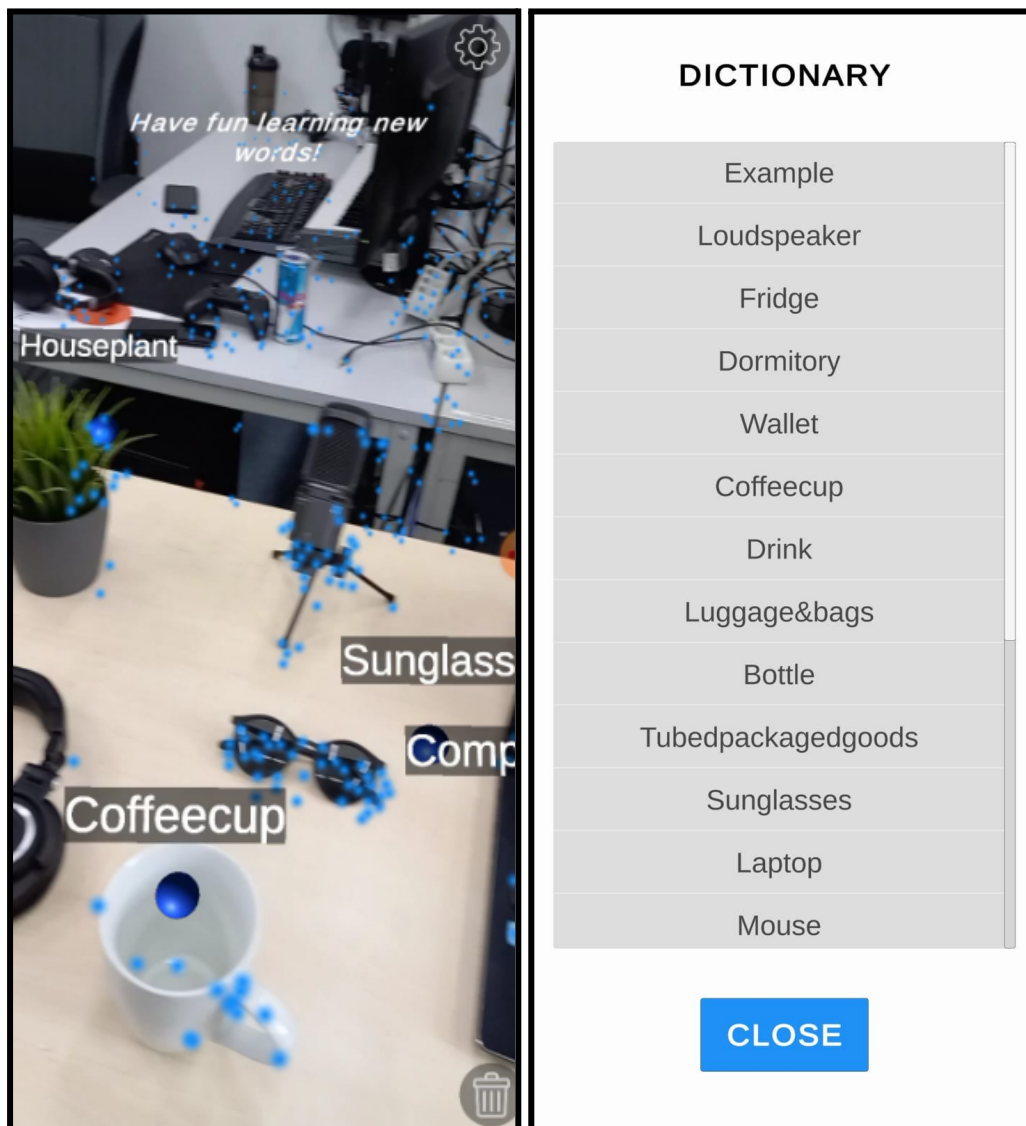
Logcat output

UI components:

In the following the key UI components and their functionalities are briefly outlined.

Main Scene:

In this scene, users see their AR camera feed. Here all the objects which were detected by the Vision API get a label that is attached to a 3D anchor. The AR Foundation anchors are attached to “feature points” (visualized as blue dots) and “planes” (not visualized). Note: For the anchoring to work the system first needs to identify feature points and planes to attach the anchors. These are detected when the user moves his camera slowly around the objects to be detected.

