How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text in green

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"

GitHub Username: Aniruddha-Tapas

Document Scanner

Description

This Document Scanner application turns your mobile into a portable scanner, which can be used for scanning handwritten notes and printed documents.
It automatically detect the edge of the paper over a contrasting surface.
When using the printed special page template it automatically detects the QR Code printed on the bottom right corner and scans the page immediately.
After the page is detected, it compensates any perspective from the image adjusting it to a 90 degree top view and saves it on a folder on the device.
It also incorporates OCR functionality which the user can use to detect text from documents and save them as editable text files in the external storage of the device.
It is also possible to launch the application from any other application that asks for a picture.

Intended User

Any individual, small businesses, organizations, governments and schools, that require to scan or detect text in handwritten or printed documents and convert them into images for personal or business purposes.

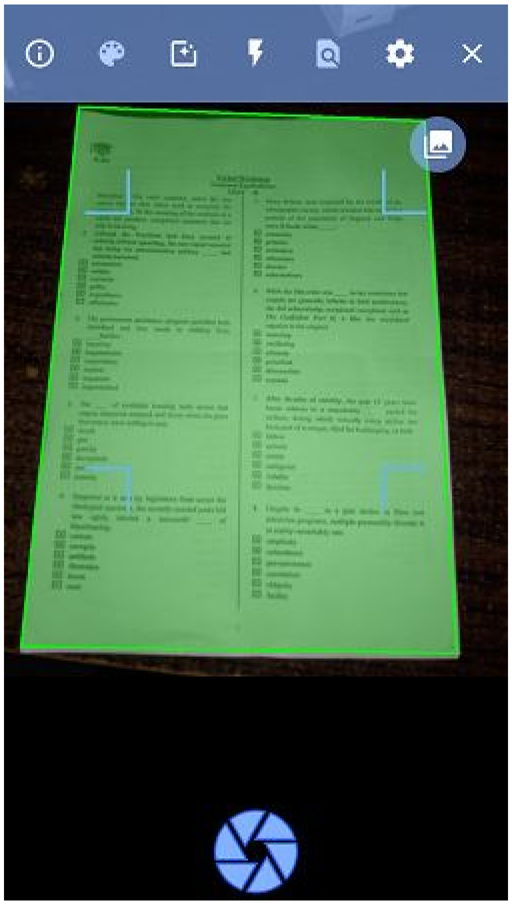
Features

Main features of Document Scanner app :

- Scan handwritten or printed documents
- Detects page frame and corrects perspective
- Fast and smooth Image Processing on the fly
- Scans are saved to your device as images
- Detect text using the OCR functionality of the app
- Save the detected texts as editable text files to your device.
- Easily share scanned docs with others via social media, email attachment or sending the doc link.

