

# EYELEARN – TECHNICAL MANUAL

## PROJECT TITLE

EyeLearn – A Machine Learning app for learning the basics of a new language.

This app will use Machine Learning techniques such as computer vision, speech recognition and text-to-speech to allow students to improve their language skills.

## STUDENT DETAILS

NAME: CATHAL HUGHES

STUDENT NUMBER: 15417922

PROGRAMME: CASE4

MODULE CODE: CA400

SUBMISSION DATE: 19/05/2019

MODULE COORDINATOR: PAUL CLARKE

SUPERVISOR: DR. CATHAL GURRIN

## ABSTRACT

EyeLearn is a Machine Learning tool in the form of a web application which allows users to learn the basics of a new language using machine learning techniques. It makes the learning of a new language fun and interactive and allows the student and teacher to keep track of a student's learning and progression.

The platform is built as a web application and is device agnostic i.e. it can be accessed from any device whether it is mobile, desktop or tablet. It was built using Python Flask and uses multiple reusable and distributed microservices. The main web application and microservices are all hosted by Google Cloud.

The application allows students to engage with multiple activities that will improve their language skills. As well as this there are tools which allow users to translate the world around them using their device camera, translate their speech and Parts-Of-Speech (POS) Tag sentences. Users of this application can join a class which enables the app to be used in a classroom setting. This enables teachers to track the

progression of their class to see who is struggling and who isn't. The user is also able to track their own performance, and this shown through a series of visualizations.

This Project also examines the benefits of using technology as a learning support tool in the educational system.

## TABLE OF CONTENTS

Project Title .....	1
student details .....	1
Name: Cathal Hughes.....	1
Student Number: 15417922 .....	1
Programme: case4 .....	1
Module Code: ca400 .....	1
Submission Date: 19/05/2019 .....	1
Module Coordinator: Paul Clarke .....	1
Supervisor: Dr. Cathal Gurrin .....	1
Abstract.....	1
1. Overview .....	4
1.1 Platform development Motivation .....	4
1.2 Research Motivations .....	4
Introduction of Ipads in Schools .....	4
Learning a Language Using Technology .....	5
1.3 Glossary.....	5
1.4 Conducted Research .....	5
Evidence of the Benefit of iPads in School.....	5
Evidence of the Benefits of using technology to learn a language .....	6
Choosing IMage Classifiers.....	6
Microservices .....	7
Learning Python Flask .....	8
Machine learning Research.....	8
Data ANALYTICS .....	10
2. Design.....	10
2.1 Model View Controller .....	10

2.2 Microservices .....	12
2.3 Architectural Overview Diagram.....	16
2.4 Low Level Class (Package Diagram).....	17
2.5 ER Diagrams .....	18
2.6 User Interface Design – Schneiderman’s Rules .....	19
3. Implementataion .....	20
3.1 Python Flask Web Application .....	20
Image Based Activity Views .....	20
Analytics View .....	22
Handwriting/Doodle Views .....	23
Speech Recognition Views .....	24
Phrase Translator & POS Tagger .....	25
Tiles Views.....	26
Doodle Tracking Views.....	27
Printability.....	28
3.2 Python Flask Microservices.....	29
3.3 Deployment.....	35
4. Problems Solved.....	38
5. Results.....	42
5.1 Research Conclusions.....	42
Machine Learning Can Be used as an Aid for learning a new language .....	42
6. Future Work .....	43
Machine Learning as a tool in Foreign Language Learning.....	43
Microservice Reuse .....	44
Improving Model Accuracy And Removing All bACKEND Machine Learning MoDELS .....	44
Using Irish.....	44
7. Appendices.....	44
7.1 Reference List.....	44
7.2 Useful Links .....	46

## 1. OVERVIEW

### 1.1 PLATFORM DEVELOPMENT MOTIVATION

The idea behind the development of this application stems from the fact that Ireland is one of a few countries which does not routinely offer a foreign language to preschool or primary school children. Research shows us that introducing foreign languages for the first time in secondary schools means that learners fail to benefit from the brain's natural capacity to acquire language with greater ease at a younger age.

More than 70% of Irish people cannot speak a language other than English or Irish. This has led the government to provide secondary school teachers with free training under a plan to boost the number of educators for French, German, Spanish, Italian, Russian and Japanese. Clearly there is a problem with the uptake of foreign languages in Ireland today and this is a problem that needs addressing.

Currently there is no platform around that allows students to monitor the activity of their students. A platform which provides an in-depth analysis of each student in a class whilst also providing a general overview for the whole class. While the motivation for this app is tailored around solving the issue with the uptake of foreign languages in the country, it also aimed at solving an issue regarding the tracking of student progress.

While the analytics for this app are all generated using the activities used in the application, the analytics are quite general, including averages and worst/best performing activities. This makes the analytics section of this app portable to any subject that a student may be participating in and not just a language being learnt through this application.

### 1.2 RESEARCH MOTIVATIONS

---

#### INTRODUCTION OF IPADS IN SCHOOLS

A 2014 article in the Irish Times stated that about 100 of Ireland's 700 secondary schools would be introducing iPads to the students in their schools. This number has been steadily rising as ICT and technology has started to play a major role in the education of children. The use of tablets has the potential to be a fantastic move by schools if executed correctly and appropriately. The student now has all the resources in the world at their fingertips allowing to expand their knowledge independently.

With regard to classwork/homework tablets open up a whole new range of possibilities as classwork/homework can be submitted to a teacher instantly but it can now take many different forms. This application exploits this opportunity opened up the introduction of tablets and seamlessly allows teachers to track their students work and their progression.

Whilst many will disagree with the introduction of iPads and tablets in secondary school, including Psychologist Dr David Carey who told the Irish Times: *"There's nothing technology can do that a highly skilled teacher who can make the curriculum interesting and accessible can't do."* This project will aim to detail that the benefits of this technology injection far outweigh the cons once it is regulated and safety is considered.

---

## LEARNING A LANGUAGE USING TECHNOLOGY

This project will also aim to highlight the benefits of learning a language using technology. Learning a language can be a daunting task and technology can be a way to facilitate an easier way of beginning this task. While technology may not make you fluent it increases interest if used and developed appropriately. Young people especially, live through technology and so in an effort to increase the uptake in languages, we must capitalise on this. The use of technology with language teaching means it does not need to be directly taught in the classroom, it means that students themselves can learn anywhere.

### 1.3 GLOSSARY

Below is a list of technologies that were new to the developer which were used in the development of this application. An explanation for each technology is also provided

- Keras – a high level neural networks API, written in Python.
- Tensorflow – An open source machine learning framework.
- Flask – a microframework for Python based on Werkzeug.
- SQL – Language for communicating with databases.
- Convolutional Neural Networks – Type of neural network that is very good at image recognition and classification.
- Transfer Learning – Using a pre-trained neural network to solve a similar but different problem.
- Classification Algorithms - is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation.
- Jinja - Jinja2 is a modern and designer-friendly templating language for Python.
- Javascript – programming language of HTML and the web.
- REST – A client-server, cacheable communications protocol. It is used for designing networked applications.
- NGINX - Nginx is a web server which can also be used as a reverse proxy, load balancer, mail proxy and HTTP cache.
- Gunicorn - The Gunicorn "Green Unicorn" is a Python Web Server Gateway Interface HTTP server. The Gunicorn server is broadly compatible with a number of web frameworks, simply implemented, light on server resources and fairly fast.
- Tensorflow JS - A WebGL accelerated, browser-based JavaScript library for training and deploying ML models.

### 1.4 CONDUCTED RESEARCH

---

#### EVIDENCE OF THE BENEFIT OF IPADS IN SCHOOL

“Seamless learning” or “anytime, anywhere learning” are ideas mentioned by Sha, Looi et al. 2011 [S1]. This means that the tablets can be used outside the classroom. Students now can participate in class from home if they are unable to attend school by using their tablet. This is a major advantage and really bridges the gap between school and their normal lives.

[S2] describes a study of teachers' attitudes towards the use of iPads in secondary schools in Ireland and found that the general attitude amongst teachers was positive. Teachers during this study mentioned how they had reduced work load and others discuss how students collaborated more on schoolwork when using iPads.

[S3] is a study about the introduction of iPads in a Northern Ireland school in 2016. From this study the teacher admits improved literacy from the participating students. This study also reports how the use of tablets led to students sharing knowledge and helping each other.

[S4] outlines how iPads can help students see the benefits of what they are actually learning. It also states how students felt more motivated regarding their coursework. This study also states how schools became frustrated with the lack of interactive content and they felt this had the most potential to improve teaching and learning.

Some pedagogical benefits of using tablets are discussed in [S5]. After a study was performed on multiple Finnish schools, some of the benefits included: increased student motivation, independent learning. Another benefit not to be overlooked included being able to support low-performing students. Much of the same benefits are discussed in [S6] which also mentions that students found it much quicker to access information and that "tablet makes learning easier when the learning process involves more interaction".

---

#### EVIDENCE OF THE BENEFITS OF USING TECHNOLOGY TO LEARN A LANGUAGE

Paper [S7] recognises that technology has made an impact on language learning. It admits that Automatic Speech Recognition (ASR) can greatly improve pronunciation in students. The study also found that the use of technology led to more dictionary look-ups amongst the students highlighting a more engaged classroom.

[S8] states that things like pronunciation can be taught with ease using a smartphone/tablet. It also outlines how traditional teaching methods are no longer motivating and that learners are more interactive, meaning a smart device is an ideal candidate to aid in learning a second language. A smart device can help with reading and listening skills due to the multi-modal output they produce.

---

#### CHOOSING IMAGE CLASSIFIERS

##### **Every-Day:**

##### **Translation: Stronger Model**

When choosing which pretrained model which would work best for this application research was carried out. For the object translation aspect of the project, a strong model was required with good accuracy as speed was not so much of an issue. For this I chose the Inception V3 Model as it has a high accuracy without having too many parameters. According to [S9] Inception V3 has a "Top-1 Accuracy" of 78% whilst only having 21 million parameters. This is compared a "Top-1 Accuracy" of 80.4% for Inception ResNet V2 which 54 million parameters [S10]. The design decision was made that the extra 2% in accuracy could not be justified due

to more than double the parameters and undoubtedly slower speed, and so Inception V3 was used.

**Activity:** Tensorflow js/MobileNet

This project required some real-time image classification in order to make activities quick, fun and interactive. This was particularly important in the “Find Me” activity which requires the student to find an object in their surroundings before the time runs out. In a paper written by Google engineers [S10] they describe a new Convolutional Neural Network architecture which has only 7 million parameters whilst providing high accuracy 71% on the ImageNet dataset. This architecture is much smaller and faster than other architectures and thus makes it the ideal candidate for real time image classification.

Other architectures under consideration for this activity was ShuffleNet yet there was no real evidence supporting a superiority over MobileNet as it had a similar accuracy.

### **Categories: Transfer Learning**

Transfer learning was applied to category classification activities. This involved creating CNNs to classify different categories such as sport, food etc. As there were no readily available datasets or models for these categories, transfer learning was applied on a smaller dataset that was generated by web scraping. Transfer learning required using a pretrained model MobileNet and using the top layers of that model and train the bottleneck of the model on the generated dataset. The next stage is to train and append your own model to the end of the pretrained model. When trained on MobileNet these models can be converted to client-side models using the Tensorflow JS converter. This creates a minimised version of the transfer learned model that can be run by the browser using the GPU of the device on which it is running.

---

## MICROSERVICES

Much research was carried out in terms of the benefits and pitfalls of using microservices. Whilst the advantages and disadvantages were discovered, the design implications of using this architecture also had to be researched.

Whilst researching the direct advantages and disadvantages of the architecture the following was discovered:

From [S11] it is clear that some of the immediate benefits of the architecture included: increased availability, improved reliability, easier maintenance. [S11] also went on to outline some of the drawbacks too though, which include reduced performance due to the increase in network latency and the reduced number of in-memory calls compared to that of a monolithic architecture. Also, it states that security may become an issue. [S12] discusses similar advantages but also indicates that clear boundaries were an advantage of the architecture as well as increased scalability and independent deployment meaning services can be deployed independently from the rest of the application. With these benefits come some disadvantages like increased complexity which is mentioned in [S12].

In order to sufficiently decide whether the microservices architecture was suitable research was needed on the direct design implications of the architecture. There are many design implications of this architecture according to the research. Some of the positive implications include: Loose Coupling/High Cohesion, Decentralized Data Governance, Language-Agnostic, Independent Deployment and Fault Isolation. Some of the negative design implications included: increased network latency, granular scaling, security vulnerabilities, code duplication, refactoring difficulty and lack of tooling. It is clear that the positive implications lead to improved scalability, whilst the negative implications lead to reduced performance. As a result, the decision to use a microservices architecture becomes a trade-off between scalability and performance. Thus, for this application it was important to first prioritize the non-functional requirements before making this decision. The most important aspect of this project was its ability to scale and thus I decided to use a microservices architecture.