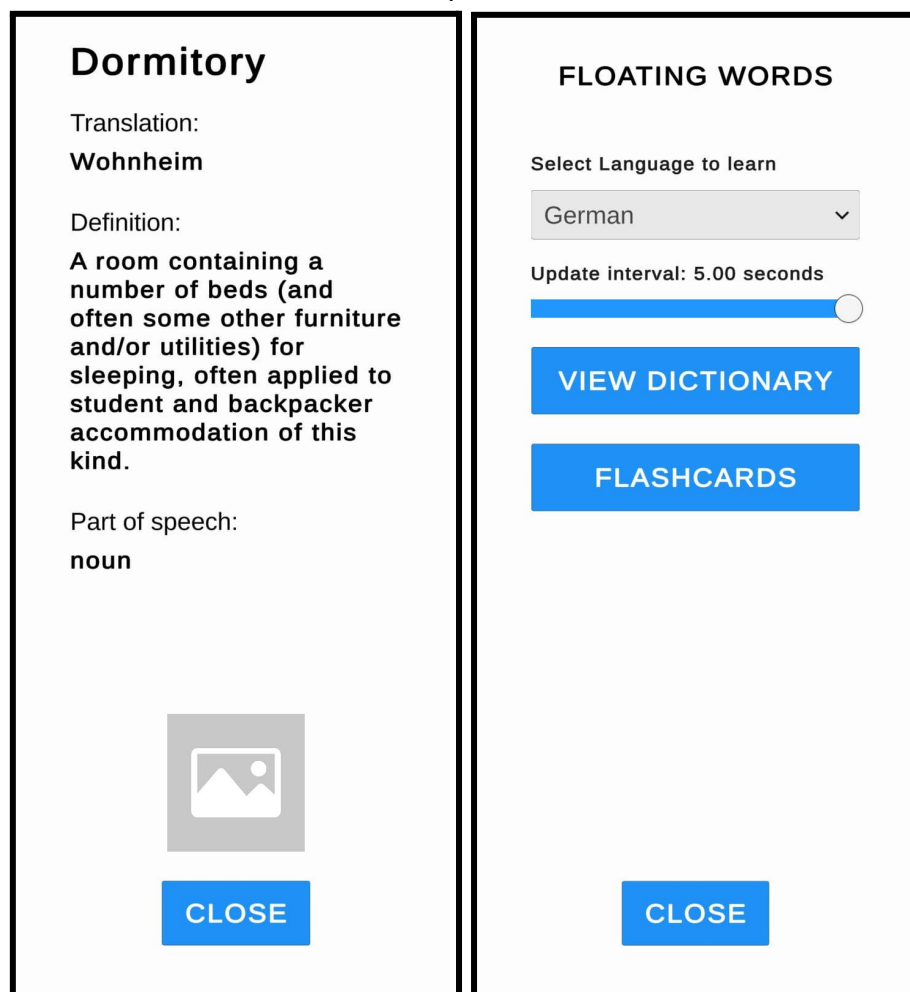


Dictionary UI:

The *Dictionary UI* window consists of a scrollable view that lists all the vocabs contained in the dictionary database as clickable buttons. They are automatically updated when the window is opened and can be clicked on to open the *Vocab Inspector UI* described in the next part.

Vocab Inspector UI:

The *Vocab Inspector UI* functions as an inspector window for a specific word. It is opened by clicking on a vocab entry in the *Dictionary UI* and reveals additional information about the vocab stored, such as the translation (which depends on the currently selected language), definition, or part of speech. In addition, it includes an interactable speaker icon that when clicked on plays an audio clip of the English vocab. This button will only appear if the active word contains an audioClipURL (The Free Dictionary API does not provide an audio clip for every English word). For now, the audio clip is always the English pronunciation but should be changed to the respective pronunciation of the translated word in later works. Lastly, the window features a screenshot button which opens the *Screenshot UI* window.



