Technological Feasibility Analysis

Team name:
LangLens

Faculty Mentor
Italo Santos

Sponsor
Dr. Okim Kang

Team Members
Stefan Mihailovic
Daniel Navarette
Martin Brian Ruiz
Sami Tanquary
Kyle Young

Overview

This is the Technological Feasibility Analysis. The purpose of this document is to identify the key technological challenges and major design decision, explore the different approaches or existing products that could address those challenges and make a preliminary decision on which approach/product the team is going to choose to solve each of these challenges.

November 10, 2022

Contents

1 Introduction

Research[1] shows that studying a foreign language enhances many different cerebral skills, such as memory, listening skills, your analytical abilities, as well as bolstering your problem-solving abilities and working with abstract concepts in general. Imagine you are learning a new language. Traditionally, the process consists of reading, memorizing, and writing, and in more recent times, online systems with direct translations and pronunciations. However, imagine instead of just reading, and trying to memorize a word, you could see it, and interact with it. You could select it from a virtual shelf and hear its name in the language you are studying, or even better, you could point at any object you can see in your surroundings, and immediately find out what it is in the foreign language. With the emergence of augmented reality (AR) over the past decade, immersive and interactive experiences like the one described above are entirely possible.

Our sponsor, Dr. Okim Kang, and her team specialize in linguistics, more specifically working in Computer-Assisted Language Learning (CALL). A glaring problem that they have encountered is that there is is a lack of tools that utilize AR technologies in the world of language learning. There have been a few attempts to fill this void, but none of them fully utilized things such as object detection, text recognition, and most importantly AR under the same umbrella.

Data shows how widely, and easily accessible smartphones are, with over 83%[2] of the world's population owning one. With the majority of the population carrying a powerful computer in their pockets on a daily basis, it's no question that a quick Google search is all you need for an on-the-go translation, but why not solidify the knowledge with a tool that assists in learning and maintaining a new language by immediately being able to identify and interact with an object or word that's sitting right in front of you. Thus, our solution to the lack of AR language learning systems is to develop a web-based mobile app, specifically to target the widest audience possible, and make it as easily accessible as possible. Our app will open the user camera in live mode, with a selection screen between two different modes, an object detection mode, and text recognition mode. The user will also be able to choose their target language for translation, for example, Spanish, French, or English to name a few. Upon selection, the app will be able to either identify the objects, or scan the words, with very high precision. Eventually, we planned to offer the user an ability to save the object, or the word, into a personal library or "PLAYlist" to allow users the opportunity to review what they have scanned and practice the language even more. Each new word will also have an external popup link that will lead them to a learning page to learn more about the word, see a graphical representation of it, as well as hear the correct pronunciation.

[1] Klimova B. Learning a Foreign Language: A Review on Recent Findings About Its Effect on the Enhancement of Cognitive Functions Among Healthy Older Individuals. Front Hum Neurosci. 2018 Jul 30;12:305. doi: 10.3389/fnhum.2018.00305. PMID: 30104969; PMCID: PMC6077199.

[2] From research conducted by bankmycell over at https://www.bankmycell.com/blog/how-many-phones-are-in-the-world

2 Technological Challenges

As we are at an early stage in our project, we are in the process of researching the technical challenges, as well as finding potential alternatives that will be useful in providing solutions to our problems. In this Technological Feasibility Analysis, we will first tackle the most important technical challenges that we will need to solve to guarantee the success of our project. Afterwards we will individually dissect every single challenge, from describing its proposed solution and alternatives, to explaining our justification behind each. Afterwards we will move onto technological integration, where we will explain the big picture, and how all the individual solutions will come together to make a functioning piece of software. Finally, we will summarize our work in a concluding section.

The challenges that we have considered for our project are:

- We will need an efficient and accurate way of detecting/scanning objects.
- We will need an efficient and accurate way of detecting/scanning text.
- We will need an easily understandable, communicative, and secure frontend web framework.
- We will need a way to represent a detected object/word with an image.
- We will need a reliable backend web database that is capable of securely storing and retrieving text/object recognition data.

3 Technology Analysis

3.1 Object Detection

The main feature behind LangLens AR Language Learning application is object detection in real time. Once the app is booted up, users will be able to turn on the camera on the back of their phones, and scan any object in their surroundings, with a label with the object's name appearing on top of the object through AR. Essentially the app will need an efficient way to scan and detect any objects presented to it.

3.1.1 | Desired Characteristics

1. **Licensing cost:** Ideally the license for an object detection library would be free. Considering that the final project will be distributed for free to students, teachers, researchers, etc. the project itself would ideally not have to use a library that relies on a paid license.
2. **Speed/Performance:** The project should be able to detect objects in real time, with a maximum wait time of a second or two. It is crucial that the project.
3. **Accuracy:** Since the accuracy of the object detection is non-negotiable as the app will be used for language learning, ideally the scanned object is properly identified at least 80% of the time.
4. **Ease of implementation:** The object detection library should be compatible with all the other technologies that will be used. It should be properly documented and ready to be implemented straight out of the box.

3.1.2 | Alternatives

The following options describe the numerous alternatives for object detection that could potentially be used within LangLens:

- **YOLOv5 & Python:** YOLOv5 (You Only Look Once) is a family of machine learning algorithms used specifically for object detection. YOLO has been around since 2016, but YOLOv5 specifically has been released in 2020. It is one of the most popular algorithms used for object detection due to its speed and accuracy.
- **Visionlib in Unity and C#:** Visionlib is enterprise level object detection in augmented reality. It is mostly used in industries such as construction and car manufacturing. Its multi-platform capabilities make it a popular choice in many AR applications. Another unique feature it offers is the ability to not only detect, but also track objects independently of each other.
- **Mediapipe Objectron - Python:** This is Google's augmented reality object tracking algorithm. It is a mobile solution that extends Objectron's previous 2D object detection capabilities and brings them into the 3D world. It utilizes a Machine Learning model trained on Objectron's dataset to detect objects. It relies on AR- Core and ARKit to function on mobile devices.

3.1.3 | Analysis

The following section provides an individual analysis of each of the alternatives listed in section 3.1.2 and their respective scores out of 10 following the conditions of satisfaction metric described in section 3.1.1.

- **YOLOv5 PythonYOLOv5 (You Only Look Once):**
  - **Licensing Cost:** YOLOv5 is an open source and completely free to use, so it perfectly fits our licensing cost requirements. It receives a score of 10.
  - **Speed/Performance:** YOLOv5: YOLOv5 is ranked among the fastest object detection algorithms. Based upon reviews of other developers, YOLOv5 takes around 0.05 seconds to successfully detect and recognize an object, with it taking slightly longer the more objects there are on the screen. While this is fast, and it satisfies our wants for the inference speed, it is not the fastest, and thus it earns a score of 8.