The main design implications that concerned this project were:

**Loose Coupling/High Cohesion:** This enables quick deployment of software according to [S13, S14]. [S14] states that components can be deployed individually leading to less errors.

**Fault Isolation:** As services are independent of each other another positive design implication is fault isolation. This means failure in one service does not mean your other services will fail [S15]. This alone has led to companies like Spotify migrating to microservices.

**Granular Scaling:** Knowing how granular to go when creating a microservice can become a tricky process. Too granular and it may lead to service duplication [S16] and increased maintenance, not granular enough and your code remains similar to that of a monolith.

---

## LEARNING PYTHON FLASK

While developing this project, an Udemy course was undertaken which facilitated the learning of Python Flask. As well as this, various YouTube channels and blog posts were extremely useful in understanding more specific aspects of the framework. In particular the following blog post was quite useful:

<https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i-hello-world>

Stackoverflow.com was an invaluable research tool when there was an occurrence of a problem or complication to which the solution had not been learned through the aforementioned courses and tutorials.

---

## MACHINE LEARNING RESEARCH

### Convolutional Neural Networks

The two sources listed below provided some great insight into Convolutional Neural Networks for beginners. YouTube videos by Siraj Raval were very useful also.

<https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>



<https://www.pyimagesearch.com/2018/04/16/keras-and-convolutional-neural-networks-cnns/>

## **Speech Recognition**

The following articles really helped with the speech recognition aspect of the project. The first article provides some really good explanations of the technology behind speech recognition, whilst the second details how to implement speech recognition using Python.

<https://medium.com/@ageitgey/machine-learning-is-fun-part-6-how-to-do-speech-recognition-with-deep-learning-28293c162f7a>

<https://realpython.com/python-speech-recognition/>

## **Text-to-Speech**

The use of text-to-speech and speech synthesis is a huge part of this application and the two resources below were really useful. The speech service used “Responsive Voice” was useful and very easily incorporated into the application.

<https://medium.com/@saxenauts/speech-synthesis-techniques-using-deep-neural-networks-38699e943861>

<https://responsivevoice.org/>

## **Learning Keras**

There are some fantastic tutorials online regarding machine learning and Keras. Most of which can be found on the Keras website <https://keras.io>.

## **TensorFlow JS (Client-Side Machine Learning)**

TensorFlow JS is a relatively new technology that enables machine learning in the browser and client-side machine learning. The TensorFlow JS website provides some great information as well as some samples to help get you started.

<https://www.tensorflow.org/js>

## **LSTM/Neural Networks**

For research on LSTMs and time series prediction I used the following:

<https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/>

<https://skymind.ai/wiki/lstm>

## **POS Tagging**

To effectively apply POS Tagging in the project the following website was consulted:

<https://nlp.stanford.edu/software/tagger.shtml>

This website is maintained by Stanford who provide a state-of-the-art POS Tagging system for multiple languages.

A paper was also consulted regarding the benefits of POS Tagging. It states that it can aid in text simplification and this is especially useful when someone is “language-impaired” or does not have complete understanding of a language [S16].

[S17] states that POS Taggers are useful as they attach a word-class membership to each word and as a result disambiguate the many words that belong to more than one part of speech. The use of this will enable the application to show whether a foreign word is a noun, verb etc. and this is quite useful for someone who newly learning a language.

---

## DATA ANALYTICS

When researching on approach to perform effective data analytics the following web sites proved useful:

<http://www.data-analysis-in-python.org/>

<https://tech.labs.oliverwyman.com/blog/2016/08/23/analytics-with-sqlalchemy/>

Using ORM in the form of SQLAlchemy served was useful and made interaction with the tables of the database far easier for producing the relevant analytics in an efficient manner.

Extra research was carried out on the benefits of Student Data Analytics, while this mostly directed to higher education it was also quite applicable to a secondary school or primary school. Here are some of the findings from this research:

It has been mentioned that predictive analytics in learning analytics can identify students that may be in need of support [S18]. [S18] also mentions how descriptive analysis, which has an understanding of what has happened (pie charts, bar charts), is crucial to making an informed action regarding students. [S3] backs up this point and also states how school analytics can help a school make steps towards improvements.

Other papers mention how some online activities can “assist in detecting students’ progression in the whole learning mechanism” [S19]. [S20] states the increase in data and data analytics can enhance the learning process and enrich the learning experience. In [S20] it is also mentioned that teachers can benefit from data analytics by tracing students’ learning process and pay attention to those students who are at risk of falling behind as stated in [S18].

Finally [S20] mentions that data analytics could be used to track the performance of teachers also. There are multiple ways this could be done but monitoring the performance of their classes as a whole would be the most obvious.

## 2. DESIGN

### 2.1 MODEL VIEW CONTROLLER

“The Model-View-Controller (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller.” – Tutorialspoint.com

The Model View Controller design pattern has been used to formulate the logical structure of this project. This includes the Flask project, but also the architectural structure of the broader system. This design pattern has driven the development of the application and is followed strictly throughout the project.

#### **Model:**

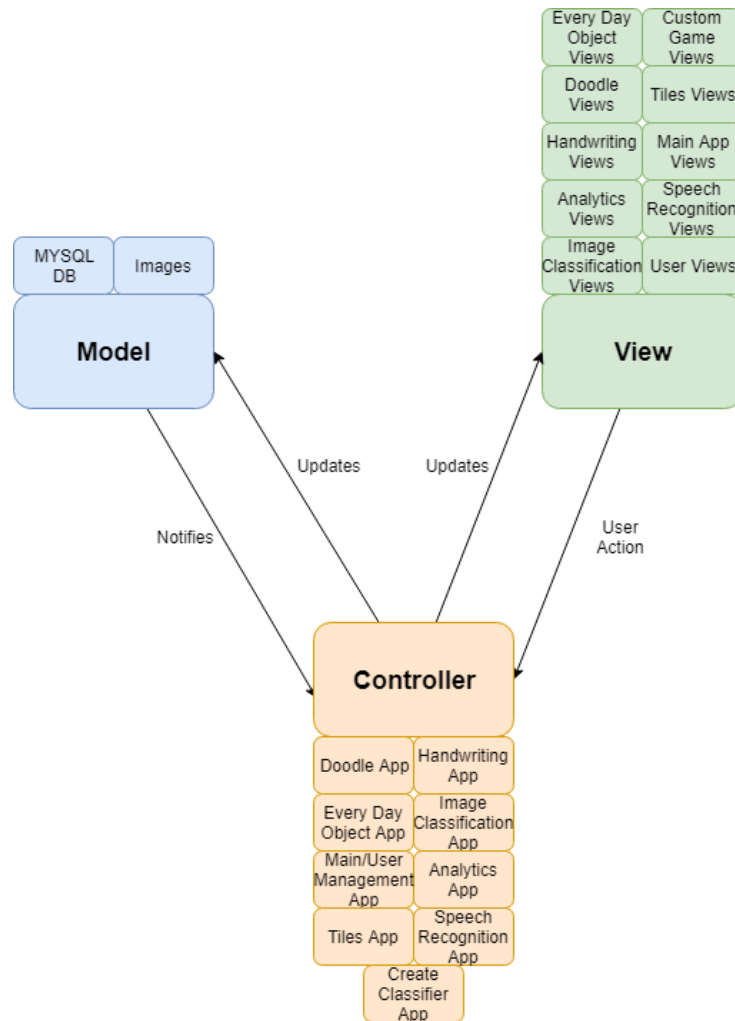
The model represents the back-end or storage capabilities of the platform. In the case of EyeLearn, on the Web application, the model is made up of MYSQL tables. This MVC model is responsible for the data storage of every component in this system.

#### **View:**

The view represents the front-end. This is where data comes together to be displayed to the user. In the case of EyeLearn, the view is comprised of each Jinja template which renders a html page. These views are compatible with every type of device and thus makes the app device-agnostic.

#### **Controller:**

The controller represents the aspect of the platform which sits between the model and the view. The controller is responsible for retrieving data from the model, potentially performing operations on said data, before either passing it back to the model or indeed passing it to the view to be visualised. In this project, the Controller is each “routes.py” file in the Flask application. These controllers also have access to microservices. These are accessed via a REST interface and the controller may receive data from them before passing it to the UI.



## 2.2 MICROSERVICES

Microservice architecture, or simply microservices, is a distinctive method of developing software systems that tries to focus on building single-function modules with well-defined interfaces and operations. It is an architectural style that structures an application as a collection of loosely coupled services, which implement business capabilities.

### Microservices in EyeLearn

As there are multiple Convolutional Neural Network Models being used by EyeLearn, it was necessary to split one into its own service or backend that performed its sole function. This choice was taken as some models need to perform different image pre-processing techniques before making predictions. Other microservices include a service to generate grade prediction for students or a class using an LSTM and another which performed phrase translation and POS Tagging simultaneously.

Dividing all the networks into a series of services makes testing them easier and also makes them reusable by EyeLearn and other apps. They all contain their own endpoint which means they can be deployed anywhere and do not need to be hosted on the same server as the front-end. It allowed for

different dependencies to be installed on different servers and meant there was less chance of conflict between dependencies.

This means the front-end could be deployed on a standard server and the backends handling the models can be deployed on a server with GPU access. While the use of microservices can lead to extra financial overhead due to increased hardware needs, it makes them far more reliable as multiple instances could be spread across multiple datacentres.

There are many design implications of using microservices some of are outlined in section 1.4. The main benefits encountered when using this architecture included increased testability and increased scalability. Granular scaling became an issue, but the positive design implications loose coupling/high cohesion and fault isolation greatly outweighed this negative. Ease of deployment was something that was quite noticeable when using microservices also.

### **Object Translation Microservice**

This microservice receives an image and the user's current language from the "image classification controller". This image has been taken using a device or has been uploaded from the user's gallery. Some pre-processing is performed on the image and it is then passed to the Inception V3 model which generates predictions for the image. These predictions are then translated into the user's chosen language and the English and translated predictions are returned to the controller to be displayed on the view.

### **Speech Recognition Microservice**

This microservice is used to classify a recording that has been passed from the view to the controller. This is in the form of a .wav file. The use of a pretrained speech recognition algorithm is used for optimal results. The recognition algorithm will convert the speech to text and this text will be passed back to the controller where it is used to see if the student has gotten the correct answer. The controller will subsequently update the view.

### **Grade Prediction Microservice**

This microservice is contacted on scheduled occasions via a script organised using CRONTab. This microservice receives an array of values representing the averages of a student or class on a daily, weekly or monthly basis. The microservice receives this data and trains an LSTM for each user/class instance and generates a grade prediction for that user or class for the next day, week or month. This prediction is returned to the scheduled script which in turn stores it in a table in the MYSQL Database to be used in the analytics views.

### **Phrase Translation & POS Tagging Microservice**

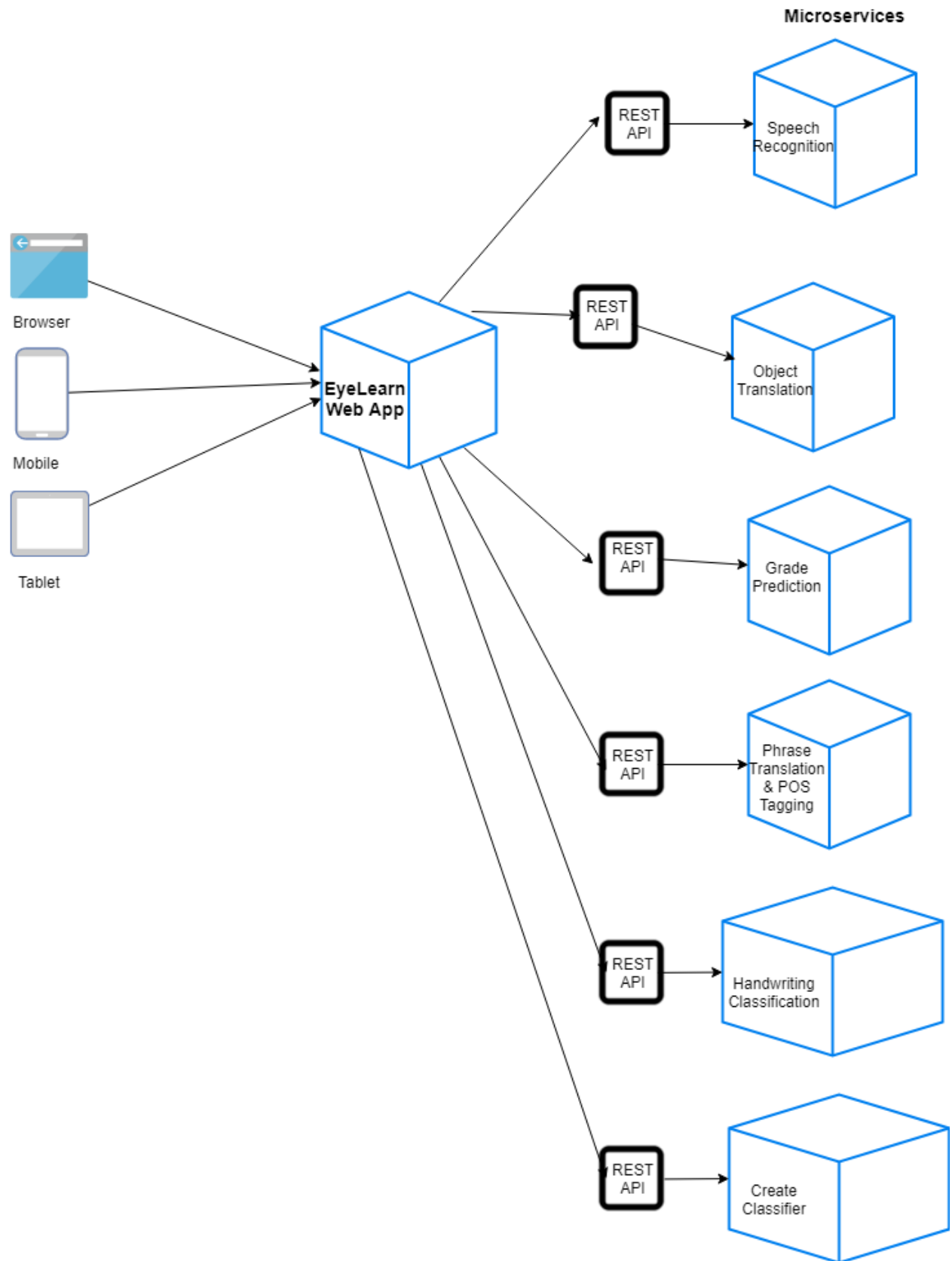
This microservice receives data from the POS Tagger view. This data is in the form of an English string and the user's chosen language. This string is duly translated into the user's chosen language. The English and its translation are then POS Tagged, and this result is returned to the view where it is displayed to the user.