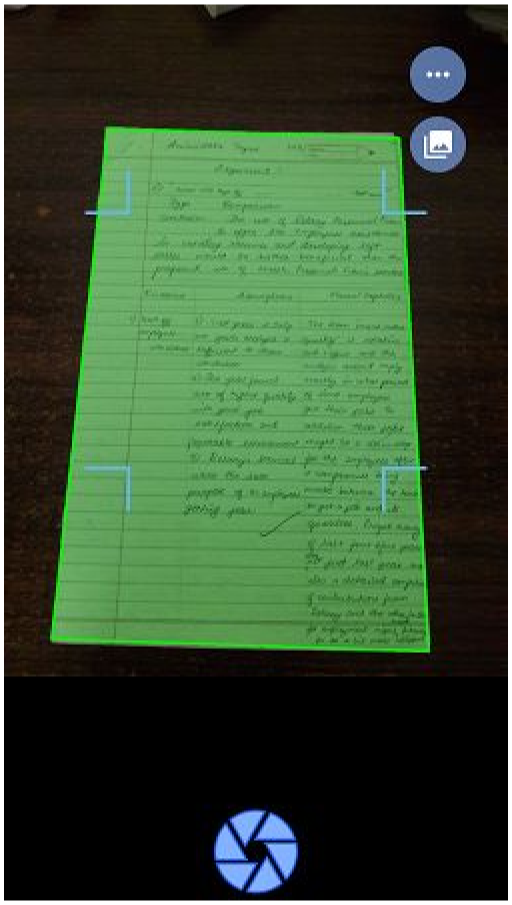
User Interface Mocks

Screen 1



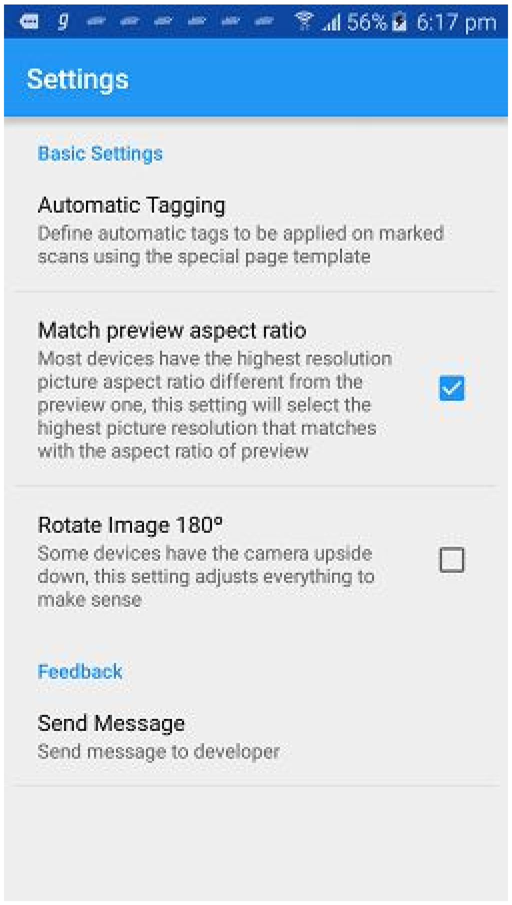
This is the main UI, with a camera view to scan the required document. It consists of a toolbar to display different options available for the scan. It also consists of a floating action button to view the scanned images. This button takes the user to an image gallery displaying the scanned documents.

Screen 2



The display of the toolbar can be toggled using a floating action button. Clicking this button displays or hides the complete toolbar after a smooth transition.

Screen 3



Settings

Basic Settings

Automatic Tagging
Define automatic tags to be applied on marked scans using the special page template

Match preview aspect ratio ✓
Most devices have the highest resolution picture aspect ratio different from the preview one, this setting will select the highest picture resolution that matches with the aspect ratio of preview

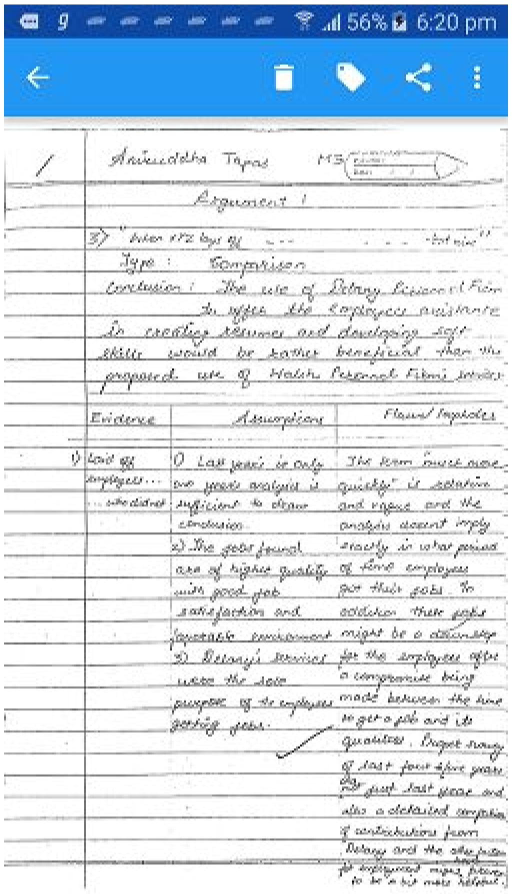
Rotate Image 180º ☐
Some devices have the camera upside down, this setting adjusts everything to make sense

Feedback

Send Message
Send message to developer

The Settings layout for the app.

Screen 4



Aniruddha Tapas M3

Argument 1

3) "[illegible] XYZ [illegible] ... [illegible]"

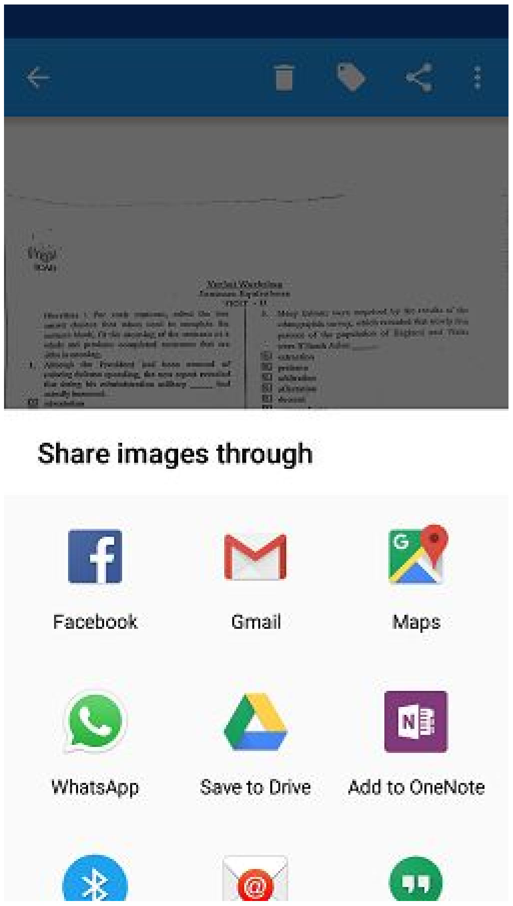
Type : Comparison

Conclusion : The use of Delany Personnel Firm to offer the employees assistance in creating resumes and developing soft skills would be rather beneficial than the proposed use of Walsh Personnel Firm's services