  - **Accuracy:** YOLOv5 developers claim that after running the algorithm through multiple test models, it consistently outputs 99.3% accuracy rate, which not only satisfies our requirements, but performs even better than we could hope for, earning a score of 9.
  - **Ease of Implementation:** YOLOv5 is relatively simple to implement, all it needs is to be downloaded and imported into the python project. After that it is ready to go right out of the box. Because of this YOLOv5 receives a score of 10.
  - **Overall Rating:** YOLOv5 receives an average score of 9.25.

- **Visionlib in Unity and CVisionlib:**
  - **Licensing Cost:** Visionlib requires a commercial license which is said to be very expensive. For the exact quote, the company requires in-depth documentation about the project it would be used for, the size of the development team, etc. They have however offered LangLens a free trial license due to the nature of our capstone. With that being said, our project is planned to be distributed to learners all around the world, and an expensive license is not desired. We give Visionlib a score of 4 for licensing cost.
  - **Speed/Performance:** According to user reviews, Visionlib is one of the absolute fastest object detection technologies, surpassing even YOLOv5. Many users have left reviews stating that the inference time was less than 0.01 seconds. Visionlib receives a perfect 10 in this category.
  - **Accuracy:** Visionlib satisfies our minimum requirement of 80% accuracy rate. We could not find exact data to pinpoint its accuracy, but according to user reviews it seems to be above our required amount. It received a score of 8.
  - **Ease of Implementation:** Visionlib is on a slightly more complicated end when it comes to implementation. The reason for this is because it comes as a Unity plug-in, a platform we would have to use specifically for Visionlib. What this means for our web application is that we would need to create extra layers within the architecture to make it functional. It received a score of 6.
  - **Overall Rating:** Overall rating for Visionlib is 7.

- **Mediapipe Objectron - Python:**
  - **Licensing Cost:** Mediapipe Objectron, similarly to YOLOv5, is open source and free, and perfectly fits our needs for this project. It receives a score of 10.
  - **Speed/Performance:** Mediapipe Objectron features fast inference speed, taking about 0.6 to 0.8 seconds according to multiple user reviews. It is not the fastest, but it is more than satisfactory for the needs of this project. We give it a rating of 8.
  - **Accuracy:** Mediapipe Objectron, while fairly accurate, seems to be inconsistent. There are plenty of user reports where the algorithm will feature a 100% accuracy for a period of time, only to go through a downward spiral with an abysmal accuracy of about 60%. Because of this we give it a score of 6, we need consistently high accuracy.
  - **Ease of Implementation:** This family of algorithms is not too difficult to implement, it's a Python API, so it simply needs to be imported into the project, however unlike YOLOv5 it does require some tinkering beforehand for it to work and produce desired output. We give it a score of 8.
  - **Overall Rating:** Mediapipe Objectron receives an overall score of 8.

3.1.4 | Chosen Approach

After reviewing several alternatives for the best object detection software, the following table compares the overall ratings of the alternatives analyzed in section 3.1.3:

Object Detection Rankings

	Objectron	**YOLOv5**	**Visionlib**
Licensing	10	10	4
Speed	8	8	10
Accuracy	6	9	8
Ease of Implementation	8	10	6
AVERAGE	**8**	**9.25**	**7**

Table 3.1: Table showcasing the comparisons between the alternatives and the desired characteristics on a scale from 0-10.

According to the Table 3.1, YOLOv5 and Objectron are the clear favorites when it comes to licensing cost, as both are free to use. Visionlib offers a free trial period to students, but for a long-term project would require our client to purchase a license.

As for the speed, most of the technologies had similar performances. They were all able to detect the objects nearly instantly on the screen.

Accuracy, similar to speed, was very identical among the alternatives. None of them were able to detect objects with 100% accuracy. However, for the purposes of this project, all offer a respectably high level of accuracy that will be sufficient.

Finally, the ease of implementation goes to YOLOv5 as the top contenders, as the other two options require a fair amount of work to just implement into the project, and then even more effort to make them work as intended. YOLOv5 is designed for ease of implementation, and as such is ready to get to work straight out of the box.

Based on the scores and the analysis, there is a clear-cut winner that satisfies all of the desired characteristics at a sufficient enough level. While YOLOv5's accuracy and speed are not bleeding edge, those two areas showed struggle from the other options as well, however its advantage and preference over the other technologies comes from the fact that it's very easy to implement, as well that it costs no money to do so. YOLOv5 is the chosen approach for handling object detection as it provides a free, open source, and easy to implement library that is both fast and accurate.

3.1.5 | Proving Feasibility

To consolidate YOLOv5 as the correct chosen approach, and to prove its feasibility, we will develop a small prototype with a single feature that will exclusively handle object detection. We will use it to verify the speed and the accuracy of YOLOv5. This will also help us confirm the ease of implementation, as we will get hands-on experience in implementing it.

3.2 | Optical Character Recognition (OCR) Engine

As one of the main features that the LangLens application is centered around, an OCR engine, or more simply defined, a text recognition engine, is essential in order for the user to be able to scan/detect single words for language learning. For the purposes of this project, any language of text will need to be able to be scanned using the user's back camera on their mobile device. A visual bounding box will highlight the user's desired text and the OCR will need to be able to translate the detected text before it can store the recognition data in a designated database. In summary, the application will need an efficient OCR engine to detect any language of text upon scanning, translate it, and implement AR for visualization purposes in the UI.

3.2.1 | Desired Characteristics

The following characteristics will be used as the conditions of satisfaction metric when evaluating alternative solutions:

1. **Speed/Performance:** The processing speed at which the application should be able to recognize text needs to be no more than 10 seconds. This is to ensure that the OCR software is efficient but still has enough time to process the text and display an accurate result. Ideally, the OCR software should be optimizable for the development of faster speeds.

2. **Accuracy:** Since this is a learning application, the result of the text scanned should be no less than 65% accurate with each run. This 35% margin of error is a large but necessary buffer as reliance on the OCR to be 100% accurate is unrealistic.

3. **Ease of implementation:** The OCR should include extensive documentation for developer use to aid in the understanding of how to fully utilize the software within a mobile web application to its fullest potential.

4. **Library Compatibility:** The OCR needs to be compatible with any type of language library that is given to it as a dependency, and it should not be reliant on only limited built-in libraries. The software should also have the ability to integrate multiple different language libraries in order for the user to be able to choose a target language that the OCR can switch between.

5. **Cross Platform Compatibility:** For the main functionality of this project, the OCR needs to be able to function seamlessly on all browser types which means that the software should have cross-platform integration available.

3.2.2 | Alternatives

The following options describe the numerous alternatives for an OCR engine that are compatible cross-platform via mobile device:

- **Tesseract Open-Source OCR Engine:** Tesseract is an open-source optical character recognition engine that was developed by HP between 1984 and 1994. It became an open-source platform in 2006 and has been sponsored by Google ever since. According to Tesseract's official documentation, the engine can be used via CLI or by an API to extract text from images. The engine comes prepackaged with a wide variety of trained data files (tessdata) of language libraries which is perfect for the purposes of this project. It works most efficiently when combined with OpenCV's EAST (Efficient and Accurate Scene Text Detector) contemporary deep learning model. Many software reviews recommend Tesseract OCR as their first choice for an OCR engine due to its functional modularity and available wrappers for various programming languages such as Python, Java, JavaScript, Objective- C, and more.