### **Handwriting Classification Microservice**

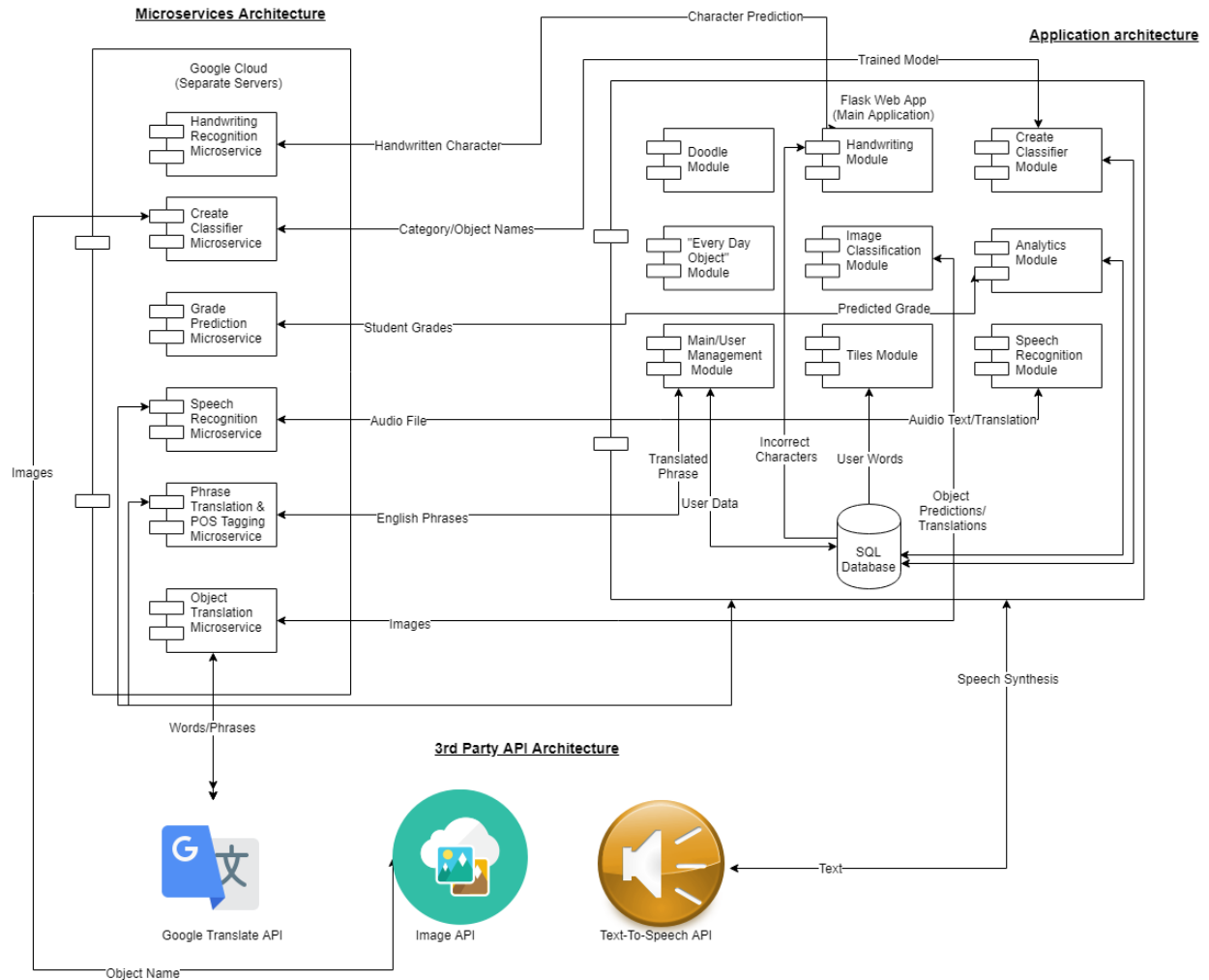
This microservice contains two endpoints. One endpoint handles the input of handwritten letters while the other handles the input of handwritten numbers. The microservice contains two models. One can return predictions for letters whilst the other returns the prediction for numbers. The endpoint to which the handwritten image is sent to is chosen by the controller. When an endpoint receives an image, it produces the prediction using the models and return the prediction the controller which passes it back to the view.

### **Create Classifier Microservice**

This microservice contains one endpoint “/trainModel” and is hosted on a server with GPU access. It receives a list of objects and a category and downloads images pertaining to the list of objects before performing transfer learning using these images to create a classifier for the specified category. This microservice uses tensorflow to perform transfer learning using MobileNet and then converts the model to Tensorflow Js format so that it can be used in the browser for client-side machine learning. This microservice then transfers the trained model to the main app to be used by the students.

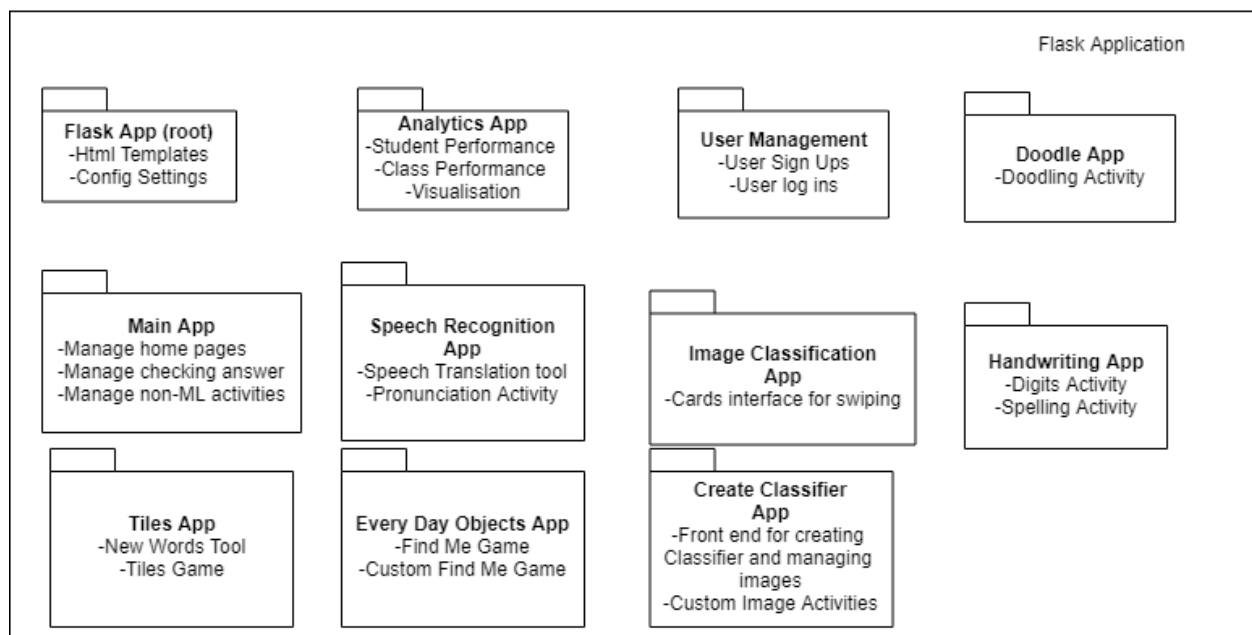
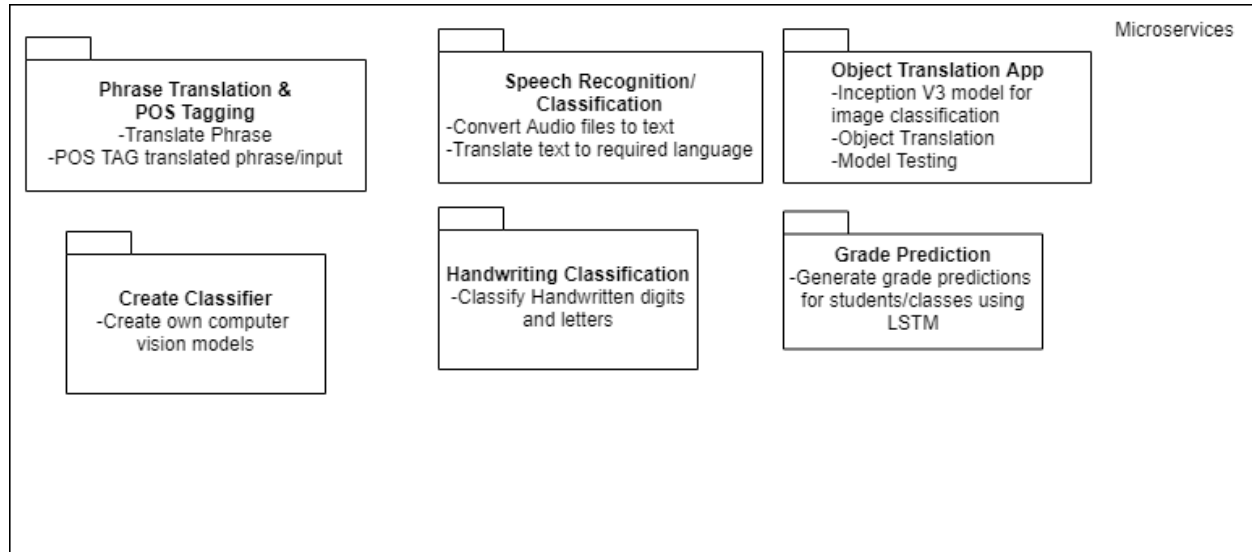


## 2.3 ARCHITECTURAL OVERVIEW DIAGRAM





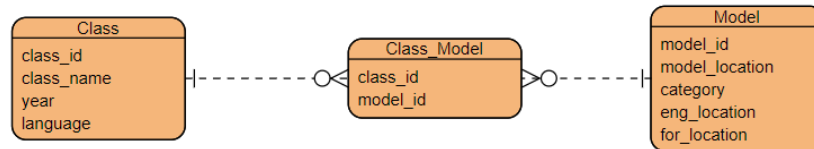
## 2.4 LOW LEVEL CLASS (PACKAGE DIAGRAM)



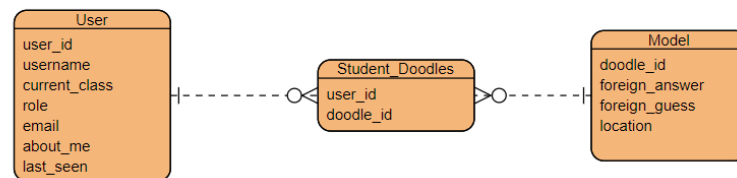
## 2.5 ER DIAGRAMS

The database is in Third Normal Form and here are some examples of the relationships between entities:

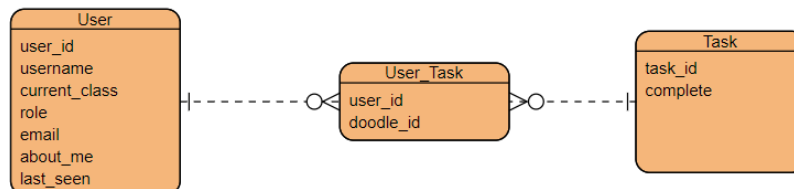
### Class Models



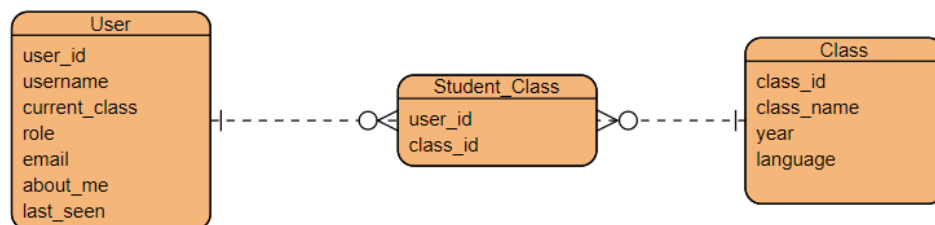
### Student Doodles



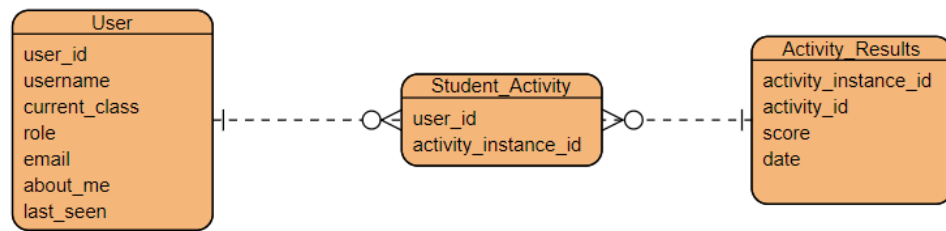
### User Tasks



### Student Classes



### Student Activities



## 2.6 USER INTERFACE DESIGN – SCHNEIDERMAN’S RULES

When designing the user interface, principles of HCI, more specifically Schneiderman’s eight golden rules of interface design were considered.

