Exploring Android VQA through Tensorflow

to assess Cognitive Visual Recognition

Matthew Velazquez

CSEE - UMKC

Kansas City, Missouri

mattvyse8@gmail.com

*Abstract*—This research seeks to expand upon the current applications of combining the Android operating system with Tensorflow by providing a visual question answering platform for image analysis. This application, Cognitive Visual Recognition Tracker (CVRT), provides an entry point by which the user can ask questions concerning any image of their choosing, and then receive cumulative metrics over time to better assess any diminishing cognitive ability (i.e. Alzheimer’s patients). In this work, recurrent neural networks as well as semantic analysis are leveraged to provide an interactive visual question answering experience for the Android user. One of the main objectives of CVRT is for physicians to be able to determine trends from patient data that could either be applicable to the individual patient, or to many patients if an aggregate is formed from many individual datasets. On an individual level, these metrics would provide a way for the physician to monitor daily cognitive capability, whereas on a grander scale, these joint datasets could be used to provide better overall treatment for the disease with the future inclusion of predictive analytics. As for short-term advantages, since this monitoring would be implemented daily, delirium can be more quickly identified as it could be the logical result of a sharp drop in image recognition capability. The final contribution is an interactive metrics platform by which other users can assess the primary user’s cognitive capacity based on features of their questioning, and to then provide them with accurate trending or possible remediation plans based on their condition.

# Introduction

While there are many demos available in regards to the topic of Visual Question Answering, very few exist that have applicable results for the medical field. With this in mind, CVRT was designed to build on the existing Visual Question Answering space, but to produce practical, medically relevant data as a result. This application is significant because it provides a unique way to track the progression of Alzheimer’s disease based on image recognition in a more data-driven way than was previously possible. With the combination of intelligent semantic analysis alongside image classification, new methods are now able to be explored in the medical domain, and with CVRT, an attempt is made to build a powerful data model alongside the added flexibility of a mobile UI. By using the COCO-QA dataset mentioned in (1), alongside the Tensorflow and Android framework, CVRT is able to serve the following features to the end user:

* User may choose any picture to be displayed in the application with a question-answer focused interface
* Users can ask questions about those images in order to discover their identities/characteristics. (i.e. What is happening in this picture? Where is this? What item is on the table? etc.)
* Ability to track metrics for the individual patient’s image recognition ability over time.
* Short-term monitoring that can detect delirium and advise for emergent medical treatment.
* Ability to aggregate multiple patients’ data to allow for better predictive analytics

A common scenario for this application would be a family wishing to track the cognitive status of a loved one with dementia or Alzheimer’s disease. This individual would be the primary user as they would be the one launching the application in order to ask questions. The application will allow them to skip as many pictures as they’d like, or they can stay focused on one particular image until their curiosities are satisfied. The application will provide the primary user with answers to their questions while simultaneously logging results to a database. This database would then allow for family members to view metrics on the individual’s cognitive visual recognition health.

# Related Work

In regards to the Visual Question Answering space, similar work has been done related to the questions being asked of an image with answers generated in response. These demos exist online and serve as a form of introduction to the domain itself, however they do not apply to the medical field. There is also research that has been done regarding impaired visual recognition as a way of predicting the onset of Alzheimer’s, but this research has not been leveraged in a deep-learning application. As for the combination of neural networks with visual semantic embedding, the work done by the University of Toronto was a major source of the CVRT Tensorflow model.

Visual Question Answering Demo: <http://visualqa.csail.mit.edu/>

Visual recognition memory in Alzheimer’s disease: repetition-lag effects: <https://www.ncbi.nlm.nih.gov/pubmed/18568983>

Impaired visual recognition memory predicts Alzheimer’s disease in amnestic mild cognitive impairment:

<https://www.ncbi.nlm.nih.gov/pubmed/23572062>

Image Question Answering: A Visual Semantic Embedding Model and a New Dataset https://arxiv.org/pdf/1505.02074v1.pdf

# Proposed Work

For the tensorflow model side, this work was primarily done using recurrent neural networks in combination with semantic analysis building off of the tensorflow model seen in (1). By leveraging the COCO-QA dataset in conjunction with the DAQUAR QA dataset, each custom image can be attributed a proper QA set from which the Android UI can interact with through verbal invocation. The second part of the work is concerned with metrics generation from the SQLite database created from the Android code.