- **Google Cloud Vision API + OpenCV for OCR:** Google Cloud Vision API is a platform that allows developers to easily integrate computer vision features and tools within any application. It comes with access to Google's massive language data set. When utilized with OpenCV (an open-source computer vision library), the functionality of the overall OCR is further improved with the added feature of text recognition via OpenCV's live camera streaming. The Cloud Vision API is, however, a paid service. This is the only con regarding Cloud Vision as LangLens would like to be fully operational at zero cost, however; after taking into consideration that this is a paid service, Google Cloud Vision API is rated as, by far, the best OCR to date with a text accuracy rating of 98%.

- **Amazon Rekognition API + Amazon Web Services SDK for OCR:** Amazon's Rekognition API is another platform that offers computer vision tools. Rekognition is both an object and text detection service and utilizes deep neural net- work models to detect and label images. In order to use Rekognition, the AWS

SDK is necessary for integration. The AWS SDK is a software development kit developed by Amazon for Amazon products offered in various programming languages such as JavaScript and Python which will work perfectly for the development of this web application. The only foreseeable con for using Amazon Rekognition API is that it is also a paid service. Taking that into consideration, Amazon Rekognition API still comes highly recommended by developers for its reliable text recognition accuracy.

3.2.3 | Analysis

The following section provides an individual analysis of each of the alternatives listed in section 3.2.2 and their respective scores out of 10 following the conditions of satisfaction metric described in section 3.2.1.

- **Tesseract Open-Source OCR Engine**
  - **Speed/Performance:** After reviewing multiple software analyses regarding Tesseract, the general consensus from developers is that Tesseract's CPU processing time gets significantly slower and CPU usage is significantly higher when scanning blurry images and executing multiple processes in parallel. However, Tesseract can be optimized with OpenCV's EAST to process im- ages at much higher FPS and therefore receives a rating of **6 out of 10** for speed/performance.
  - **Accuracy:** In terms of accuracy, Tesseract has been generally rated with an accuracy above 95% when excluding handwritten documentation. Since handwriting is not as important of a factor as printed text for the purposes of this project, Tesseract receives a rating of **7 out of 10** for OCR accuracy.
  - **Ease of implementation:** Tesseract provides extensive documentation for developer use and thoroughly provides tutorials and explanations for every single possible use case that it offers. Therefore, Tesseract receives a rating of **10 out of 10** for ease of implementation.

    Library Compatibility: Tesseract comes prepackaged with a large quantity of trained data files (tessdata) which include over 100 different languages for translation purposes. For this reason, it is unnecessary for Tesseract to be externally compatible with other language libraries as it covers all necessary translation features. Therefore, Tesseract receives a rating of **10 out of 10** for library compatibility.

- **Cross Platform Compatibility:** Tesseract offers cross platform compatibility and a JavaScript wrapper which allows for implementation on all browser types. For these reasons, Tesseract receives a rating of **10 out of 10** for cross platform compatibility.
- **Overall Rating:** Tesseract receives an average rating of **8.6 out of 10**.

- **Google Cloud Vision API + OpenCV for OCR**
  - **Speed/Performance:** After reviewing multiple software analyses regarding Google Cloud Vision API, the general consensus from developers is that the processing speed image by image averages around 5 to 15 seconds. Because Cloud Vision + OpenCV is roughly the same speed as Tesseract + OpenCV, it is rated **6 out of 10** for speed/performance.
  - **Accuracy:** In terms of accuracy, Google Cloud Vision API is rated across the board by hundreds of developers as the most accurate OCR engine and has been assessed to have a 98% accuracy rate. Therefore, Google Cloud Vision API is rated **9 out of 10** for OCR accuracy.
  - **Ease of implementation:** Google Cloud Vision API provides extensive documentation for developer use as well detailed tutorials by Google themselves on step-by-step how to implement the OCR engine into any application. Therefore, Google Cloud Vision API receives a rating of **10 out of 10** for ease of implementation.
  - **Library Compatibility:** Google Cloud Vision API is dependent on Google Client libraries, however; this is not an issue as Google themselves provides the necessary language libraries needed for text translation and therefore external libraries will not need to be provided for translation purposes. For this reason, Google Cloud Vision API receives a rating of **10 out of 10** for library compatibility.
  - **Cross Platform Compatibility:** Google Cloud Vision API is compatible with any browser so long as it has JavaScript enabled, and since JavaScript is a cross platform, object-oriented scripting language, the API will work on all browser types. Google Cloud Vision API therefore receives a rating of **10 out of 10** for cross platform compatibility.
  - **Overall Rating:** Google Cloud Vision API receives an average rating of 9 out of 10, however; because this is a paid service the rating is being deducted 2 points for an overall rating of **7 out of 10**.

- **Amazon Rekognition API + Amazon Web Services SDK for OCR**
  - **Speed/Performance:** Amazon themselves have given a speed review for their Rekognition API at a video streaming rate of 5 FPS. Rekognition unfortunately also caps image sizes at 5mb and will only process 100 words per im- age. This could be a potential issue for future development as modularity is restricted. The general consensus is that Amazon Rekognition API for OCR is roughly the same as Google Cloud Vision and for these reasons, receives a rating of **6 out of 10** for speed/performance.
  - **Accuracy:** Amazon Rekognition is highly reliable when it comes to OCR accuracy. A nice addition with Rekognition is the API actually comes with an evaluations metric console which allows developers to perform evaluation testing on their models and receive real time feedback on the accuracy of the software. Therefore, Amazon Rekognition receives a rating of **10 out of 10** for OCR accuracy.
  - **Ease of implementation:** Amazon Rekognition provides extensive documentation for developer use, however; it is not as honest as it should be. After reading numerous reviews by customers, Amazon S3 (which is required for AWS) has many limitations that are not well-documented within the user manuals. It is also extremely hard to look at which makes finding information efficiently not as smooth as the Google Cloud Vision documentation and Tesseract documentation and therefore receives a rating of **5 out of 10** for ease of implementation.
  - **Library Compatibility:** Amazon Rekognition is dependent on the AWS SDK for development. While the necessary language libraries are provided for translation purposes, the requirement of the AWS SDK is a bit problematic as operations within Amazon S3 directories are often slow and there are limitations via AWS requiring additional money for upgrades. Therefore, Amazon Rekognition receives a rating of **7 out of 10** for library compatibility.
  - **Cross Platform Compatibility:** Amazon Rekognition is compatible with any browser as the AWS SDK is offered for JavaScript which is available on all browser types that have enabled JavaScript access. Amazon Rekognition therefore receives a rating of **10 out of 10** for cross platform compatibility.
  - **Overall Rating:** Amazon Rekognition receives an average rating of **8.6 out of 10**, however; because this is a paid service the rating is being deducted 2 points for an overall rating of **6.6 out of 10**.