**Consistency:** Being consistent in the design across all the activities/tools was essential. As such, the same theme has been used, as well as the same coloured buttons. The application has the exact same colour theme aimed at giving the user a seamless and integrated experience. When they see bright blue and white, they know it is time to use EyeLearn.

**Shortcuts:** The navbar is persistent throughout the whole application. It also contains an “Analytics” link which brings the user directly to the analytics page of the current class they are using in the application.

**Dialog to yield closure:** Each Activity starts and begins in the same way. The user visits the activity pages begins their chosen activity, is returned to the answer screen where they can either continue playing or end the game. Upon ending the game, they are always returned to the EyeLearn home page unless they have completed a course when they will then be redirected to their dashboard.

**Informative feedback:** When the use plays an activity and submits an answer they are immediately told if they are correct or incorrect. If they get a question incorrect they have the option to view the answer or to try again. If they translate an object, both the English and the translation is provided, similarly this occurs with the speech translation and phrase translation/tagging.

**Error handling:** When a page is loaded incorrectly, or a form receives incorrect input, the user will be told by the application that there is an error and the type of error that has occurred e.g. “Incorrect Password”.

**Easy reversal of actions:** The user can easily change between classes if they are using the application for multiple teachers or languages.

**Support internal locus of control:** The user is in complete control of the navigation of the application as the navbar is present regardless of where one navigates, meaning they can quickly and easily be taken to any page of the platform in one click.

**Reduce Short term memory load:** To achieve this, the application was developed with the magical number of seven (plus or minus two) in mind. This stems from a famous psychology paper which (in an

extremely oversimplified explanation) showed that human judgement ability is dampened when more information than can be handled is presented. As such, extreme care has been given to stay below this magic number throughout the design of the application. Examples:

- 4 navigational toolbar options
- 3 pieces of analysis information given on the analytics pages
- 9 different statistical visualisations on the performance analytics page

These eight principles make for a far more appealing and intuitive application user interface.

### 3. IMPLEMENTATION

The implementation of this platform's functionality is spread across its one main application and six microservices:

#### 1) The Python-Flask Web Application:

1. Signing Up/Logging in
2. Tracking of user/class performance and visualisation over time
3. Participate in multiple activities (images, speech, sound, text, handwriting, doodling)
4. Provide the use of translations tools
5. Keep track of new words learnt through the application
6. Track common mistakes made by the users in spelling (dyslexia)
7. Allow teachers to create their own computer vision games.

#### 2) Microservices:

1. Object Translation
2. Speech Translation
3. Phrase Translation & POS Tagging
4. Handwriting Classification
5. Future grade prediction using an LSTM and previous results
6. Create Classifier

#### 3.1 PYTHON FLASK WEB APPLICATION

---

##### IMAGE BASED ACTIVITY VIEWS

There are number of image-based activities in EyeLearn. The image on the left displays the "Find Me" activity which requires the user to find every-day objects around them. The "Find Me" Activity classifies objects in real-time using Tensorflow JS which enables client-side machine learning. The student is required to locate the object in the top gamebar before time runs out. Also, when the user points the camera at an object the item is read out in the user's language. When the camera is pointed at the correct item the user is redirected to the "Correct Page" and if times run out they are sent to the "Incorrect Page". As you can see the student is required to find "Souris" which is a keyboard "mouse". In the example below, you can see that a hand is shown to the camera and the client-side model correctly

classifies it as “main” (hand in French) as can be seen in the speech bubble. This whole implementation was made possible using some clever JavaScript:

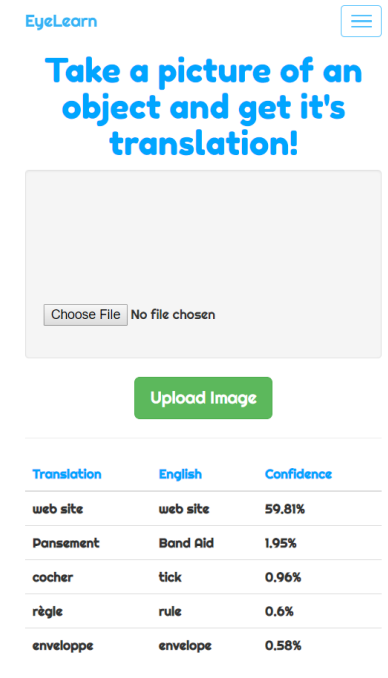
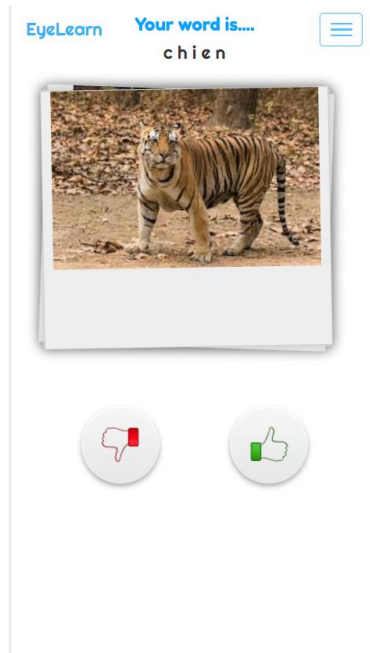
```
async function predict() {  
  const result = tf.tidy(() => {  
    const pixels = tf.fromPixels(cameraView);  
    const centerHeight = pixels.shape[0] / 2;  
    const beginHeight = centerHeight - (224 / 2);  
    const centerWidth = pixels.shape[1] / 2;  
    const beginWidth = centerWidth - (224 / 2);  
    const pixelsCropped =  
      pixels.slice([beginHeight, beginWidth, 0],  
                  [224, 224, 3]);  
    return modelPredict(pixelsCropped);  
  });  
  
  const topK =  
    await getTopKClasses(result, 10);  
  speak(topK);  
  
  loop = requestAnimationFrame(() => predict());  
}
```

Here can be seen that `requestAnimationFrame()` is used to keep calling the `predict` function recursively on the input from the camera. As a result, real-time predictions can be generated for what the camera is pointed at. In this code it is clear that the input is resized to the same input size that the model requires to ensure the predictions have a better chance of being accurate. The input size for the MobileNet model is (224, 224, 3).

The second image classification activity “Swipe Animals” uses “swipable” cards to make the activity fun and interactive. The user is given a word and host of cards that can be swiped. When the user sees an image that corresponds to the given word they must swipe right on that card, otherwise they should swipe left. If the user runs out of cards a GET Request is made to get new cards. If the user swipes right the image is classified by the client-side models. If the classification matches the answer on the screen the user is sent to the “Correct Page” otherwise they are sent to the “Incorrect Page”. The JavaScript for both the GET request and image classification can be seen below.

```
function addCards(category) {  
  var imgs = document.getElementsByTagName("img");  
  $("#tinderslide").remove();  
  $('#.wrap').prepend("<div id='tinderslide'><ul></ul></div>");  
  $.ajax({  
    url: '/getMorePictures/' + category,  
    type: 'GET',  
    success: function (data) {  
      console.log("here");  
      console.log(data.data);  
  
      for(var i = 0; i < data.data.length; i++) {  
        var temp = i + 1;  
        var elem = "<li class='pane' + temp + '><div clas";  
        $('#tinderslide ul').prepend(elem);  
      }  
      $("#tinderslide").jTinder(options);  
    }  
  });  
}
```

```
// like callback  
onLike: function (item) {  
  
  var image = item.find('img')[0];  
  predict(image).then(x => {  
    if(x.includes(answer)) {  
      var formInfo = document.forms['send'];  
      formInfo.guess.value = answer;  
      $('#send').trigger('submit');  
    } else {  
      var formInfo = document.forms['send'];  
      formInfo.guess.value = x[0];  
      $('#send').trigger('submit');  
    }  
  });  
}
```



## ANALYTICS VIEW

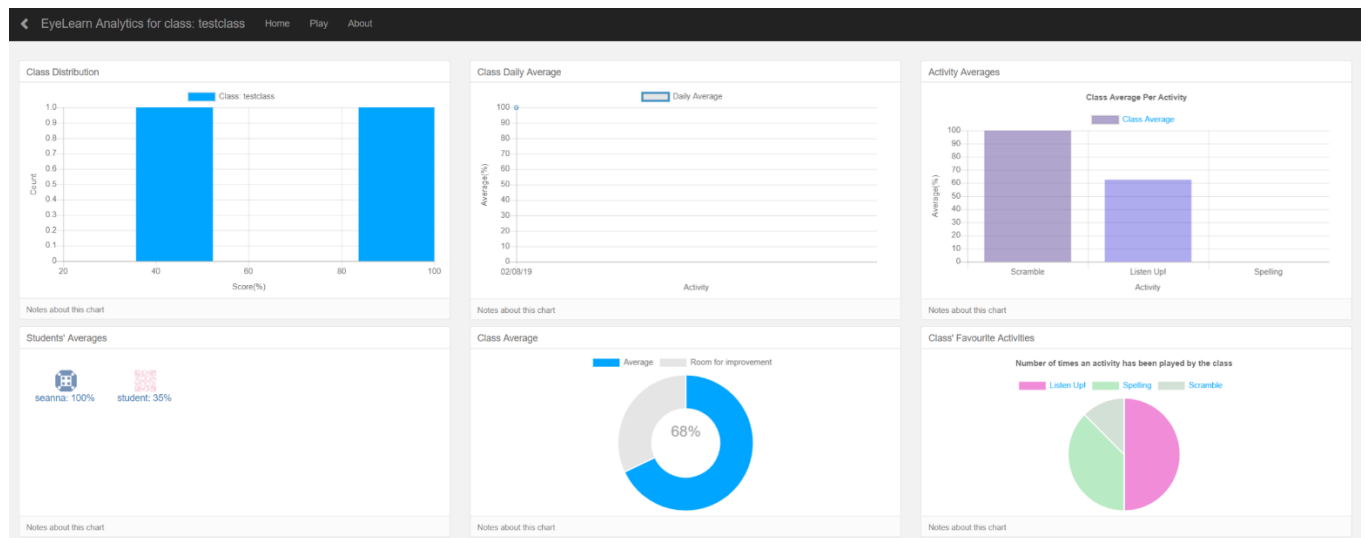
The analytics views can be used by both students and teachers to monitor their progress or class progress respectively. The charts have been designed using Chart JS. The use of this library means that the charts are interactive and load dynamically. The data for these charts is retrieved from the MYSQL Database using “User Analytics” controller. The controller in turn passes the data to the client-side JS to be used in the view.

Both a student and a class have similar analytics, but some of the differences include:

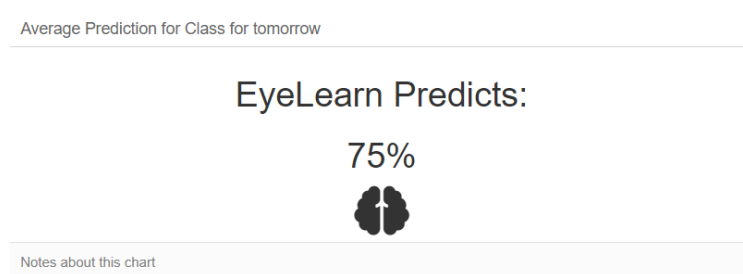
A class analytics page includes a list of all the students in the class with their averages. The teacher from there can click on the students name to be brought to that student’s analytics page where they can monitor the student in depth.

A student can see their position in the class without seeing who is above and below them. This enables the student to see where they rank in the class without giving away any details about their colleagues.