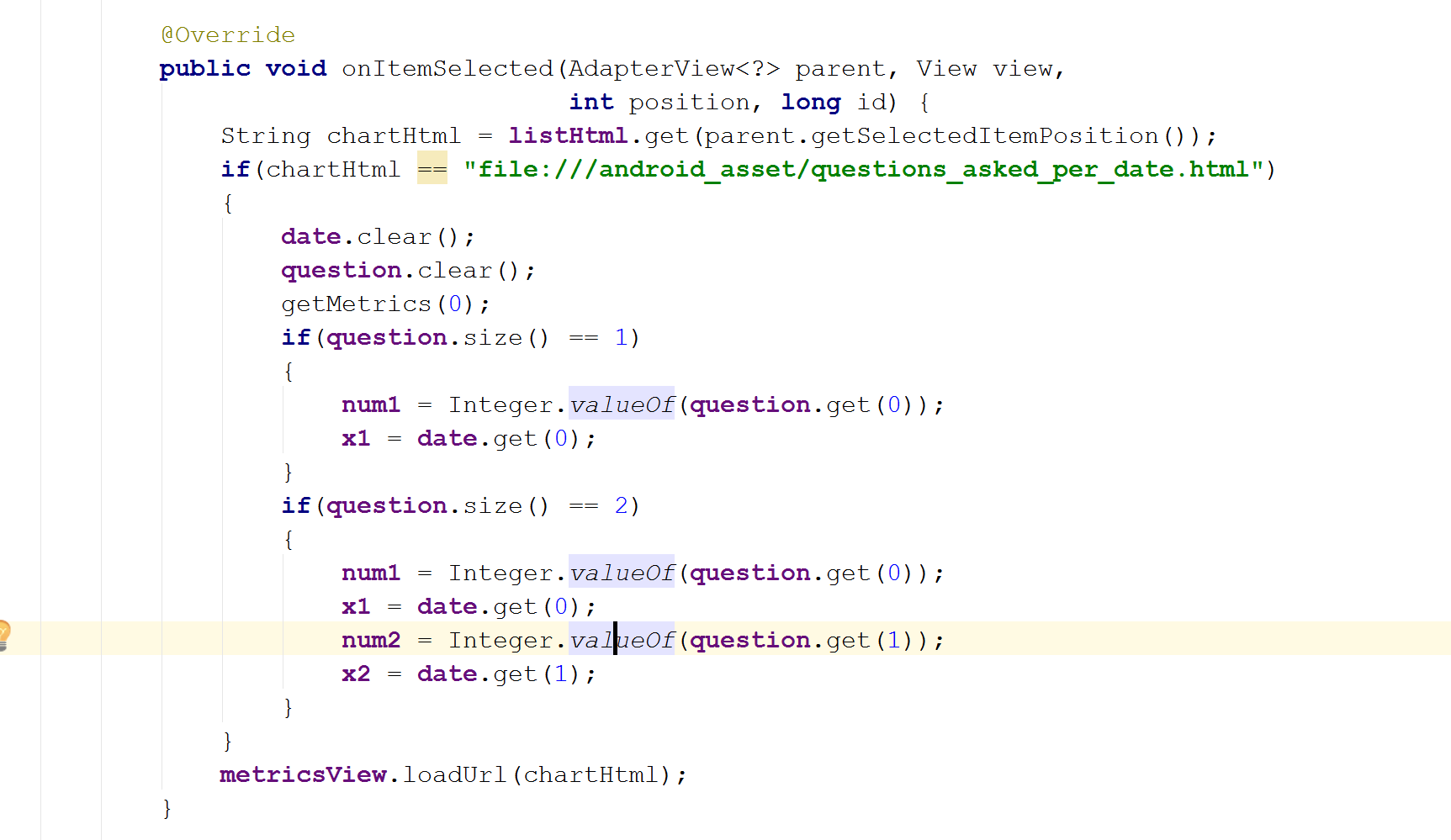
The main analytic tool that will be used for the metrics component of this application will be done with the Google Charting API. This API allows for our stored database information (imageID, question, answer) to be rendered as user-friendly charts that will be displayed via an Android WebView. Bar graphs and line charts will be featured heavily to measure the frequency of the patient’s questions in relation to the image that they decided to focus upon. Over time, this can be leveraged to determine trends that could possibly predict the patient’s rate of cognitive visual recognition impairment. In addition to identifying trends, these types of graphs will help recognize outliers which could signify the sudden emergence of delirium.