3.2.4 | Chosen Approach

After reviewing several alternatives for the best Optical Character Recognition Engine, the following table compares the overall ratings of Tesseract OCR, Google Cloud Vision API + OpenCV for OCR, and Amazon Rekognition + AWS SDK for OCR.

Optical Character Recognition Engine Rankings

	Tesseract OCR	Google Cloud Vision	Amazon Rekognition
Speed / Performance	6	6	6
Accuracy	7	9	10
Ease of Implementation	10	10	5
Library Compatibility	10	10	7
Cross Platform	10	10	10
Deduction for Paid Service	0	-2	-2
AVERAGE	**8.6**	**7**	**6.6**

Table 3.2: Table featuring the comparisons of alternatives for OCR engines based on the conditions of satisfaction metric. Scores range from 0 to 10.

According to the table shown above, it is clear based on the ratings that the best choice for the OCR engine is Tesseract OCR. With that said, it is also noticeable that without the deduction for being paid services, that both Google Cloud Vision and Amazon Rekognition would have scored significantly higher at 9 out of 10 and 8.6 out of 10 respectively. While Tesseract does not show to perform at a faster or more accurate rate, it is the only option of the three that is open source. This is perfect for the purposes of this project as there will need to be major modifications made to the OCR for optimization of speed and accuracy. Tesseract's extremely straightforward documentation, modularity, extensive free trained data file set, and cross platform compatibility, LangLens is confident that Tesseract will perform just as well as Google Cloud Vision and Amazon Rekognition when integrated using OpenCV's EAST, which according to research, is highly recommended to use in tandem to get optimal results from Tesseract.

In conclusion, Tesseract is the chosen approach for the OCR engine as it provides a free, open-source engine that is compatible with numerous API options, and is user-friendly in terms of development, and provides the ability to modify and optimize the software to the project's needs.

3.2.5 | Proving Feasibility

As development progresses, proving the feasibility of the Tesseract OCR engine will be verified by evaluation testing on the OCR text accuracy and performance speed. To demonstrate this evaluation in a quantifiable way, an evaluation script will be developed and run on the model with every execution and stored in a collective dataset that will show whether Tesseract is performing at optimal rates and where improvements need to be made.

3.3 Understandable and communicative frontend framework

As part of our project, LangLens is going to be utilizing the user's camera to scan objects in real time. For this we will need to create a frontend framework that is easy for the user to use with their device. Additionally, we will also need the framework to be able to communicate with the backend framework in order to retrieve the necessary data when scanning the objects, as well as retrieving the correct language to translate the words. On the other hand, the application will need access to the user's camera in order to take in the live feed to start scanning. With that being said, we will need a framework that utilizes the camera and once the desired object has been scanned present the user the object in the translated text. We also want to allow the user the ability to change the type of language they want to scan objects in, as well as change the language of the application in order to use it in the user's appropriate language.

3.3.1 Desired Characteristics

The following characteristics will be used as the conditions of satisfaction metric when evaluating alternative solutions for frontend frameworks:

- **Ease of Implementation:** The framework we want should provide ways for us to create an understandable and readable user interface for the user. That way the user can have an easy way of navigating and using the application. We want to be able to have the framework bring up links to objects and words for the user to learn pronunciations and definitions. For objects we need the framework to bring up images related to the object that is being scanned. Lastly, we need the framework to be able to change to different languages that the user desires to use. This is all necessary for the project to scan objects and words and be able to translate them.
- **Backend Communication:** Since our project will be working with libraries to pull up images and definitions, the framework we need must have good communication with the backend to have access to the libraries. Not only that, but we also need to have access to other languages in order to translate text. This is to ensure that when the user decides the desired language, the text will be displayed in that language.

- **Camera Compatibility:** We need to have access to the camera so that we can take in the information that is being scanned. This is necessary for the project because for us to scan objects we are going to need access from the user's camera. In addition, the framework needs to be able to utilize the camera alongside the scanner to take in live feed.
- **Flexible Learning Curve:** From our research there are many different frameworks out there and many of them are easy to use. However, some of the frameworks seem to have a bit of a learning curve to them, which would imply we would need a framework that does what we want and be relatively easy to learn with the time we have. The lower the score the easier the learning curve is.

3.3.2 | Alternatives

These are alternative frameworks that we propose to use if one of the other ones does not satisfy our standards of satisfaction:

- **ReactJS:** Though React is not actually a framework and instead is a library, it is still one of the best tools for building user interfaces. Some members of our teams have watched videos on React before, and so we decided to do some research on it. React has been used to create familiar applications such as: Instagram, Discord, Dropbox, and Pinterest. ReactJS has been around since May of 2013 and ever since has been getting more and more popular and better at front-end development. However, through some user reviews it seems that React does have a bit of a learning curve.
- **Bootstrap:** Bootstrap is a frontend framework used to create web applications. We decided to research bootstrap because some members of the team mentioned that it is an easy to use and implement framework. Bootstrap has been used in certain applications such as: Twitter, Lyft, Udemy, Duolingo, and Snapchat. Compared to ReactJS, Bootstrap appears to have less of a learning curve. The members on the team also say that Bootstrap is much easier to use.
- **Angular:** Angular is another frontend framework for creating web applications. Microsoft Office, Forbes, and Gmail have all used Angular to create their applications. One of the members on the team mentioned that he has used Angular and from our research Angular also seems to have a learning curve to it. Angu- lar has been around since September of 2016. Angular started development at Google developed by Misko in 2009. Later it would be released to be used by all developers.

3.3.3 | Analysis

The following section provides an individual analysis of each of the alternatives listed in section 3.3.2 and their respective scores out of 10 following the conditions of satisfaction metric described in section 3.3.1.

- **ReactJS:**
  - **Ease of Implementation:** ReactJS is quite easy to implement since it has many different template models for us to experiment with. Has excellent cross- platform support and has reusable components, therefore allowing for reusability for different components if we choose to reuse them. However, ReactJS does seem to have a bit of a learning curve to it, which we will go into, giving it a 9 out of 10 for ease of implementation.
  - **Backend Communication:** Through many of the user reviews ReactJS is compatible with most, if not all the different backend frameworks that are out there. Although, for the most part backend compatibility differs from user to user so React will get a 9 out of 10 for backend communication.
  - **Camera Compatibility:** ReactJS has ways of accessing the camera, therefore it receives a 10 out of 10 for camera compatibility. Additionally, it was used to create Instagram and Discord, which are two different applications that can utilize the user's camera.
  - **Flexible Learning Curve:** One thing we took to account for was the learning curve of React. From many of the different user reviews ReactJS is more of an advanced frontend framework. Many users said that React is much harder to learn than other frameworks because it utilizes many different concepts that other frameworks do not use. This would mean our team might end up spending more time trying to figure out these concepts. Not only that, but if we do not know how to use the concepts properly, then there may be more errors. ReactJS will receive a 6 out of 10 for a flexible learning curve.
  - **Overall Rating:** ReactJS is a very powerful framework to use and offers a lot of different benefits, however, since it is one of the more advanced frameworks, it is much harder to work in. Therefore, ReactJS will receive an average rating of 8.5 out of 10.