Some useful analytics include the ability for a teacher to monitor if any of their students regularly mix up letters such as “b” & “p” or “f” & “t”. This could help in the early detection of dyslexia detection.



The below snippet is a piece of analytics from the analytics page. This is prediction from the Grade Prediction Microservice. It tells the student or teacher what their grade could be in the next day.



## HANDWRITING/DOODLE VIEWS

The below images show the use of a canvas to receive user input. In the first image, it shows a version of the spelling game. In this activity, the user receives a word in English and must spell it letter by letter in their foreign language using the canvas. Each letter is classified by the handwritten digits microservice. This activity helps student with their spelling of foreign words but also tracks their handwriting to make sure they do not mix up letters or numbers frequently ("b" & "p" or "6" & "9").

The second image represents the "Doodle" activity. This activity provides the student with a foreign word and they in turn must doodle the word. In the below image "prise de courant" can be translated from French to "plug". The user must in turn try doodle a plug before time runs out. This has only been made possible again using Tensorflow JS as every time the user lifts their finger the doodle is passed to a model to classify the doodle in real-time. Again, this is being handled using some every clever Javascript which will be shown below.

```

canvas.on('mouse:up', function(e) {
  getFrame();
  mousePressed = false
});

```

```

function getFrame() {
  //make sure we have at least two recorded coordinates
  if (coords.length >= 2) {

    //get the image data from the canvas
    const imgData = getImageData()

    //get the prediction
    const pred = model.predict(preprocess(imgData)).dataSync()

    //find the top 5 predictions
    const indices = findIndicesOfMax(pred, 5)
    const probs = findTopValues(pred, 5)
    const names = getClassNames(indices)

    //Display and speak the results
    speak(names, probs)
  }
}

```

From this you can see every time a user lifts their finger from the doodle canvas the `getFrame()` function is called which makes a prediction about what has been drawn before speaking and displaying the results.

EyeLearn

Your word is....

c a r



Add Letter

Clear

Your Guess:

Submit Answer

Remove Letter

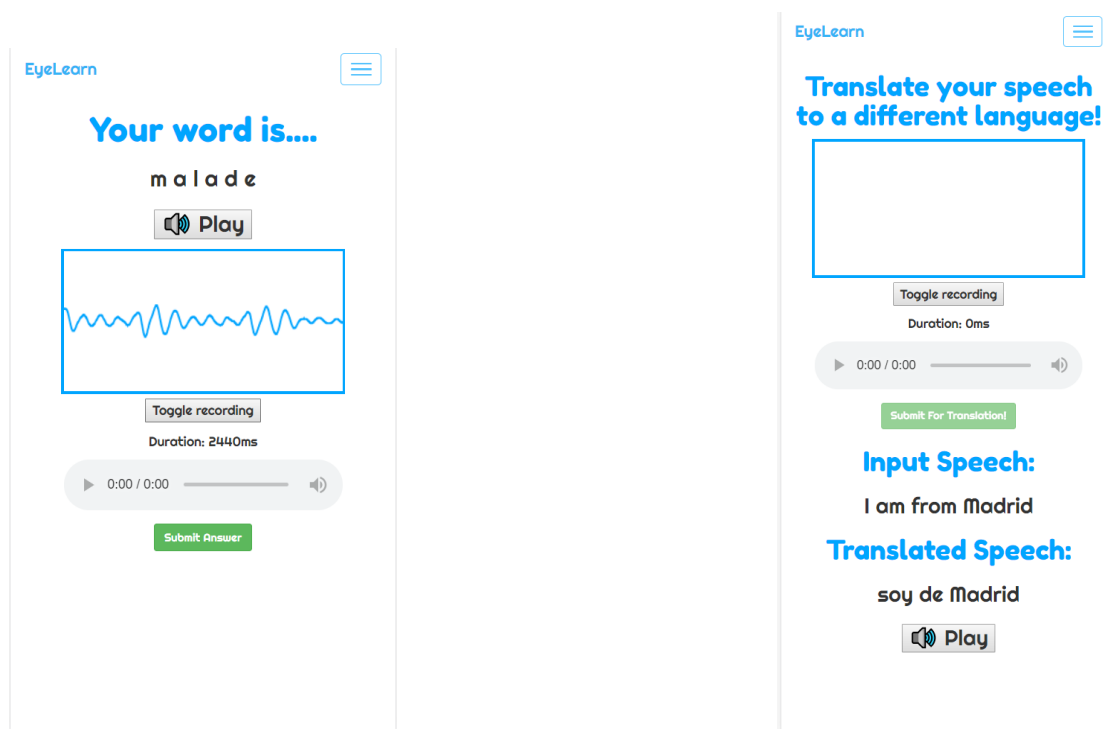


## SPEECH RECOGNITION VIEWS

In the speech recognition activity shown on the left below the user is presented with a word in their chosen language. They have the option to listen to the pronunciation of this word. They then must record themselves saying this word as close as possible to the correct pronunciation to get the answer correct. This is quite a difficult activity as your pronunciation must be near perfect in order for the French speech recognition microservice to recognise the student's utterance. Then user can toggle their



recording using the “Toggle Recording” Button. When they speak their speech frequency will be shown in a box as shown below. Then when the user clicks submit, the speech is sent to the Speech Recognition Microservice using JavaScript Ajax Post request.



---

## PHRASE TRANSLATOR & POS TAGGER

The Translator and Tagger Tool provides a nice visualisation for the user of the word type for both their input sentence and their translation. This helps the user match up the words from their input to their translation. For someone newly learning a language it may be easy to assume “bleue” means “house” and “maison” means “blue” due to position alone. This tool removes the potential of this mix up occurring as “maison” and “house” are both highlighted as nouns and “bleue” and “blue” are both highlighted as adjectives. As mentioned earlier in the research section, POS Taggers can be used for text simplification when learning a new language.

The highlighting of the words has been performed using JavaScript which changes the colour of the word based on the tags that have been returned from the Translator & Tagger Microservice. The snippet which performs the word highlighting can be found here:

```
// Start tagging
function tag() {
  $('#spinner').show();
  var text = $('#text').val();
  text = text.replace('»', ' ');
  text = text.replace('«', ' ');

  // No proxy when developing (.*local.* in host)
  var url = "https://postagger.eyelearn.club/getPhraseTranslation1"
  console.log(language);
  $.ajax({
    url: url,
    type: "POST",
    data: {"phrase": text, "language": language},
    success: callback
  });
  return false;
}
```

Here is the POST Request of an input string and a translation language to the Translation & Tagger Microservice. This returns both the POS Tagged input string and POS Tagged translated string. When this finished successfully it goes to the callback() function a snippet of which that can be seen below which colours the words based on what is returned from the microservice. This callback() function also adds the tooltip to the word which can be seen in the UI snippet.

```
if (tagMap[tag] != undefined && color[tagMap[tag][0]] != undefined) {
  taggedHTML += '<span class="taggedWord" style="background-color: ' + color[tagMap[tag][0]] + '>' + word + '</span>' + tag + '</span></span>';
}
else if(tag == "NEWLINE") {
  taggedHTML += '<br/>';
}
else {
  taggedHTML += '<span class="taggedWord">' + word + '</span>' + tag + '</span></span>';
}
});
$('#textTagged').html(taggedHTML);
```

---

[EyeLearn](#)
[Home](#)
[Play](#)
[About](#)

## Phrase Translator & Tagger

Enter a **complete sentence** and click at "Translate & POS-tag!". The tagging works better when grammar and orthography are correct.

**Text:**

John likes the blue house at the end of the street .

John aime la maison bleue au bout de la rue .

Edit text

▲

Noun, NOUN

Adjective

Adverb

Conjunction

Determiner

Noun

Number

Preposition

Pronoun

Verb

---

## TILES VIEWS

This view is being used as both as a tool and as an activity. As a tool, all the users newly learned words appear randomly on these tiles in their chosen foreign language. When a user taps on a tile the English translation for the word is read out by the application.

As an activity the user is given a foreign word and all the words appear in English on the tiles and the user must tap the correct tile for the given word.

The tiles on this page cleverly slide in and out of the web page and constantly change colour. When a word is displayed on tile is printed character by character. When this page is loaded, that particular user's newest words are loaded. The client side and back end code are shown below.

```
function getWords() {
  $.ajax({
    url: '/getWordsForTiles',
    type: 'GET',
    success: function (data) {
      translations = data.translation;
      english = data.english;
    },
    async: false
  });
}
```

On the client side, a GET Request is made to the backend to get the user's new vocab. In the below image we see the backend endpoint in charge of getting these words for the client. In the backend we make a call to the model to return the new vocab for the "current\_user" from the MYSQL DB. This is returned to the client as a JSON response.

```
@bp.route("/getWordsForTiles")
def getWordsForTiles():
    newVocab = current_user.get_students_new_vocab(current_user.current_class).all()
    vocabData = {"english": [vocab.english for vocab in newVocab],
                "translation": [vocab.translation for vocab in newVocab]}
    return jsonify(vocabData)
```

S'il vous	Saint Bernard	S'il vous plaît pa	S'il	Épagneul galloi
S'il vous plaît parler plus claireme	Sai		S'i	
clumber	Bretagne	Épagneul gallois	Saint	Poseur anglais
Saint Bernard		Poseur	Épagneul gallois	Poseu
clumber	Bretag	Bretagne	S'il vous plaît parler plus clairement	Bretagne

## DOODLE TRACKING VIEWS

All doodles drawn by a student using EyeLearn is tracked along with what model thought it saw as well as what the answer should have been. This allows students to go back and learn vocabulary they may have gotten wrong using this game and provides them with a fun memento that they can look back on. Here is the view for the doodles which provides a swipe image gallery and the code that hands the saving of the doodle.

## Your Doodles



Client-Side Code (JavaScript) – Pulls the image from the canvas and sends it to the “/saveDoodle” endpoint.

```
function saveDoodle(guess) {
    var canvasObj = document.getElementById("canvas");
    var img = canvasObj.toDataURL();
    data = {'doodle':img, "answer":answer, "guess":guess};
    data = JSON.stringify(data);
    $.ajax({
        type: "POST",
        url: "/saveDoodle",
        data: data,
        success: function(data) {
            console.log(data);
        },
        error: function(xhr, status, error) {
            console.log(status);
        }
    });
}
```

```
@bp.route('/saveDoodle', methods=['POST'])
def saveDoodle():
    imgPost = request.get_data(as_text=True)
    data = json.loads(imgPost)

    foreignGuess = data['guess'].strip()
    translated, model = findWordForDoodle(data['answer'].strip(), 0)
    foreignAnswer = translated

    filename = convertDoodleImage(data['doodle'], current_user.username)
    doodle = Doodles(foreign_answer=foreignAnswer, foreign_guess=foreignGuess, location=filename)
    db.session.add(doodle)
    db.session.commit()
    student_doodle = Student_Doodles(user_id=current_user.user_id, doodle_id=doodle.doodle_id)
    class_doodle = Class_Doodles(class_id=current_user.current_class, doodle_id=doodle.doodle_id)
    db.session.add_all([student_doodle, class_doodle])
    db.session.commit()
    return "saved"
```