**3.2 – Algorithms/Models**

For CVRT, the image recognition is being done via the python tensorflow model for neural VQA. Once an image has been selected from the user’s phone, it imports as a bitmap and is then converted into a byte array. After this conversion, it is then compressed in order to pass to the python server. This is done as a background Async task so that it does not interfere with the processes occurring on the main thread. The python predict.py script asks for only two parameters (image and question) in order to generate a proper answer back to the Android device. An example of the default model parameters as well as some of the initial model options can be seen in the image below:



In regards to the metrics algorithms, once the image, question, and answer are stored in the database, each chart references the data points in a similar fashion to the image below:



This is slightly different depending on the amount of dependent or independent variables used in the chart, however, they are essentially referencing variables in the javascript portion of the CVRT Android model.



# Implementation and Evaluation

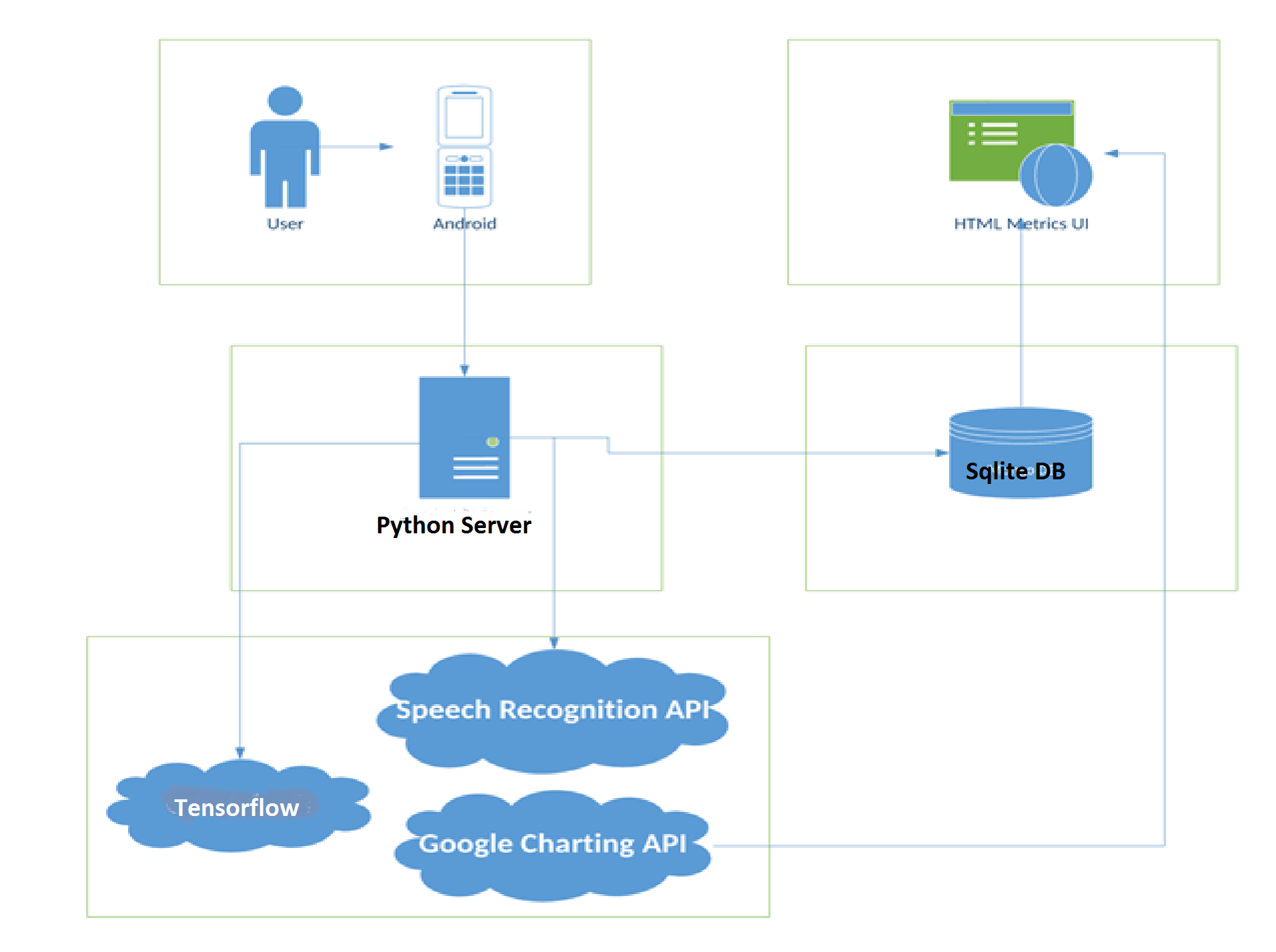
The primary data source when the application launches will be the Android device’s gallery or sdcard photos. Ideal images to select would be common locations that the patient should be familiar with to hopefully gauge if they’ve retained their familiarity with those locations. If not, how much has it declined? Once the application has loaded, the relevant information will come via the Google Voice Speech Recognizer as the patient asks questions concerning the image. This will be stored in a variable that is then passed to the rest of the application’s workflow.