- **Bootstrap:**
  - **Ease of Implementation:** Bootstrap has many different tutorials on the internet on how to use the frontend framework. Bootstrap is also recommended to be one of the best frameworks for beginners. With that being said, it comes with a lot of support and has a vast number of plugins for developers to use. Additionally, it has cross-browser compatibility and has templates for different elements of the user interface. Bootstrap receives a 9 out of 10 for ease of implementation.
  - **Backend Communication:** Bootstrap is very compatible with many different backend frameworks. In fact, it is recommended to utilize a backend with Bootstrap, otherwise it would just be a static website. Many user reviews even agree to this since they are able to implement it with many different other backends. There are even blogs that tell you how to create websites using Bootstrap and NodeJS. Bootstrap receives a 10 out of 10 for backend communication.
  - **Camera Compatibility:** Bootstrap has ways of utilizing the camera. There are even tutorials on how to access the camera live. Snapchat is one of the biggest companies that utilizes the user's camera, in fact Snapchat uses Bootstrap for certain user interface components. Therefore, Bootstrap receives a 10 out of 10 for camera compatibility.
  - **Flexible Learning Curve:** With Bootstrap being one of the best frontend frameworks for beginners the learning curve will not be that bad. Most of the team already has experience using HTML, CSS, and JavaScript, which will make learning Bootstrap a lot easier for us. Bootstrap receives a 2 out of 10 for a flexible learning curve.
  - **Overall Rating:** Overall, Bootstrap is a very handy frontend framework for us to use to implement our user interface. Even if Bootstrap is not the most advanced option, it is still one of the easiest to learn making it one of the best choices for us. Bootstrap receives an average score of 7.75 out of 10.
- **Angular:**
  - **Ease of Implementation:** Angular offers many different tutorials on how to utilize the framework. Angular has better speed and performance when it comes to loading in order to conduct better testing. Angular also has cross- platform development and offers problem-solving services to reduce productivity issues for developers. Angular receives a 9 out of 10 for ease of implementation.

- **Backend Communication:** Many user reviews say that Angular is very compatible with many different backend frameworks and that for the most part it is usually just user preference. Therefore, Angular makes a good option for us to use for our project. Angular receives a 9 out of 10 for backend communication.
- **Camera Compatibility:** Angular is also capable of using the camera. There are several tutorials that teach you how to use Angular to access your camera. There are not that many applications that utilize Angular to access the camera, nonetheless it is still capable of accessing it. Angular receives a 10 out of 10 for camera compatibility.
- **Flexible Learning Curve:** Although Angular is not as hard to learn as React, it utilizes another type of language that is built on JavaScript which is Typescript. Typescript would be something new for us to learn, which depending on how fast we are to learn it, could take up some more time than is needed. Angular receives a 5 out of 10 for a flexible learning curve.
- **Overall Rating:** Overall, Angular is a really good option for us for our project and is also advanced at many things, however, with its learning curve it might take us some time for us to learn in order to start developing. Angular gets an average rating of 8.25 out of 10.

3.3.4 | Chosen Approach

After reviewing several alternatives for the best frontend framework, the following table compares the overall ratings of ReactJS, Bootstrap, and Angular for frontend frameworks based on conditions of satisfaction described in section 3.3.1.

Frontend Framework Rankings

	ReactJS	Bootstrap	Angular
Ease of Use	9	9	9
Back-end Compatibility	9	10	9
Camera Access	10	10	10
Learning Curve	6	2	5
AVERAGE	8.5	7.75	8.25

Table 3.3: Table showing the comparisons between the alternatives and the desired characteristics on a scale from 1-10.

For our chosen approach for our frontend framework, we decided to go with Bootstrap as shown in Figure 2 above. Although most of our alternatives provide pretty much the same tools for a frontend framework, Bootstrap would have the smallest learning curve out of the three. Bootstrap utilizes HTML, CSS, and JavaScript, which are tools we are most familiar with, whereas ReactJS is mainly JavaScript and Angu- lar utilizes HTML, JavaScript and Typescript. The bootstrap framework also improves responsiveness, which would be more useful here in order to bring up data on the scanned items. Most, if not all, frameworks have a way to access the user's camera and are quite handy with communicating with the backend of applications. Another reason as to why we chose Bootstrap is because it is very compatible with many other backend frameworks. We are confident that all of the technologies that have been chosen in this document are compatible with Bootstrap.

3.3.5 | Proving Feasibility

Since Bootstrap is the framework we have chosen we will have to demonstrate how we will be using the framework for our project. For our demo we will construct a user interface that utilizes the user's camera to demonstrate the interface can utilize the camera.

3.4 Retrieve an image from the scanned word / object

One desired stretch goal of the LangLens application is the possibility to review the saved words by the user and interact with them via PLAYlists (a card swiping feature). To do that, the application will show a card where an image represents a saved word in the user's PLAYlist will be displayed. The user has to try to guess the word based on the image. Once the user has thought of a word, they can flip the card to see the correct word. If the user guesses the word, they can click in the right side of the image, removing this card from the game. In case the user fails, the left side of the image can be clicked, adding the card to the queue again. In order to achieve this, an image retriever based on a given word is necessary.

3.4.1 | Desired Characteristics

The following characteristics are what we expect it to meet to measure performance.

1. **Speed / Performance:** Ideally, the processing speed at which the application should be able to retrieve an image in an efficient way avoiding long waiting times. This is to ensure that the user experience is fluent.
2. **Accuracy:** The accuracy of the image retriever is another important characteristic that will determine which solution is chosen. The image has to represent the word in a faithful way to avoid causing confusion to the user.
3. **Ease of Implementation:** The image retriever should be easily implementable and include extensive documentation for developers.
4. **Library Compatibility:** The image retriever software needs to be compatible with any type of language / framework.
5. **Requests Per Month:** If the image retriever is an API, it has to provide enough requests per month to support heavy workflow for the case of a big amount of users monthly.

