RISK IDENTIFICATION FOR TEAM 06

1. Working with an OCR API – Adrian Lane
   1. Research the Azure cognitive services OCR API
   2. Create a small PoC to use as a reference for our app approach
   3. OCR Update - I'm able to use the API without issue in standard java using primarily HttpUrlConnection, and I can get a json response with the detected words from both a local image and a hosted image. I'm still working through how to get it all working within the Android framework. A couple of operations involve chaining API calls, so I'm still working through how to do that with threading and async but still in order, as well as how I might better encapsulate my code. More to come on that front. I think some of the work with the OpenWeatherAPI an Android has given me more ideas.
2. Working with a natural speech engine API – Graham Walker
   1. Research the Azure cognitive services Text to Speech API
      1. Update: Found a good guide from Azure: [Android PoC example](https://github.com/Azure-Samples/Cognitive-Speech-TTS/tree/master/Samples-Http/Android/TTSSample)
      2. Update: Java Azure TTS guide: [Java TTS PoC](https://github.com/Azure-Samples/Cognitive-Speech-TTS/blob/master/Samples-Http/Java/TTSSample/src/com/microsoft/cognitiveservices/ttssample/TTSService.java)
   2. Create a small PoC to use as a reference for our app approach
      1. Update: Created PoC around the Android Text to Speech API
         1. Update: Tried to implement the following guide above from Azure for Text to Speech (TTS), but ran into some issues. Will continue research trying to implement a PoC with Azure for project. – 6/1/19 GW
3. Cloud storage (database, blob, what?) – Ruben
   1. Dependent upon data model, but likely:
      1. Azure aBlob storage for stored images
      2. Azure Database for all other data persistence
4. Working with the camera/images API on the phone – Kenyon Bunker
   1. Update: Done - Research using the camera within the app - Done
   2. Research storing images from the camera elsewhere, (i.e. where our app would likely need to reference it from)
      1. Update: Internal and Blob
   3. Research accessing the photo album from within the app for our app’s use
      1. Update:
      2. It should only be visible to the app
      3. And images to be removed if the app is uninstalled
      4. Delete images on local device after a week
   4. Update: Create a small PoC to use as a reference for our app approach
      1. In progress
   5. Update: Good guide: <https://developer.android.com/guide/topics/media/camera>
   6. Update: High level steps that need to occur to use the camera
      1. Detect and Access Camera - Create code to check for the existence of cameras and request access.
      2. Create a Preview Class - Create a camera preview class that extends SurfaceView and implements the SurfaceHolder interface. This class previews the live images from the camera.
      3. Build a Preview Layout - Once you have the camera preview class, create a view layout that incorporates the preview and the user interface controls you want.
      4. Setup Listeners for Capture - Connect listeners for your interface controls to start image or video capture in response to user actions, such as pressing a button.
      5. Capture and Save Files - Setup the code for capturing pictures or videos and saving the output.
      6. Release the Camera - After using the camera, your application must properly release it for use by other applications.