The primary input for CVRT (Cognitive Visual Recognition Tracker) are the questions that are asked via voice recognition by the patient. After this speech is recognized and assigned to a variable, it is then passed to the Python Server alongside the image details to perform the image analysis. This combination of inputs is then used to generate an appropriate answer to the user’s question as the output. Another output that can be expected is the write to the database of the image ID, question, and answer. These will then output to a WebView for statistical viewing via the Android device.

**4.2 – Services Used**

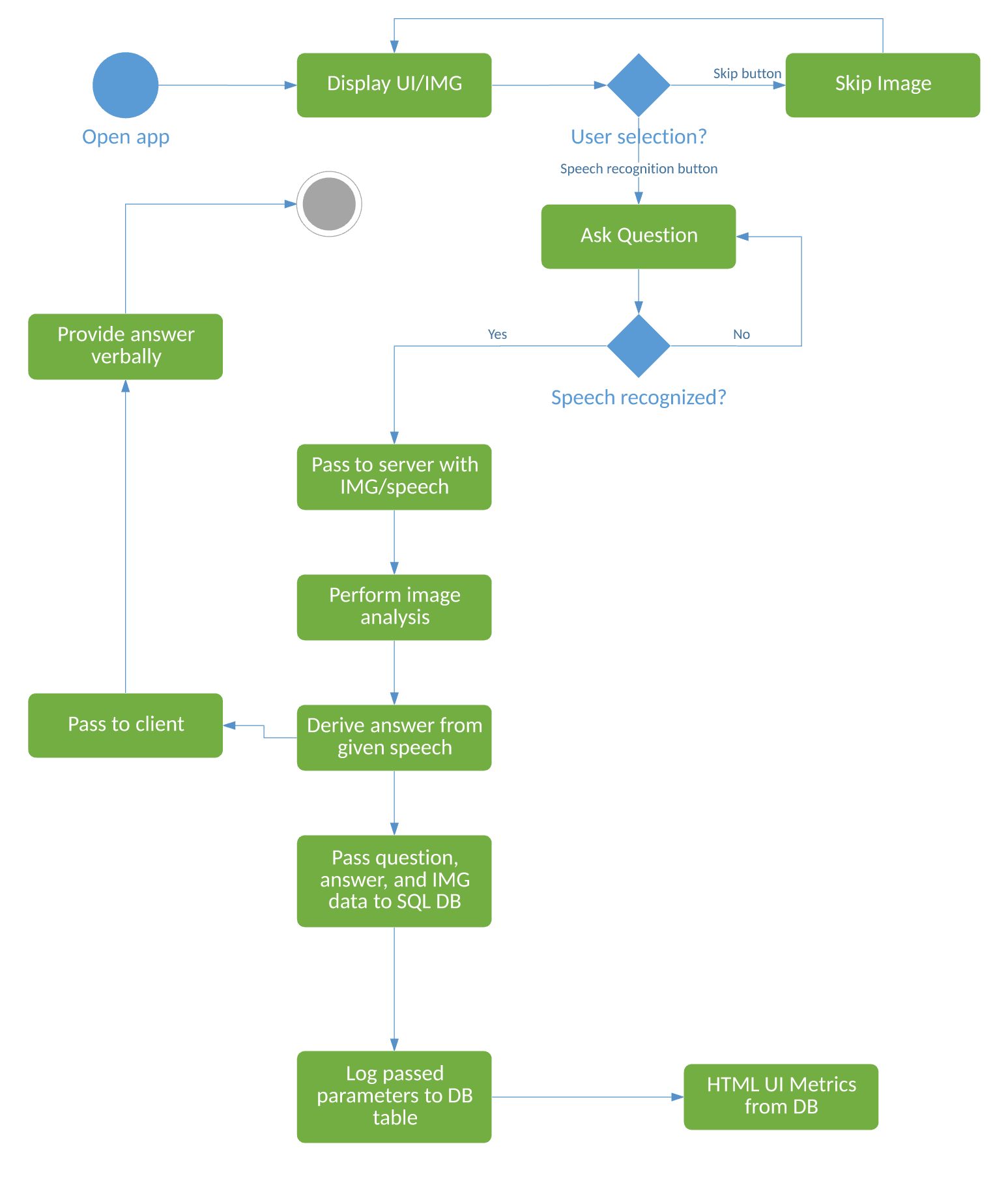
|  |  |  |
| --- | --- | --- |
| Service Name | URL | Description |
| Heroku | https://heroku.com | PaaS that allows for web app delivery |
| SQLite | https://www.sqlite.org/ | Stores data on the local Android device |
| Android | https://developer.android.com/index.html | Primary OS for mobile platforms |
| Google Charting API | https://developers.google.com/chart/ | Charting engine for data rendering |
| Tensorflow | https://www.tensorflow.org/ | Machine learning library |
| Google Speech Recognition API | https://cloud.google.com/speech/ | Provides Speech Recognition |