3.4.2 | Alternatives

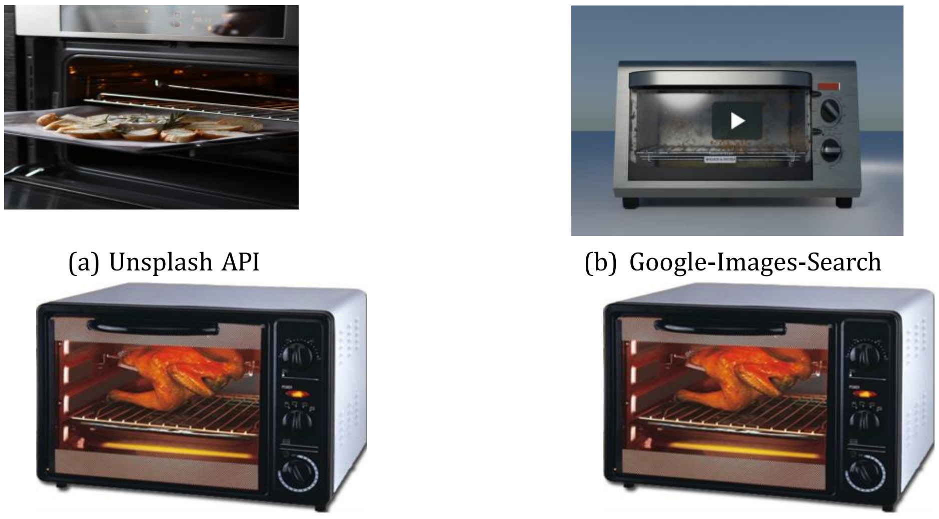
The following options describe the numerous alternatives for a way to retrieve an image based on a word that could potentially be used for this stretch goal feature:

- **Unsplash API:** It is a free images library where you can download professional photos to use in any project. It has a free API for developers, allowing you to retrieve any photo from your program.
- **Google-Images-Search:** Python library that allows you to search for Google Images.
- **Bing-Image-Downloader:** Python library that downloads images from Bing explorer.
- **Bing-Image-Urls:** Python library that gets url images from Bing explorer.

3.4.3 | Analysis

For this analysis section, a set of tests will be carried out. This implies searching a list of everyday words using every different alternative and evaluating how they perform based on how well they represent the given word. The list contains the following words: freezer, oven, chair, table, laptop, keyboard.

We can see the result of the four alternatives for the word oven.



(a) Unsplash API (b) Google-Images-Search

(c) Bing-Image-Urls. (d) Bing-Image-Downloader

Figure 3.1: Oven image search.

As we can see in the last pictures, Google-Images-Search does not get a really representative picture. Bing-Image-Urls and Bing-Image-Donwloader get the same photos because they are using the same explorer, but they are not either good quality photos. Finally, Unsplash API gets the most representative and good quality pictures.

3.4.4 | Chosen Approach

- **Unsplash API:** It is not the fastest one, but it gets accurate pictures, its implementation is easy since it incorporates a Python library making it compatible with our project. The number of requests per month is the only negative point, because they may not be enough at the beginning.
- **Google-Images-Search:** This one got the worst test results. It is slow and inaccurate, as well as its implementation is a little tricky. As positive points have good compatibility and a high number of requests per month.
- **Bing-Image-Downloader:** It got good results in the test, but there is no way to get the url of the image, forcing you to download the images. Accuracy is not good enough.
- **Bing-Image-Urls:** It got same results as its "brother", but you get the url image with this library, being faster, but the accuracy is not good enough.

Image Retrieval Rankings

	Unsplash	**Google-Images-Search**	**Bing-Image-Downloader**	**Bing-Image-Urls**
Speed	7	4	8	10
Accuracy	10	4	6	6
Ease of Implementation	10	6	10	10
Library Compatibility	10	10	10	10
Requests Per Month	6	8	10	10
AVERAGE	8.6	6.4	8.8	9.2

Table 3.4: Table showing the comparisons between the alternatives and the desired characteristics on a scale from 1-10.

Analyzing the table, Bing-Image libraries get the best average because of the unlimited number of requests per month, but Unsplash API is the best tool for our goal. It may be slower compared to its competitors, but it gets the highest quality and fidelity images. The low requests number per hour will not be an inconvenience, because every word will be stored with its url image, so it will never search two times for the same word, and the more the application is used, the fewer requests will be necessary each time.

3.4.5 | Proving Feasibility

To prove that the chosen approach solves our problem, a demo consisting of searching images for every word in a list will be developed. The task will be solved if a fidelity image is obtained for every word in the list. The words will be from different everyday topics.

3.5 | Backend Database

The LangLens application needs to have a secure database for storing detected text and objects. A desired stretch goal would be to eventually allow users the ability to save the object labels into a reviewable PLAYlist so they may be able to recall what they have scanned and experience an interactive way of learning. The bare bones for the application will require some kind of backend web database api that can save and store into a server, communicate with the detection engines, and eventually communicate with the user via the PLAYlists.

3.5.1 | Desired Characteristics

The following characteristics will be used as the conditions of satisfaction metric when evaluating alternative solutions for a backend database:

1. **Speed / Performance:** Ideally, the processing speed at which the application should be able to save and load the objects and object labels in an efficient way avoiding long waiting times. This is to ensure that the user experience is fluent.
2. **Ease of implementation / Features:** Ideally, our team should not have to jump through hoops to develop the backend, and also use special features of individual API's to help the ease of development.
3. **Security:** In terms of our customers' experience, their security is a top-notch priority so that the information does not get stolen or corrupted.
4. **Testing:** A backend development API that is testable will be incredibly important for the development of this application. Most of next semester will be proving that our software works with a variety of tests, so if the testing on the API is slow or non-descriptive it will make doing that significantly harder.
5. **Storage Variability:** The backend database needs to be able to hold a variety of data that can easily be accessed and communicated with via the frontend.
6. **Features:** The key features we want to work with is the ability to work with the camera so that so that there is a seamless save between scanning the image, and saving it to the back-end environment.

3.5.2 | Alternatives

The following options describe the numerous alternatives for a backend database that could potentially be used within LangLens:

- **Django:** Django is a web development API. The API is free and open source so we would not have to pay for any needed features. It is created in python, which is a language that everyone in our project is familiar with, and seems very easy to set up quickly. Django has been used to hold backends with huge datasets in such popular applications as Instagram, Spotify, Pinterest, and Eventbrite. There is a steeper learning curve within Django, as there are predefined variables and set preset files that one must learn about before creating their backend through Django, that if used incorrectly will not allow you to deploy using Django.
- **Firebase:** Firebase is Google's app development interface platform that was launched in 2012. Firebase is a real time database, which means it can synchronize data through its cloud storage. Firebase's database is quick to implement and has a wide range of services and features. Firebase relies on a flat data hierarchy, so complicated queries and other forms of data saving may not be possible. Firebase provides authentication through a google account. Firebase's scaling is automatic, so it calculates the minimum number of updates needed to keep the backend synchronized with the application. Firebase is used by the New York Times, Half- brick and Alibaba as their backend and user authentication systems. Firebase is generally a quick and easy tool for beginner web application developers.
- **Node.js:** Node.js is an asynchronous event-driven JavaScript backend. Node.js is designed to build scalable network applications and is designed without threads and locks. It is written in JavaScript and uses object-oriented programming, which is common within our development environment. Since Node.js is single threaded, calling many processes will be slow. There are many packages out there for Node.js but finding them and implementing them will take time. Node.js has been used to develop the following popular applications: LinkedIn, Netflix, Uber, PayPal, eBay, and even used by NASA. Node.js has been around for significantly longer than Django or Firebase, so there is extensive documentation over the years for it.

3.5.3 | Analysis

The following section provides an individual analysis of each of the alternatives listed in section 3.5.2 and their respective scores out of 10 following the conditions of satisfaction metric described in section 3.5.1.