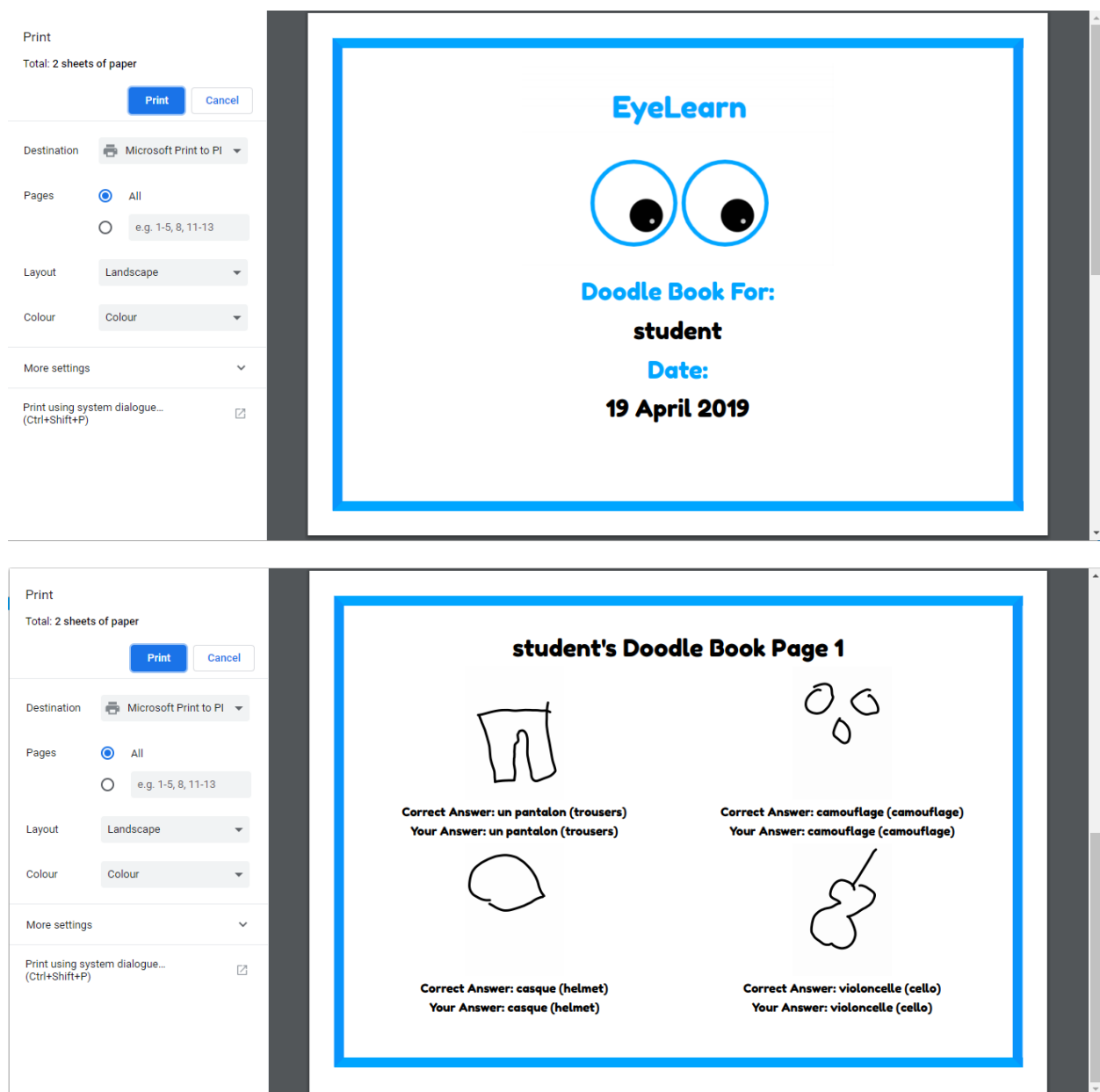
This endpoint receives the canvas image and converts it to a PNG using the “convertDoodleImage()” function. The location of this image is stored in the database along with the correct answer for the doodle and what the model saw in the student’s doodle. As you can see the Doodle, Student\_Doodle and Class\_Doodle Tables are used.

## PRINTABILITY

There are many pages in the EyeLearn app that can be printed by the user. These include images for different games like “Animals, Sports, Vehicles and Custom Find Me”. Also, when a teacher creates their own computer vision game teacher and students can print off the images to use with the game. The student can also print off their stats when they complete a course in a class, also when they complete all courses in a class they can print off a certificate of completion and also, from their dashboard they can view and print off their doodles they have completed along with their translation. Any page that is printable or can be saved will have the following button attached to it:

Print/Save

Here is the printable version of the doodle tracking page shown above:



All printed pages on EyeLearn use the same layout. This required writing separate CSS for pages when they needed to be printed and using specific print styles. When the print button is clicked the javascript “window.print()” is called in a new tab.

### 3.2 PYTHON FLASK MICROSERVICES

Each microservice has been deployed to their own separate server based on the requirements that they need. Each microservice has been deployed using NGINX and Gunicorn.

The microservices were designed using the API Gateway Pattern and were decomposed by business capability. A business capability is something a business does in order to generate value. The business capabilities for EyeLearn include object translation, speech translation, phrase translation & POS Tagging, Grade Prediction and Handwriting Classification. Each of these business capabilities can be viewed as a service.

The API Gateway acts a single point of entry for any microservices call. It can be contacted using some lightweight mechanism like REST.

Here are the exposed endpoints for each microservice:

### Object Translation Microservice

```
@app.route('/translateObject', methods=['POST'])
def translateObject():

    image = request.files['image1']
    image.save("object.png")
    language = request.form['language']

    basewidth = 300
    try:
        image = Image.open("object.png")
        for orientation in ExifTags.TAGS.keys():
            if ExifTags.TAGS[orientation] == 'Orientation':
                break
        exif = dict(image._getexif().items())

        if exif[orientation] == 3:
            image = image.rotate(180, expand=True)
        elif exif[orientation] == 6:
            image = image.rotate(270, expand=True)
        elif exif[orientation] == 8:
            image = image.rotate(90, expand=True)
        wpercent = (basewidth / float(image.size[0]))
        hsize = int(float(image.size[1]) * float(wpercent))
        image = image.resize((basewidth, hsize), Image.ANTIALIAS)
        image.save("object.png")
        image.close()

    except (AttributeError, KeyError, IndexError):
        # cases: image don't have getexif
        pass

    target_size = (299, 299)
    image = processImage("object.png", target_size)
    with graph.as_default():
        results = model.predict(image)
        labels = decode_predictions(results)
    translations = get_translation_free(labels[0], language)
```

Here is the endpoint for the Object Translation Microservice. As you can see this endpoint receives both an image and language from the client in the form of a POST request. The image is initially saved to be processed (it is later deleted) and then the image is resized and reoriented based on how it was uploaded e.g. if an image is taken using a mobile device the image is resized and reoriented.

The next stage is that the image is prepared to be passed through the Inception V3 model using the `processImage()` function. Once the preprocessing has occurred the image is classified using the `"model.predict(image)"`. This generates predictions which are then passed to the translation function which retrieves translations for these predictions in the language given in the POST request.

Finally, both the English predictions and Translated predictions are returned to the client who made the request.

## Phrase Translator & Tagger Microservice

```
@app.route('/getPhraseTranslation', methods=['GET', 'POST'])
def getPhraseTranslation():
    phrase = request.form['phrase']
    language = request.form['language']

    translatedPhrase = get_translation_free(phrase, language)

    res_english = pos_tagger_english.tag(word_tokenize(phrase))
    simplified_pos_tags_english = [(word, map_tag('en-ptb', 'universal', tag)) for word, tag in res_english]
    simplified_pos_tags_translated = []
    if language == "fr":
        res_french = pos_tagger_french.tag(word_tokenize(translatedPhrase))
        print(res_french, res_english)
        simplified_pos_tags_translated = map_french_tag_to_universal(res_french)
    elif language == "es":
        res_spanish = pos_tagger_spanish.tag(word_tokenize(translatedPhrase))
        simplified_pos_tags_translated = map_spanish_tag_to_universal(res_spanish)

    taggedPhrase = [' '.join(str(i) for i in tup) for tup in simplified_pos_tags_english]
    taggedTranslatedPhrase = [' '.join(str(i) for i in tup) for tup in simplified_pos_tags_translated]
    taggedPhrase.append("NEWLINE")
    taggedPhrase = taggedPhrase + taggedTranslatedPhrase
    data = {"taggedText":taggedPhrase}
    print(data)
    return jsonify(data)
```

The “getPhraseTranslation” endpoint receives an English input phrase and a translation language in a POST request. The input phrase is translated into the chosen language initially. Then the English input is tokenised and POS Tagged using the Stanford NLP tagger. Based on the language, the translated phrase is then tagged and mapped to the universal POS Tagset using some functions defined below the endpoint. Both the tagged English phrase and tagged translated phrase are returned.

```
b'{"taggedText":["She_PRON","is_VERB","running_VERB","very_ADV","fast_ADV","NEWLINE","Ella_PRON","esta_VERB","corriendo_VERB","muy_ADV","rapido_ADJ"]}\n'
```

The output that is returned to the client can be seen above. This output is then displayed with its respective tags as shown earlier in the snippet of the UI of the Phrase Translator and Tagger Tool.

## Speech Recognition Microservice

In the below images are displayed the endpoint for the Speech Recognition Microservice and the code which actually generates the transcript from the audio.

```
@app.route('/predictWord', methods=['GET', 'POST'])
def predictWord():
    #
    print('lang' in request.files)
    audio = request.files["audio_data"]
    recognitionLanguage = str(request.files['recognition_language'].read(), 'utf-8')
    print(recognitionLanguage)

    print(audio)
    audio = get_audio(audio, recogniser)
    response = recognise_audio(audio, recogniser, recognitionLanguage)
    if 'lang' in request.files != None:
        print("here")
        language = str(request.files['lang'].read(), 'utf-8')
        print(language)
        response['translation'] = get_translation_free(response['transcription'], language)
    print(response)
    return jsonify(response)
```

The above code receives the audio from the client along with the recognition language i.e. the language the audio has been spoken in. The audio is then formatted and sent to speech recognizer which is shown in the next snippet.

```
def recognise_audio(audio, recogniser, recognitionLanguage):
    print("in recognise audio")
    response = {
        "success": True,
        "error": None,
        "transcription": None
    }

    try:
        print("try")
        response["transcription"] = recogniser.recognize_google(audio, language=recognitionLanguage)
    except sr.RequestError:
        print("except 1")
        # API was unreachable or unresponsive
        response["success"] = False
        response["error"] = "API unavailable"
    except sr.UnknownValueError:
        print("except 2 ")
        # speech was unintelligible
        response["error"] = "Unable to recognize speech"
        response["transcription"] = "Please speak more clearly"
    except TimeoutError:
        print("timeout")
        response["success"] = False
        response["error"] = "API Unavailable - Timed out!"
        response["transcription"] = "Service down, please try again later!"

    return response
```

This code attempts to generate a transcript from the audio sent to the endpoint in a given language. As you can see there are some exception put in place in the event that the service may not be available or in the event that the audio could not be recognised. These exceptions were quite important as when trying to generate a transcript for foreign audio, your pronunciation must be precise in order for the recogniser to recognise what is being said.

## Handwriting Recognition Microservice

```
emnistModel, emnistGraph = initEMNISTModel()
mnistModel, mnistGraph = initMNISTModel()
mapping = pickle.load(open('mapping.p', 'rb'))
letters = [_l: 'a', 2: 'b', 3: 'c', 4: 'd', 5: 'e', 6: 'f', 7: 'g', 8: 'h', 9: 'i', 10: 'j',
11: 'k', 12: 'l', 13: 'm', 14: 'n', 15: 'o', 16: 'p', 17: 'q', 18: 'r', 19: 's', 20: 't',
21: 'u', 22: 'v', 23: 'w', 24: 'x', 25: 'y', 26: 'z', 27: '-']

@app.route('/predictCharacter', methods=['GET', 'POST'])
def predictCharacter():
    print("here")
    imgPost = request.get_data(as_text=True)
    print(type(imgPost))
    convertHandwrittenImage(imgPost)
    img = processImage('output.png')
    #img /= 255
    with emnistGraph.as_default():
        characterPrediction = emnistModel.predict(img)
        print(characterPrediction)
        response = {'prediction': str(letters[int(np.argmax(characterPrediction) + 1)]),
                    'confidence': str(max(characterPrediction[0]) * 100)[:6]}
        response = jsonify(response)
        print(response)
        response.headers.add('Access-Control-Allow-Origin', '')
        return response
```

Here is the “/predictCharacter” endpoint which receives a handwritten image and returns a prediction for what letter it the image contains. This uses a model that I have trained using the EMNIST dataset to make predictions about the likelihood of a letter. As you can see both the letter with the highest confidence is returned to the client. A similar endpoint is also available for number recognition and works in exactly the same way.



## Grade Prediction Microservice

```
@app.route('/getAveragePrediction', methods=['POST'])
def getAveragePrediction():
    data = request.get_json()
    dataArray = data["dataArray"]
    predictions = []
    for data in dataArray:
        generator = getDataGeneratorForData(data)
        model = createModel()
        trained_model = trainModel(model, generator)
        prediction = getPrediction(trained_model, data[-3:])
        predictions.append(prediction)
        K.clear_session()

    print(predictions)
    return jsonify({"predictions": predictions})

def getDataGeneratorForData(dataArray):
    series = array(dataArray)
    # reshape
    series = series.reshape((len(series), 1))
    # define generator
    generator = TimeseriesGenerator(series, series, length=3, batch_size=8)
    return generator

def createModel():
    model = Sequential()
    model.add(LSTM(100, activation='relu', input_shape=(3, 1)))
    model.add(Dense(1))
    model.compile(optimizer='adam', loss='mse')
    return model

def trainModel(model, generator):
    model.fit_generator(generator, steps_per_epoch=1, epochs=500, verbose=0)
    return model

def getPrediction(model, data):
    x_input = array(data).reshape((1, 3, 1))
    yhat = model.predict(x_input, verbose=0)
    return int(round(yhat[0][0]))
```

As can be seen this endpoint “/getAveragePrediction” receives a JSON object of a series of averages pertaining to different students. For each array in this json object the data is pre-processed to be used in an LSTM. Then the model is created, and the model is then trained on that student’s data. A prediction is then made using the student’s data and appended to a list of results. This service is useful as all predictions are made based on the user’s own data. Each user has a model created for them trained on their own data and as a result this makes the predictions tailored to them specifically.

These results are returned to the client and the database is subsequently updated with the predicted averages along with the user’s ID.

This microservice is only contacted via a CRONTab scheduled script. It is never contacted as a result of direct input from the User.