**4.3 – Software Architecture**

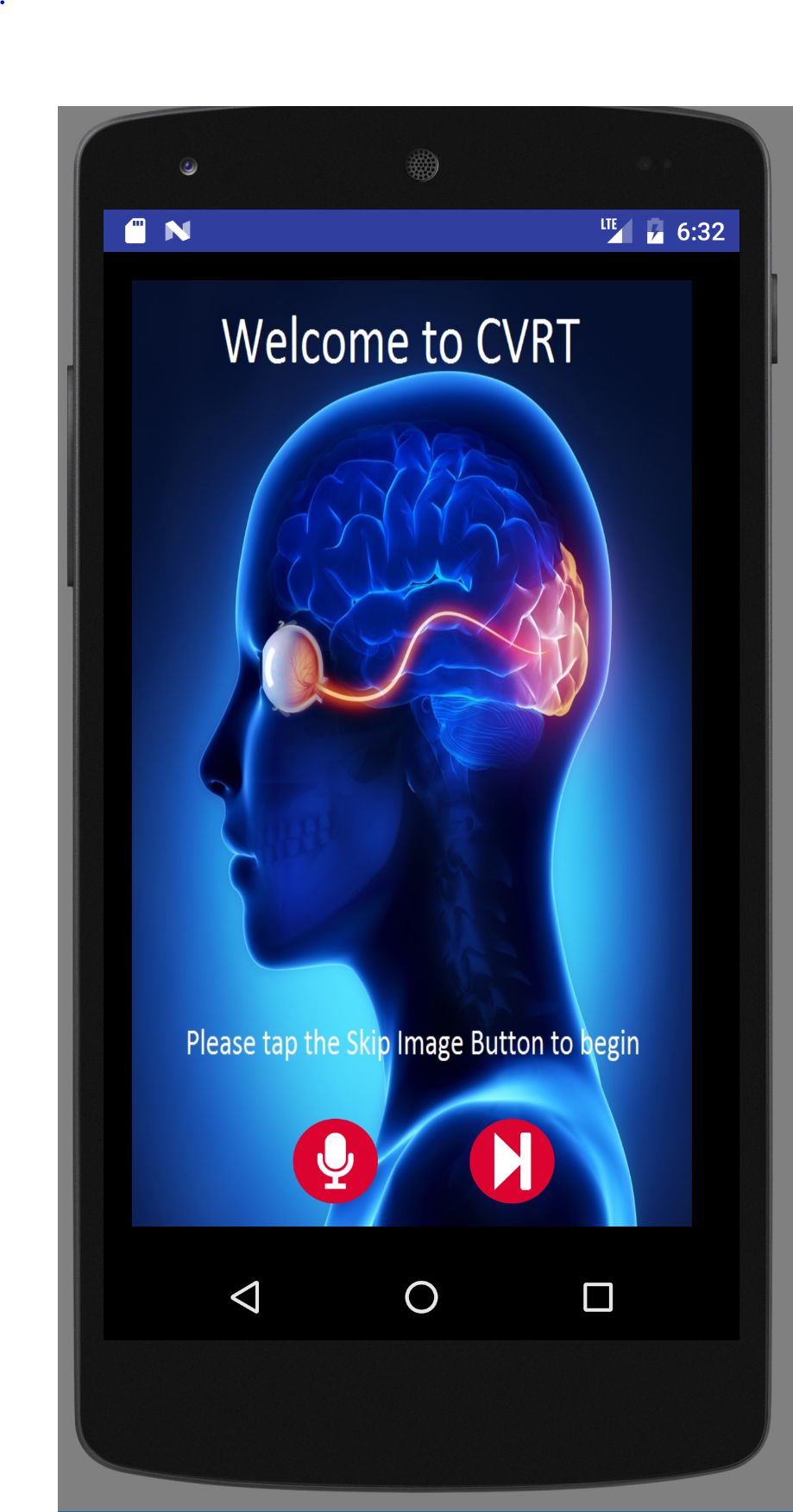


**4.4 – Workflow**

The activity diagram below gives a good example of the typical workflow that a user can be expected to experience while using CVRT. The speech recognition button triggers the first of our API usages and prepares the intent for the ‘Ask Question’ node. The question variable is then assigned based on whether the ‘Speech Recognized’ node can interpret the user’s speech or not.



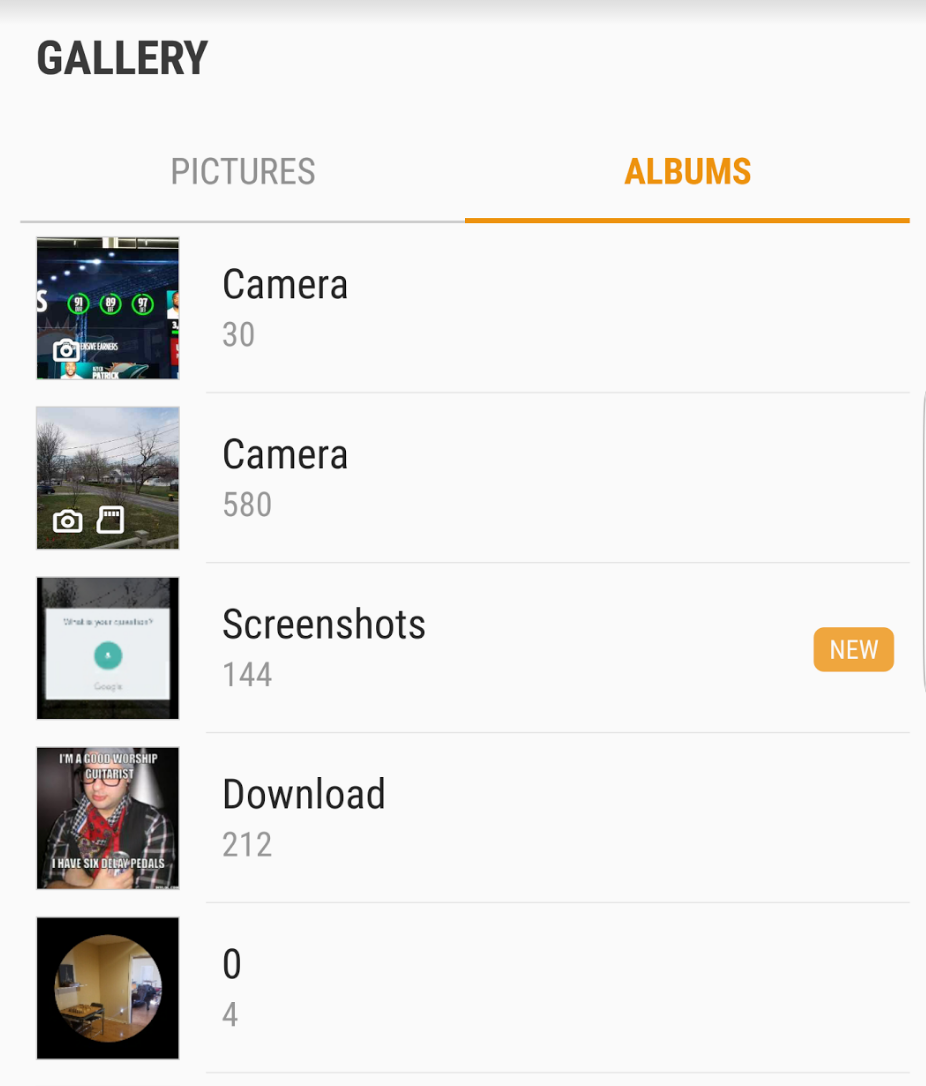
The implementation of CVRT begins with the user opening the application on their Android device. This presents them with a simple UI that has two options (as seen below):