- **DjangoDB:**
  - **Speed/Performance:** Django uses python, which is a high-level language that offers speedy execution of back-end processes. Django is also a lightweight framework which combined with efficient python code can result in fast native execution code. The speed of execution with the lightweight framework nets Django's Speed / Performance rating of 9 out of 10.
  - **Ease of Implementation:** Django's ease of implementation comes down to the language, being python, which everyone on Team LangLens is familiar with. Although we can and will implement separate libraries for cases that are very specific, everything general that we need does not need to come as a separate library. As mentioned, there are predefined variables, as well as preset files that one needs to make sure are correctly implemented before being able to set up a Django backend. For these hurdles, LangLens ranks Django's ease of implementation as a 6 out of 10
  - **Security:** Django provides good security measures right out the box when you install the framework. The security measures it comes with include mechanisms to defend against common attacks to backend servers, while also having good permission policies. Because of these desired security measures, without much implementation nets Django's s Security rating of 9 out of 10
  - **Testing:** Django's testing-execution framework and assorted utilities allows you to be able to simulate requests, insert test data and inspect the output of the application. Writing tests can be done by using the unit test module, which is built into the standard python library. For these reasons, LangLens ranks Django's testing rating as an 8 out of 10.
  - **Storage Variability:** It is possible to store multiple types of data within Django's Storage Class. Being able to store objects detected by the camera with YOLOV5 and store learned words from our front end with bootstrap, is why LangLens ranks Django's storage variability as a 9 out of 10.

   - **Overall Rating:** Django seems to be an obvious choice, and our group seems to be behind implementing it into the LangLens application, and it has an average rating of 8.2 out of 10

- **FireBase:**

   - **Ease of Implementation:** The Quickest and easiest way to implement database capabilities, while offering a wide range of services and features that would help ease development. This easy setup gives Firebase's Ease of implementation a 9 out of 10.
   - **Security:** Firebase security rules work by matching a pattern against database paths, and then applying custom conditions to allow access to data at those points. Firebase services encrypt data in transit using HTTPS and logically isolate customer data, so all of the data that would be sent to the firebase backend would be encrypted. The trouble that comes from Firebase's security is within Google themselves, as Firebase has been claimed to be used by Google to track users without their knowledge. In July of 2020, a lawsuit was filed accusing Google of violating federal wiretap law and collecting stored user data, logging what the user was looking at in many types of apps. For these reasons, LangLens ranks Firebase's score as a 4 out of 10.
   - **Testing:** The Firebase test lab lets you run Espresso, Robotium and UIAutomator 2.0 instrumentation tests to work on testing an application. These tests can analyze the structure of the app's user interface and simulate user activities. These are very desirable functions for testing, but it comes at a price. Firebase test labs are free but are limited to up to 15 test runs per day, which depending on how we are developing could be far too little. The paid option allows for 5 per hour for each physical device and 1 per hour for each virtual device. Although Firebase has 9 out 10 features for testing, the constraints it puts on the development bump it down to a 7 out of 10.
   - **Storage Variability:** Firebase might have some troubles working with our data that we are sending to the backend, as it is a flat data hierarchy, and can only process a limited amount of concurrent data, which gives Firebase's Storage Variability a 6 out of 10.
   - **Overall Rating:** Firebase has an average rating of 6.2 out of 10

- **Node.JS:**

   - **Speed/Performance**: Node.js uses single threaded programming and asynchronous handling of events, so having a single threaded program handle very large amounts of requests that our programming is scanning may have
   us face problems, and for that reason LangLens ranks Node.js's Speed/Performance as a 5 out of 10.
   - **Ease of Implementation**: Easy to learn, as many know JavaScript with object-oriented

programming. There are also many tutorials on Node.js as it is one of the longest standing backend databases, which could help with less strain on the development process. These easy to implement features give Node.js's ease of implementation a score of 8 out of 10.

- **Security:** Node.js itself claims to be a secure platform, but since we might have to implement third-party open-source packages to implement camera functionality, it could be vulnerable to attacks, which would cause strain on our development, and for this reason, LangLens ranks node.js's security as a 6 out of 10.
- **Testing:** For unit testing, Node.js would need its own testing framework, which come in a variety of options, including mocha, which is flexible but needs a complex setup, AVA and Tape which are minimalistic testing frame- works, and Jest, which can be used for a variety of different frontends and backends. Since node.js has no built-in testing, but has options to choose from, LangLens ranks Node.js testing as a 4 out of 10.
- **Storage Variability**: Node.Js has the ability to store multiple different types of data, as well as being able to store a large amount of data, which will be important for storing large amounts of data when scanning for object and text recognition. For this reason, LangLens rates Node.js as an 8 out of 10.
- **Overall Rating:** Node.js's average rating is 7.75 out of 10.

3.5.4 | Chosen Approach

After reviewing several alternatives for the best backend database, the following table compares the overall ratings of backend databases based on conditions of satisfaction described in section 3.5.1.

Backend Database Rankings

	Django API	Firebase	Node.js
Speed	9	6	5
Security	9	4	6
Ease of Implementation	6	9	8
Testing	8	7	4
Features	9	6	8
AVERAGE	8.2	6.4	7.75

Table 3.5: Table showing the comparisons between the alternatives and the desired characteristics on a scale from 0-10.

Based on the average ratings displayed in table 3.5 above, we have decided to go with Django as our backend framework. The other frameworks, Firebase and Node.js do not hold a candle to what Django can provide for us regarding speed, performance, ease of implementation, security, testing, and features. Django utilizes Python, which is a tool the members of LangLens are familiar with. Although Firebase might have easier implementation and easier security, Django beats them out in speed and needed features. The amount of data that we will need to process, while giving a seam- less augmented reality experience is incredibly high, so Django's speed, performance and features make up for its learning curve.

3.5.5 | Proving Feasibility

To demonstrate have Django will be used as our chosen backend framework, we will display the detected text and objects that are stored and called upon via some type of external output ie. pdf, CLI, etc.

4 Technological Integration

The following section will discuss and demonstrate how each of the solutions chosen for the main technological challenges will come together to form the coherent architecture for LangLens. Figure 4.1 below depicts the overall envisioned system and how each major technological challenge is anticipated to be addressed and integrated.