Screenshot UI:

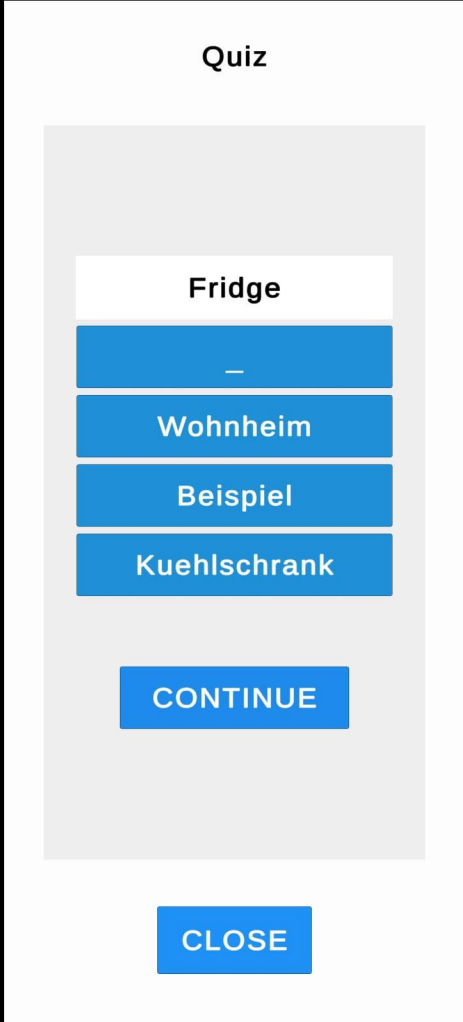
This window will load and display an image (screenshot) of the real word object linked to the active vocab. This screenshot is captured once whenever a new word is added to the dictionary database. Screenshots are stored persistently in the application's data path.

Update Interval Slider:

The main settings menu includes a slider to adjust the update rate of the Google Cloud Vision API requests. When the value is decreased (lower interval) the program will send more frequent web requests with the current camera image to detect object labels. In essence, this means lowering the value leads to better real-time object detection but can slow down the application due to the increased amount of web requests, detected objects, and created AR anchors. The slider's value ranges from 1 to 5 seconds.

Flashcards UI:

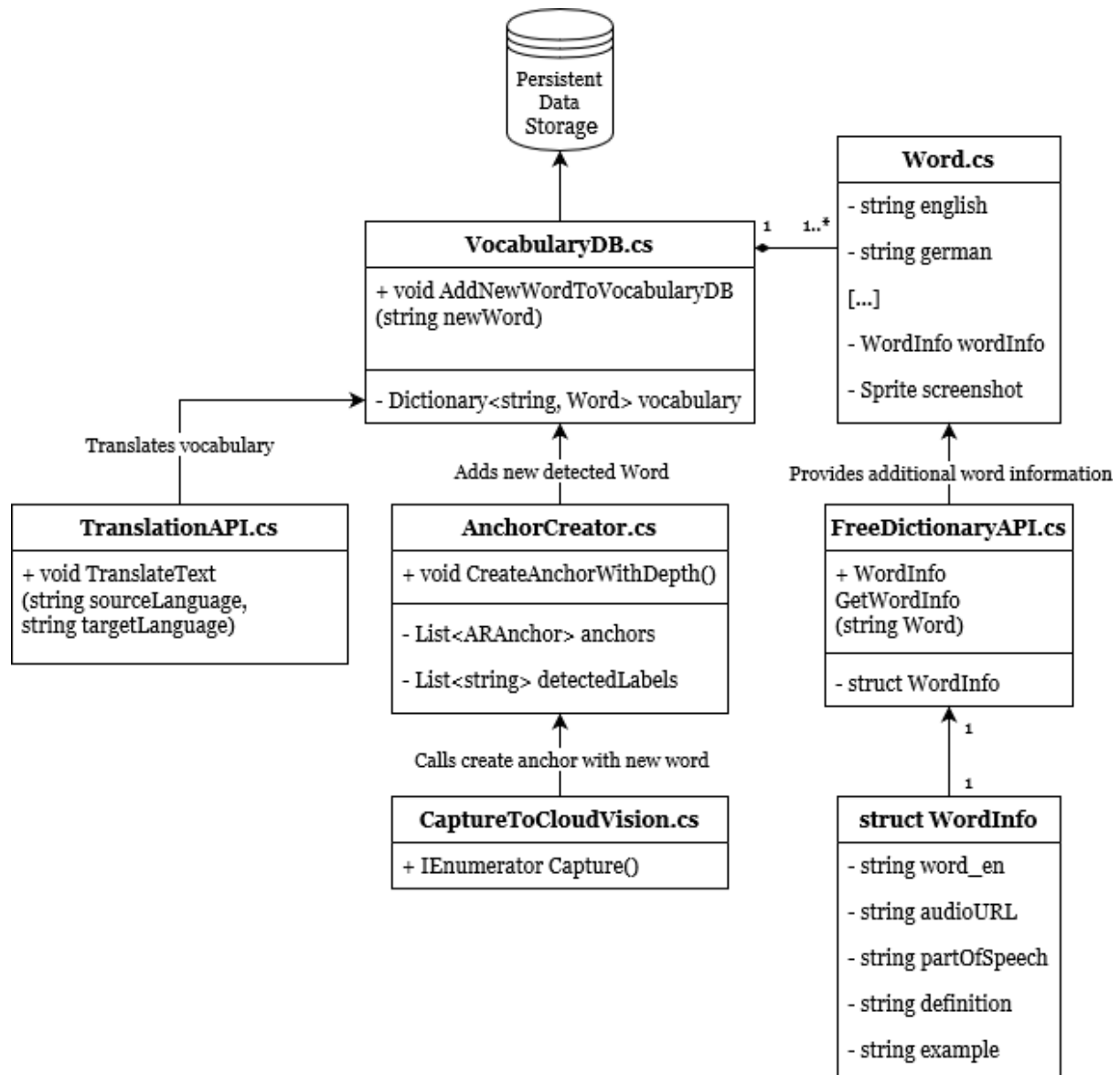
The flashcards feature can be accessed by clicking on the respective button in the main settings menu. This view will display a random vocabulary from the dictionary and ask the user for the correct translation of the word by providing four optional answers gathered from other existing words.