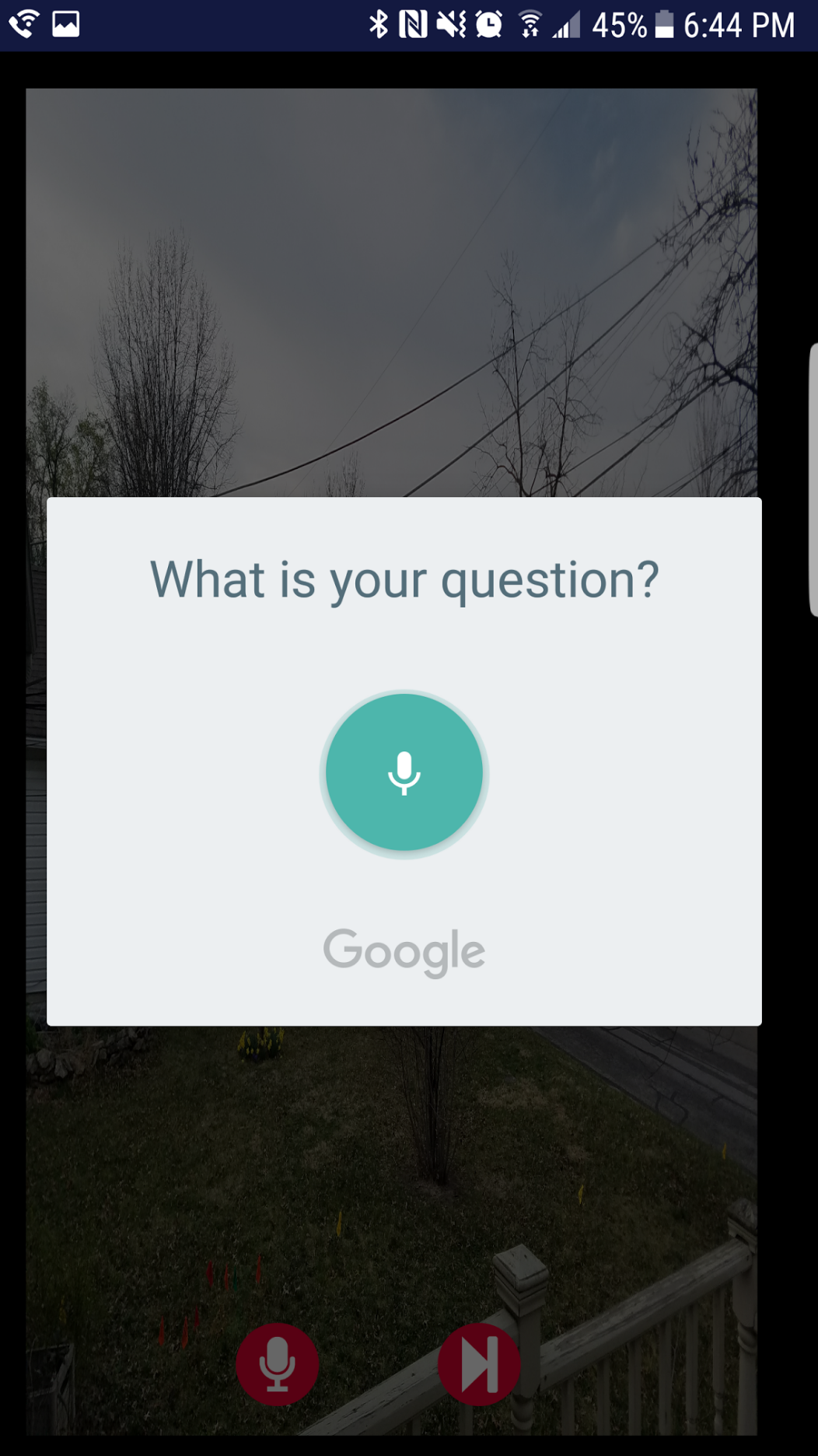
The button on the right acts as a ‘Skip’ option which prompts the user to select a new image from their phone’s gallery in case they are not interested in questioning the existing image. This gallery selection is implemented with the following algorithm upon clicking the button:



This will allow for increased flexibility as the user can choose any image that is on their phone. This functionality is exemplified by the screenshot below of the selection screen.