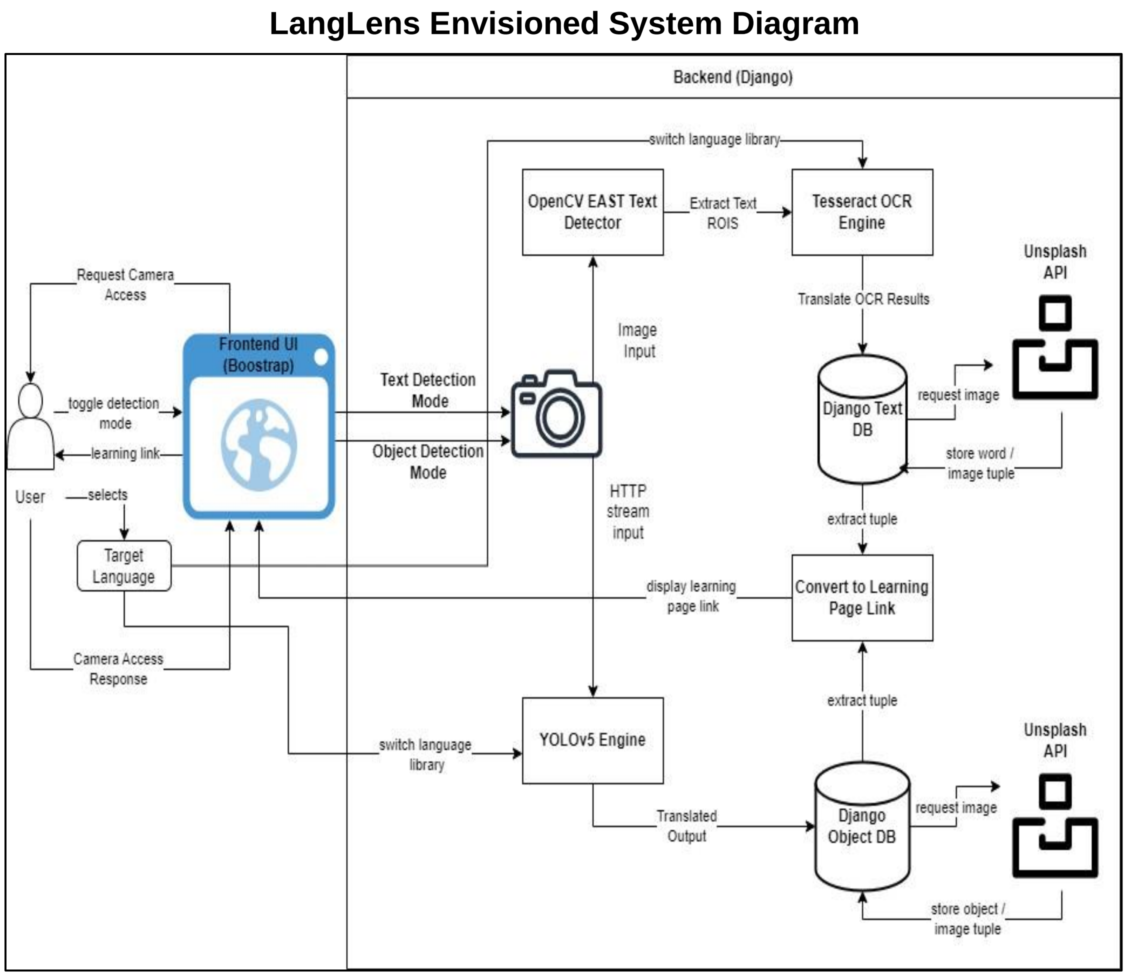


Figure 4.1: Envisioned System Diagram

When users access the site, they will have the option of toggling between two detection modes that are prompted on the UI: Text Detection and Object Detection. Users will also need to select their target language for translation as Tesseract and YOLOv5 will automatically detect the base language. The frontend UI will be constructed using Bootstrap and will be able to communicate to the Django backend for either mode selected and will also communicate with the detection engines to indicate which translation library to use based on the user's target language preference. Upon choosing a detection mode, a permission request will be prompted to the user for access to their mobile camera. This permission is a major factor in whether the application will function as if the user declines camera access no further process will be initiated for personal security purposes.

1. **Text Detection Case:** If the user selects the text detection mode, the camera will begin scanning the environment that the user provides with their mobile back camera. The bounding boxes of the OpenCV EAST Text Detector will be displayed on screen to the user as they highlight the word they are trying to translate. Once the word is detected within the image, EAST will extract the text ROI and send it to the Tesseract OCR engine. Tesseract will utilize its prepacked deep learning algorithm for text recognition. According to the user's selected target language, Tesseract will translate the recognized text and send the translated result to the text-designated Django database. Django will then make a request to Unsplash API based off the translated text result and will use Unsplash's large open collection of images to find the corresponding image to the text and then store the text and image as a tuple back into the Django DB. The tuple will be extracted and converted into a language learning page via a bootstrap template and then displayed back to the user on the UI from which the user can click the link or restart scanning.
2. **Object Detection Case:** If the user selects the object detection mode, the camera will begin scanning the environment that the user provides with their mobile back camera. The bounding boxes that YOLOv5 provides will immediately be displayed on screen to the user as the desired object in their FOV is highlighted and labeled according to their chosen target language. Simultaneously, the translated object label will be stored in the object-designated Django database. This allows for the database to communicate with Unsplash API and request a graphical 2D representation of the object scanned and stored back into the object database as a tuple consisting of the object label and the respective image. As this is happening in real time, the backend processes will need to happen in parallel with the scanning. Once the tuple is stored, it will be extracted and converted into a language learning page via a bootstrap template and then displayed back to the user on the UI from which the user can click the link or restart scanning.

It is not displayed within the envisioned system diagram above, but eventually a stretch goal for LangLens would be for users to be able to toggle to a practice learning mode to review the objects or words they have saved in PLAYlists that they can then use for continuous learning. If there is a chance to implement such a feature during the project timeline, it will utilize a third Django database as well as Unsplash API and will store the graphical representation of the label as a picture and allow the user to practice learning their saved scans in a card swiping game. With the way that the system is currently envisioned, it shouldn't be very difficult to integrate that stretch goal in the future.

5 Conclusion

Formally studying a new language has the learners read, write, memorize, and sometimes speak the words they are learning. However, being able to interact with, and learn about any object or item that a user would see or point towards in their surroundings, would help solidify one's knowledge in a very interactive manner. As previously discussed, Augmented Reality (AR) is nearly completely absent from computer assisted language learning tools. LangLens aspires to utilize AR and make language learning experiences more immersive and interactive than ever.

In this document, we discussed the technological challenges we anticipate encountering and have done thorough research and technical analysis on every single one. Our analysis consisted of stating desired characteristics for each of our solutions, providing some alternative solutions, justifying our chosen approach, and explaining how we will go about proving feasibility. Finally, we explained how all these solutions will come together to form a functioning web application in our tech integration section.

The major challenges, and their respective solutions include:

- **Object detection -** YOLOv5.
- **Optical Character Recognition -** Tesseract OCR.
- **Front-end framework -** Bootstrap.
- **Retrieve an image from scanned word / object -** Unsplash API.
- **Backend Database -** Django.

The next step for LangLens is to prove the feasibility of our project. Different technological challenges have different methods of proving the feasibility, such as creating an object detection prototype for YOLOv5 or running the Tesseract OCR engine through an evaluation script, in order to obtain quantifiable results for optical character recognition accuracy and speed. We are confident that through our research, LangLens will be able to develop a fully functional and freely accessible language learning web application that utilizes the most recent advances in AR technology.