## Create Classifier Microservice

This microservice is in charge of creating personalised computer vision models for the teacher and their class. It enables the teacher to create their own custom “find me” game using their own predetermined objects. It receives a JSON object which contains the category name e.g. fruits, foods, body-parts, items list which are all the objects that are being used in the category and the ID of the teacher’s class for which they are making the model as well as the teacher’s user ID.

When the “/trainModel” endpoint is hit with those parameters it immediately returns a response to the user stating that they will see their model in their dashboard once the training has completed, but also the machine learning process is kick started.

```

@app.route('/trainModel', methods=['GET', 'POST'])
def trainModel():
    print("here")
    data = request.get_data(as_text=True)
    data = json.loads(data)
    category_name = data['category']
    search_terms = data['items']
    class_name = data['classname']
    user_id = data["user_id"]
    language = data['language']
    language = languages[language]
    Thread(target=downloadImagesAndTrainClassifier, args=(category_name, search_terms, class_name, user_id, language,)).start()
    return "We will notify you when your model is trained!"

def downloadImagesAndTrainClassifier(category_name, search_terms, class_name, user_id, language):
    print(category_name, search_terms)
    category_name = "_".join(category_name.split(" "))
    if not os.path.exists(category_name + "_images"):
        os.makedirs(category_name + "_images")

    print("querying Api")
    dict = queryApi(search_terms)

    print("getting urls")
    dict = getUrlsForImages(dict)

    print("starting download")
    downloadImages(dict, category_name)

    print("deleting broken images")
    removeBrokenImages(category_name + "_images/")

    print("starting training")
    output_dir = train(category_name + "_images", category_name)

    print("converting model")
    convertModelToTfjs(output_dir, category_name, class_name, language)

    print("transfer files")
    transferFiles(output_dir + "/" + category_name + "_" + class_name + "_saved_model_web/")

    print("sending completion notification")
    sendCompletionData(category_name, class_name, user_id)

```

From the code snippet above you can see that the downloading/training/converting process occurs in a separate thread and this allows a response to be returned immediately to the user. In the thread the “downloadImagesAndTrainClassifier()” function is called which has a number of steps. The first step is to query the Bing Search API with all the items the teacher has specified for the category. The next step is to download all the images that have been found using the search API. Handling exceptions here was essential as there are many issues that can go wrong when trying to download an image. After downloading the images, all broken images needed to be removed in order to prevent the model crashing while training. Once all broken images are removed, the model is trained using transfer learning using MobileNet, this requires the use of a GPU to speed up training time. Once the model has been successfully trained it needs to be converted into a browser-friendly format for client side machine learning using the “convertToTfjs()” function which converts the model into a Tensorflow JS model. Finally, the model is transferred into a specific folder in the main EyeLearn application and the database is updated with the location of the model. The students in the class will now see a new activity added to the activities page and the teacher will see the image configuration on their dashboard.

As this is a computationally expensive task for the application, the user is only allowed to create one model at a time and wait until their first model has finished training before being able to create a new one. To enforce this, I created a “Task” & “User\_Task” table in the database to ensure a user could only create one model at a time.

### 3.3 DEPLOYMENT

EyeLearn is an application that has been designed to be used in a classroom setting and as a result it needs to be able to handle multiple users simultaneously. Thus, the app needs to be deployed correctly and needs to be distributed in order to ensure that the system will not cease when used by a number of users at once.

#### Gunicorn

Flask comes with its own web server which can be called using “app.run()”. While this webserver is very useful when developing or testing, it is not a good choice as a production server as Flask can only handle one thread or request at a time. Also, it is not robust enough and does not have high performance. This can be highlighted using load tests

Gunicorn is a pure Python robust web server capable of handling much more than that of the Flask web server. Gunicorn works by internally handling the calling of your flask code. This is done by having workers ready to handle the requests instead of the sequential one-at-a-time model that the default flask server provides. The end result is the app can handle more requests per second.

```
gunicorn -b localhost:8000 -w 4 run:app
```

The `-b` option tells Gunicorn where to listen for requests. Python applications should generally be run without external access, and then have a very fast web server that is optimised to serve static files accepting all requests from clients. This webserver will be NGINX and will serve static files directly, and forward any requests intended for the application to the internal server. NGINX is used as the public facing web server and will be discussed in detail in the next section.

The `-w` option dictates the number of workers gunicorn will have. Four workers allow the app to handle four clients concurrently, this is an adequate amount as not all of them are constantly requesting content. The number of workers gunicorn can run is RAM and CPU Core dependent. In general applications are I/O bound than CPU bound. As a result, a bottleneck is caused by the disk rather than the processing power on the virtual server. Thus, when a worker is busy with disk operations, another still utilizes the CPU dealing with requests. The Gunicorn documentation states “Gunicorn should only need 4-12 worker processes to handle hundreds or thousands of requests per second.”

Thus, there is a simple formula to calculate the number of workers based on the number of CPU Cores available:

$(2 \text{ Workers} * \text{CPU Cores}) + 1$

While Gunicorn is very simple to set up, running the server from the command-line is not a good solution for a production server. A better solution to this is to have the server running in the background, and have it under constant monitoring, because if for any reason the server crashes and exits, it is essential to make sure a new server is automatically started to take its place so that there is little or no down time. Also, it is important to make sure that if the machine is rebooted, the server runs automatically upon start-up, without having to log in and start things up manually. To facilitate all this, the supervisor package is required.

The supervisor utility uses configuration files that tell it what programs to monitor and how to restart them when necessary. Configuration files must be stored in `/etc/supervisor/conf.d`.

## NGINX

The application now running via Gunicorn is now running privately on port 8000. This app needs to be exposed to the outside world by enabling a public facing webserver on ports 80 and 443. These are ports that have been opened in the server firewall. To make this deployment secure, port 80 forwards all traffic to port 443 which is encrypted.

To have a web site served by NGINX, it requires a configuration file. This needs to be placed in `/etc/nginx/sites-enabled` directory. Here is the configuration for the main EyeLearn web app:

```
server {  
    listen 80;  
    location / {  
        # redirect any requests to the same URL but on https  
        return 301 https://$host$request_uri;  
    }  
}  
  
server {  
    # listen on port 443 (https)  
    listen 443 ssl;  
    server_name eyelearn.club www.eyelearn.club;  
    # location of the self-signed SSL certificate  
    ssl_certificate /etc/letsencrypt/live/eyelearn.club/fullchain.pem;  
    ssl_certificate_key /etc/letsencrypt/live/eyelearn.club/privkey.pem;  
    # write access and error logs to /var/log  
    access_log /var/log/eyelearn_access.log;  
    error_log /var/log/eyelearn_error.log;
```

```

location / {

    # forward application requests to the gunicorn server

    proxy_pass http://localhost:8000;

    proxy_redirect off;

    proxy_set_header Host $host;

    proxy_set_header X-Real-IP $remote_addr;

    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;

}

location /static {

    # handle static files directly, without forwarding to the application

    alias /home/rsa-key-20190215/eyelearn-frontend/app/static;

    expires 30d;

}

}

```

This configuration ensures that the application is deployed and now the application can be visited using the public facing IP address of the server or via the domain.

Handling large numbers of connections with very little CPU and memory cost is what asynchronous servers like Nginx are good at. They aren't affected in the same negative manner by slow clients because they are adept at handling large numbers of clients simultaneously. In NGINX's case, running on modern hardware it can handle tens of thousands of connections at once.

NGINX in front of Gunicorn is a great combination. Nginx handles communications with clients and doesn't suffer a penalty for handling slow clients. It sends requests to the backend as fast as the backend can handle those requests, enabling the backend to be as efficient with server resources as possible. The backend returns the result as soon as it calculates it, and Nginx buffers that response to feed it to slow clients at their own pace. Meanwhile, the backend can move on to handling another request even as the slow client is still receiving the result.

NGINX is also the best service for serving static files. EyeLearn uses multiple types of media and thus it benefits from the fantastic performance of NGINX. It also allows for caching which can boost performance.

Compared to Apache, four times the number of concurrent connections are handled. NGINX is the most efficient and light-weight web server today. NGINX can handle more traffic concurrently while having less memory usage.

NGINX is an answer to the C10K problem which basically means: “how to get one web server to handle 10,000 connections, given the limitations of the operating systems?”. It can be easily scaled on minimal hardware, focuses on core web server and proxy features. This was ideal for the developer due to the limited funds at his disposal.

## **SSL/HTTPS**

The security for HTTP is implemented through Transport Layer Security (TLS) protocol. TLS is the standard way to make any network communication channel secure. When a client establishes a connection with the server, the server will respond with an SSL certificate if an encrypted connection is required. This certificate identifies the server as it contains a server name and a domain. This certificate is signed by a certificate authority (CA). If the client trusts the CA the client can be certain that the server it connected to is legitimate.

When the client verifies the certificate, an encryption key is created for communication with the server. The Diffie-Hellman key exchange occurs between the client and the server and so all the traffic between them is encrypted.

The CA used for this project was Let's Encrypt using the domain “eyelearn.club”. As a result, the application can now be accessed via [www.eyelearn.club](http://www.eyelearn.club) and it is run on HTTPS. All the microservices are also hosted over HTTPS. This was essential for the project as it is very important to make sure the client is secure when recording speech in the browser.

The use of less SSL Certificates and HTTPS ensures:

- No one can decipher a user's information.
- Better search ranking
- Improved user trust

## **CRONTAB Scheduling**

Crontab Scheduling is used to collect daily average data for every student and class using EyeLearn. This data is collected and then passed to the Grade Prediction Microservice which in turn generates a grade prediction for that user for the following day.

This script is scheduled to run every day at the same time. Once this data is collected it is then stored in a table in the MYSQL database to be used by the analytics pages.

## **4. PROBLEMS SOLVED**

Given the complexity and span of this project, there were many problem-solving tasks involved in its creation.

### **Learning a new framework**

As I had very little experience in Python Web App development I had to learn a new framework, Flask. A Python based framework would suit best as it would allow for use of the Keras library for machine learning.

**Resolution:** To overcome this task multiple tutorials were followed. A tutorial located here: <https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i-hello-world> proved quite useful. Also, an Udemy course was undertaken to ensure I understood the best practices when it came to Flask and creating a web app.

### **Self-teaching Machine Learning**

While previous modules in DCU have covered aspects of Machine Learning, none covered computer vision in depth. It was important to research this effectively as it was a huge aspect of this application.

**Resolution:** This involved hours of tutorials and videos from online. Books from Adrien Rosenstock from PyImageSearch.com proved useful. While much of the content was quite complex I was able to pick out what was relevant for the project. Videos online from Siraj Raval were used as an introduction into the topic also.

### **Validating Convolutional Neural Networks**

As the use of Convolutional Neural Networks was new to me, I had never tested or validated one before. The CNNs required decent accuracy in order to be used by the application, if not inaccurate predictions would be returned for a user's input.

**Resolution:** To validate a Convolutional Neural Network, a K-Fold Cross Validation is used. These are computationally expensive as it requires training multiple networks using different testing and training splits while tweaking parameters to see which yields the best results. This required hiring out a server with GPU access from Google Cloud, as I did not have access to a GPU.

### **Building an App that could handle multiple users concurrently**

As this app is designed to be used in a classroom set up, it needed to be able handle multiple users and classes concurrently. If a user was blocked due to another user, the app would lose appeal.

**Resolution:** While this took much research and investigation, the use of NGINX and Gunicorn was used to ensure the app could handle multiple users. Containerisation was an option but after further research, the potential for data leaks ruled them out.

### **Building a scalable app**

To make a fully-fledged language app would require much more time than 9 months. Thus, I wanted to build an app that could easily be expanded after the college year.

**Resolution:** This led me to build the application around microservices. This architecture allowed for decoupled services which could be written in different languages to coexist in harmony with other services. This made the app far more scalable than a monolithic solution. The use of microservices allows me to add new components to the system painlessly or scale services separately from one another.

### **Ensure the app is hosted securely over HTTPS**

It was important to ensure that all traffic between the client and server was encrypted to ensure the security of the user. The use of HTTPS would also create trust between the user and the application.

**Resolution:** This involved purchasing a domain “eyelearn.club” and getting an SSL Certificate from a Certificate Authority, Let’s Encrypt. Also, it required a Linux firewall “ufw” to allow traffic only enter via certain ports (80 and 443). This also had to be configured using NGINX as shown above in the configuration.

### Creating Accessible and Reusable Microservices

To ensure the microservices were available for reuse by other products and applications. There was no point in using a microservices architecture if the microservices themselves were not going to be able to be used by any other applications.

**Resolution:** The microservices are all hosted over HTTPS using a variation of the “eyelearn.club” domain name e.g. “imagenet.eyelearn.club” is the address for the object translation microservice. They are accessible via an API gateway and all have their own endpoints. Also, the data that needs to be passed to the microservices is not complex ensuring they can be reused. To translate an object, an image and a language need to be sent to <https://imagenet.eyelearn.club/translateObject>

```
r = requests.post("https://imagenet.eyelearn.club/translateObject", files={'image1': open('foo.png', 'rb')}, data={'language': language})
```

### Real Time Classification/Client-Side Machine Learning

When the user is trying to locate everyday objects in the “Find Me” game and drawing doodles in “Doodle” to make the games run more smoothly and quickly client-side machine learning was required. This would ensure classifications would be performed in real time. In the find me game when a user points the camera at something new, new classifications are made instantly. Similarly, when the user lifts their finger in the doodle game, a classification is made instantly.