The image shows a mobile application interface for a flashcard quiz. At the top, the word "Quiz" is displayed. Below it, the word "Fridge" is shown in a white box. Underneath "Fridge" are four blue buttons with white text: a hyphen "-", "Wohnheim", "Beispiel", and "Kuehlschrank". Below these buttons is a blue button with the text "CONTINUE". At the bottom of the screen is a blue button with the text "CLOSE".

Architecture:

The following diagram displays the key components of our architecture:



Important Links:

- [Floating Words GitHub page](#)
- [Google Cloud Vision API](#)
- [Google Translation API](#)
- [Free Dictionary API](#)
- [AR Foundation](#)

Doxygen:

We use Doxygen to generate documentation for our project, the configuration file is located in /Documents/Doxyfile, every scripts we wrote have been commented properly, you can open the HTML Doxygen file from /Documents/html/annotated.html

Floating Words

Main Page Packages Classes Search

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

[detail level: 1 2 3 4 5]

SimpleJSON	
JSONArray	
JSONClass	
JSONData	
JSONNode	
UnityEngine	
XR	
ARFoundation	
Samples	
AnchorCreator	Manages creating and removing anchors
CaptureToCloudVision	Class with functionalities to send Google Cloud Vision API requests using a REST API and the Unity Render Texture as input image. See: Google Cloud Vision API RESTSee: Unity Render Texture
AnnotateImageRequest	
AnnotateImageRequests	
Feature	
Image	
DictionaryUI	Manages the Dictionary UI components
DrawingBoundingBox	Draws bounding boxes
DropdownHandler	Manages the language dropdown
Flashcards	Controls the Flashcard/Quiz window and provides functionalities for creating question and answer sets from the vocabs in the providedictionary
FreeDictionaryAPI	Fetches additional information to a given english word from the Free Dictionary APIand returns it in a WordInfo struct
WordInfo	
GoogleCloudVisionJsonParser	Extracts the label and bounding box of a gcloud vision REST response
DetectedObjBoundingBox	
InteractionManager	
PauseController	Pausing and resuming the game by modifying the Time.timeScale
SaveLoadController	
SceneFader	Functions to custom a scenebut not only it is the application data path. Load it only with the referenced scene as key.