The button on the left side of the UI acts as a ‘Ask Question’ option and allows the user to verbally dictate a question that they have concerning the image. By clicking this option, the following dialog will display:



If no speech is recognized, then CVRT will prompt the user to speak again. Once speech has been detected, the speech is converted to a string and then stored within a variable. This ‘question’ variable, along with the image information, will be passed to the Python Server. The Android application will then create an Async task to leverage the tensorflow model for image analysis, and then provide an answer based on its’ findings. This answer will be verbally returned to the user on the main UI where they can choose to continue asking questions, thus repeating the process, until they are satisfied.

An example of this implementation is as follows. After choosing a picture in my phone’s gallery of a woman on a bicycle, I then tapped on the ‘Ask a Question’ button. This triggered the “What is your question?” speech prompt, which I answered with “Tell me about this picture.” After passing this to the Tensorflow model, the application provides the user with a verbal answer. Also, included with the verbal response is a toast message with the same results in case the user is deaf or hard of hearing. An example of these returned results can be seen in the following screenshot:



After the answer has been returned to the user; the image ID, question, and answer will be stored into a database table. This will then be leveraged by an HTML page that takes the database information, calls the Google Charting API, and then displays the report metrics. This web portal will be viewable to the patient as well as any family members/doctors that would be interested in the individual’s cognitive progress/decline.

**4.5 – Comparative Evaluation**

For the first iteration of CVRT work, the Clarifai API was used in order to generate the top captions related to the passed image. While this did have better overall accuracy than the tensorflow model in regards to basic image captioning, it was limited in regards to its inability of handling dynamic questions and providing appropriate answers. While the Clarifai API model only took a few seconds to call the API and also did not require a local webserver, its lack of VQA essentially relegated it to a foundation for the eventual classification path that CVRT would take. After transitioning to a tensorflow model, CVRT experienced lessened accuracy at around 50%, however this was in relation to dynamic questioning, as opposed to generic image captioning. This tensorflow implementation also required the python prediction script to be referenced over HTTP via Android. This required a local server to be setup that was previously not the case when dealing with the Clarifai API. However, similar functionality existed from the Android side as only one additional parameter (question) was necessary to be passed to the python server in comparison to Clarifai.

# Discussion & Limitation

In the current state of CVRT, some of the unanswered questions are more related to the medical side. The cognitive visual recognition tracking outlined above has very little research done to show its impact on a patient’s long-term/short-term mental health. While these metrics are interesting from a familial perspective to see the rate of a person’s cognitive decline, it is unproved as to whether these results would be useful to a medical professional given the large quantity of existing medical procedures. As for the data model, future studies could focus on further semantic analysis to determine the frequency of a patient’s vocabulary and how that might correlate with changes in their visual recognition ability.

# Conclusion

With CVRT, the VQA problem is presented through a new interface via Android. In conjunction with the tensorflow model and semantic analysis, it is able to take any user-selected image, process it’s features and the passed question, and then return an appropriate answer verbally, providing for an engaging experience for the user. While this model can be vague when answering certain types of questioning, it is able to provide a general answer for most question types and images that are presented to it. With the addition of the metrics platform that allows for truly data-driven results, my hope is that further research can be done to best exemplify possible treatment plans to better help anyone with this condition.

# References

* 1. Image Question Answering: A Visual Semantic Embedding Model and a New Dataset https://arxiv.org/pdf/1505.02074v1.pdf
  2. Visual Question Answering Demo: <http://visualqa.csail.mit.edu/>

2. Visual recognition memory in Alzheimer’s disease: repetition-lag effects: <https://www.ncbi.nlm.nih.gov/pubmed/18568983>

3. Impaired visual recognition memory predicts Alzheimer’s disease in amnestic mild cognitive impairment: <https://www.ncbi.nlm.nih.gov/pubmed/23572062>