**Resolution:** To make this possible it required a library that is in its infancy, Tensorflow Js. This enables complex machine learning models to be stored on the client’s side without hindering load times that much. This library enables for real-time classification and means that user input did not need to be passed to the back end for those activities.

### Tracking Student/Class Performance

Creating an educational tool without being able to track a student’s or class’ performance would be futile. The user would not know how they are progressing in the different activities.

**Resolution:** Provide data analytics for both the student and teachers so that they can track their performance and see areas where they need to improve.

### Tracking Student Mistakes and Problem Words

The use of this would enable the student and teacher to track problem words and also any common mistakes that students make. Also, the ability to be able to track if a student commonly mixes up letters and numbers such as “b & p” and “6 & 9” allows the teacher to track a symptom of dyslexia.



**Resolution:** This involved monitoring all student answers and having a table in the MYSQL database for tracking student mistakes.

### **Creating powerful image classifiers with limited data.**

As I did not have multiple datasets at my disposal for the category classification CNNs I needed an efficient way to make accurate classifiers without much data.

**Resolution:** I used a technique called Transfer Learning which allows you to train a pretrained model and your own data and yield high accuracy with little data. Much of the techniques described here were learnt from the Keras blog.

### **Improving the Speed and Size of very large models**

At the beginning of development there was a “Category Classification” Microservice was available that classified animals, vehicles and sports. These models were all in excess of 500mb and required a server with GPU access and 16GB of memory to run, even then inference was quite slow and made some activities unplayable as a result. These models had been created using Transfer Learning using Inception.

#### **Resolution:**

Using Tensorflow JS converter and Transfer Learning with MobileNet these models were now created to work on the client side, allowing for real-time classification and really small model sizes that could be loaded as the activity is loaded.

### **Training a large Model with a huge dataset without access to a GPU**

Without access to a GPU on my own machine, training large networks was going to be a difficult. The use of my own machine would have taken days and GPU servers are very expensive and so I needed a way to train a model in an efficient way without costing a large sum.

**Resolution:** Google Colab allow free access to a top of the range NVIDIA GPU for a maximum of ten hours at a time. To access it, it is required to write a Python Notebook that can be uploaded to their website. In a python notebook you can execute pieces of code piece by piece and this enabled me to train a large model using the quick draw dataset. It also allows you to download any created files.

### **Giving the teacher full control and enabling to create their own games/computer vision models.**

Allowing the teacher to create their own computer vision games as simply as possible whilst they have no experience in machine learning or convolutional neural networks was a tough problem to solve. The teacher needed to be able to create machine learning model that had decent accuracy but could also be used in the browser without having the application blocked. It was essential that they were not overwhelmed with a convoluted task in order to create their own model and that it required little time and effort. Also, the game needed to be added to the teacher’s students’ activity pages seamlessly, as well as giving access to both teacher and students to images of the objects that the classifier created should recognise.

#### **Resolution:**

To solve all these problems there were several resolutions. To make the process simple and to prevent the teacher being overwhelmed when creating a computer vision game, they can create a model from their dashboard via a modal that requires three inputs from them:

1. The class they want to create the game for
2. The category of the classifier e.g. fruit, food, clothes
3. The objects they want the students to find e.g. for fruit, kiwi, grapes, tomato, orange etc.

Also, the use of an image search API would ensure the teacher would not have to provide their own images.

To ensure the model has relatively good accuracy, I needed to employ transfer learning to yield high accuracy with limited amounts of data.

To ensure the creation of a classifier was non-blocking and did not impede the performance of the main EyeLearn Application, a microservice was created to manage the creation of a new classifier. This microservice was hosted on a server with GPU access. When the microservice is contacted a response is sent back immediately to the UI and the training and downloading process is triggered in a separate thread.

To ensure the model could be used in the browser and would work like the other find me games, transfer learning was performed using MobileNet which has much less parameters than other pre-trained models such as Inception and VGG. Also, the use of the “Tensorflow Js Converter” enables the model to be shrunk down so it can be used client side and allow for real-time classification.

## 5. RESULTS

### 5.1 RESEARCH CONCLUSIONS

---

#### MACHINE LEARNING CAN BE USED AS AN AID FOR LEARNING A NEW LANGUAGE

From completing this project, it is evident that Machine learning will play a huge part in second language learning in the future. Machine Learning is a field that is expanding rapidly and that is improving all the time. Advances in the likes of Speech Recognition and Machine Translation will play a huge part in Machine learning’s ability to teach a second language.

While much of what machine learning can teach is vocabulary, even grammar can be taught when using POS Tagging in conjunction with speech recognition or simple translation. This can give users a clear indication of what has been translated into what.

The use of POS Tagging here makes the translation mapping quite easy to see and understand.

John -> John, likes -> aime, the -> la, bleue -> blue, house -> maison, at -> au, end of -> bout de, the -> la, street -> rue

John	likes	the	blue	house	at	the	end	of	the	street	.
John	aime	la	maison	bleue	au	bout	de	la	rue	.	

Adjective  
 Adverb  
 Conjunction  
 Determiner  
 Noun  
 Number  
 Preposition  
 Pronoun  
 Verb

The use of computer vision in this project allows the user to translate the world around them. The user is now able to get translation for objects they use every day using this translation and are then able to test their knowledge on these newly learned words using a number of activities. Again, this highlights the benefits of using machine learning as an aid to learn a second language. It also makes learning the language fun and interactive as the users get to use their device camera to translate the world. Also, classifiers can be trained to classify certain categories to teach words. This allows vocabulary to be taught visually and students can make associations with different images rather than rote-learning vocabulary.




image.jpg
 Change
Clear

Upload Image

Translation	English	Confidence
hippopotame	hippopotamus	95.64%
wombat	wombat	0.11%
loutre	otter	0.09%
Lion de mer	sea lion	0.06%
tirelire	piggy bank	0.06%

The ability to perform speech recognition and text-to-speech in multiple languages further enables students to improve their pronunciation and listening skills. This has been made possible with advances in Machine Learning being able to synthesize new voices and accents. While understanding vocabulary and grammar is important when learning a language, being able to comprehend speech and having the correct pronunciation is just as important.

## 6. FUTURE WORK

The developer would like to further the research of the benefits and pitfalls of using Machine Learning in second language learning. Understanding the benefits and pitfalls of its use in second language learning would enable the developer to improve the app by leveraging the benefits and avoiding the pitfalls.

One future application for Machine Learning in second language learning would be text generation using a Bidirectional LSTM. This would enable a user to input an input sentence and then the application would generate the next. While work has been performed on this using English corpuses nothing has been done as a way of generating coherent foreign text. If implemented correctly this could be a huge benefit to second language learning.

---

## MICROSERVICE REUSE

The microservices created by the developer are not designed specifically for the EyeLearn Application. As a result, they could be used in any application. Developing a native app which revolves solely around different means of translation is an area the developer would like to explore. This would involve creating an installable app that would allow users to translate their speech, translate objects and translate text.

The POS Tagger and Translator Microservice could be used as the baseline for a new application that could take any input string in a specified language and translation language and generate the translation whilst also providing a breakdown of the translation word by word using POS tags so that the user could see what words match up.

---

## IMPROVING MODEL ACCURACY AND REMOVING ALL BACKEND MACHINE LEARNING MODELS

A goal for the future of the application would be to improve model accuracy for all models. Also, the ability to convert the backend models to client-side models using Tensorflow JS would also improve the app as a whole and make inference far quicker. This was not possible for some models as they ended up being too large and were too slow to load for the client.

---

## USING IRISH

Currently there is no available API that allows speech synthesis or text-to-speech for the Irish Language. As a result, the Irish Language was not used in EyeLearn as the text-to-speech aspect of the app is so important. It would be easy to add in but, if the activities are silent especially during the “Doodle” or “Find Me” Games, it loses much of its appeal.

While there is an Irish Text-To-Speech system online – [abair.ie](http://abair.ie), developed in Trinity College Dublin, it is currently not available to be incorporated into third party apps.

## 7. APPENDICES

### 7.1 REFERENCE LIST

[S1] Sha L, Looi CK, Chen W, Zhang BH. Understanding mobile learning from the perspective of self-regulated learning. *Journal of Computer Assisted Learning*. 2012 Aug 1;28(4):366-78.

- [S2] Young K. Teachers' attitudes to using iPads or tablet computers; Implications for developing new skills, pedagogies and school-provided support. *TechTrends*. 2016 Mar 1;60(2):183-9.
- [S3] Clarke L, Abbott L. Young pupils', their teacher's and classroom assistants' experiences of iPads in a Northern Ireland school: "Four and five years old, who would have thought they could do that?". *British Journal of Educational Technology*. 2016 Nov;47(6):1051-64.
- [S4] Clarke B, Svanaes S, Zimmermann S. One-to-one tablets in secondary schools: an evaluation study. *Tablets for schools*. 2013 Jan.
- [S5] Rikala J, Vesisenaho M, Mylläri J. Actual and potential pedagogical use of tablets in schools. *Human technology: an interdisciplinary journal on humans in ICT environments*. 2013.
- [S6] Kongsgården P, Krumsvik RJ. Use of tablets in primary and secondary school-a case study. *Nordic Journal of Digital Literacy*. 2016 Jan 1;11(04):248-70.
- [S7] Golonka EM, Bowles AR, Frank VM, Richardson DL, Freynik S. Technologies for foreign language learning: a review of technology types and their effectiveness. *Computer assisted language learning*. 2014 Feb 1;27(1):70-105.
- [S8] Altun M. The integration of technology into foreign language teaching. *International Journal on New Trends in Education and Their Implications*. 2015 Jan 1;6(1):22-7.
- [S9] Huang J, Rathod V, Sun C, Zhu M, Korattikara A, Fathi A, Fischer I, Wojna Z, Song Y, Guadarrama S, Murphy K. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition 2017* (pp. 7310-7311).
- [S10] Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*. 2017 Apr 17.
- [S11] Dragoni N, Giallorenzo S, Lafuente AL, Mazzara M, Montesi F, Mustafin R, Safina L. Microservices: yesterday, today, and tomorrow. In *Present and Ulterior Software Engineering 2017* (pp. 195-216). Springer, Cham.
- [S12] Taibi D, Lenarduzzi V, Pahl C, Janes A. Microservices in agile software development: a workshop-based study into issues, advantages, and disadvantages. In *Proceedings of the XP2017 Scientific Workshops 2017 May 22* (p. 23). ACM.
- [S13] Esposito C, Castiglione A, Choo KK (2016) Challenges in delivering software in the cloud as microservices. *IEEE Cloud Computing* 3(5): 10-4
- [S14] Mazzara M, Dragoni N, Bucchiarone A, Giarretta A, Larsen ST, Dustdar S (2018) Micro-services: Migration of a Mission Critical System. *IEEE Transactions on Services Computing*: 99
- [S15] Dragoni N, Lanese I, Larsen ST, Mazzara M, Mustafin R, Safina L (2017) Microservices: How to make your application scale. In: *International Andrei Ershov Memorial Conference on Perspectives of System Informatics*, Springer, Moscow, Russia, pp. 95-104
- [S15] Salah T, Zemerly MJ, Yeun CY, Al-Qutayri M, Al-Hammadi Y (2016) The evolution of distributed systems towards microservices architecture. In: *11th International Conference*
- [S16] Carroll J, Minnen G, Pearce D, Canning Y, Devlin S, Tait J. Simplifying text for language-impaired readers. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics 1999*.

[S17] Granger S. Learner corpora in foreign language education. Encyclopaedia of language and education. 2008:1427-41.

[S18] National Forum for the Enhancement of Teaching and Learning in Higher Education, "Using Learning Analytics to Support the Enhancement of Teaching and Learning in Higher Education," in teachingandlearning.ie, Published February 28, 2017, Last Accessed March 15, 2019.

[S19] Khalil M, Ebner M. A STEM MOOC for school children—What does learning analytics tell us?. In 2015 International Conference on Interactive Collaborative Learning (ICL) 2015 Sep 20 (pp. 1217-1221). IEEE.

[S20] Vanthienen, Jan & De Witte, Kristof. (2017). Data Analytics Applications in Education.

## 7.2 USEFUL LINKS

- <https://dzone.com/articles/design-patterns-for-microservices>
- <https://dzone.com/articles/microservice-design-patterns>
- <https://www.digitalocean.com/community/tutorials/how-to-deploy-python-wsgi-apps-using-gunicorn-http-server-behind-nginx>
- <https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-xvii-deployment-on-linux>
- <https://medium.com/dreamcommerce/apache-vs-nginx-what-are-the-biggest-advantages-380ad8ebd4a4>
- <https://blog.coolicehost.com/ten-great-advantages-of-nginx/>