Evidence	Assumptions	Flaws/Loopholes
1) Laid off	1) Last year's is only	The term "found more
employees...	one year's analysis is	quickly" is relative
...who didn't	sufficient to draw	and vague and the
	conclusion.	analysis doesn't imply
	2) The jobs found	exactly in what period
	are of higher quality	of time employees
	with good job	got their jobs. In
	satisfaction and	addition their jobs
	favorable environment	might be a disadvantage
	3) Delany's services	for the employees after
	were the sole	a compromise being
	purpose of the employees	made between the time
	getting jobs.	to get a job and its
		qualities. Budget survey
		of last four & five years
		not just last year and
		also a detailed comparison
		of contributions from
		Delany and the other factors
		for employment might prove
		to be a bit more helpful.

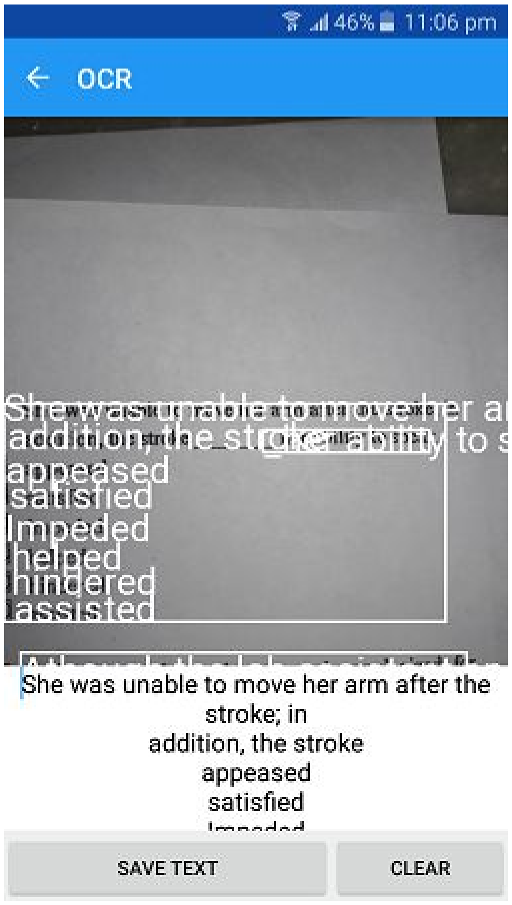
Example of a scanned handwritten document. The user can delete, tag or share the scanned image.

Screen 5



User can easily share scanned docs with others via social media, email attachment or sending the doc link.

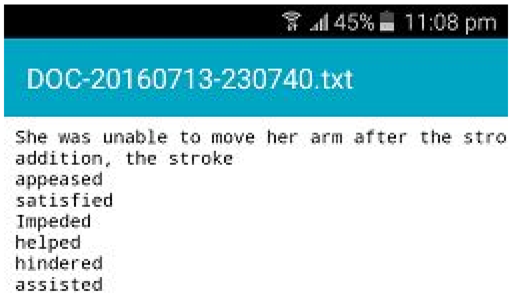
Screen 6



She was unable to move her arm after the stroke; in
addition, the stroke
appeased
satisfied

User can detect text from documents using the device camera.and has the choice to save the detected text to the device external storage.

Screen 7

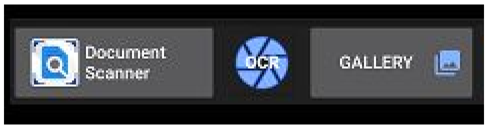


DOC-20160713-230740.txt

```
She was unable to move her arm after the stro
addition, the stroke
appeased
satisfied
Impeded
helped
hindered
assisted
```

The detected text using the OCR Activity is saved as editable text files in the folder .DocumentScanner/OCR-Texts created in the External Storage of the device.

Screen 8



The DocumentScanner App Widget. This consists of three buttons that would launch the 3 primary functionalities on click , making them quickly accessible off the homescreen/keyguard.

Key Considerations

How will your app handle data persistence?

The Document Scanner app will use the External Storage for saving files i.e captured images and will perform image processing techniques using OpenCV for Android and the Google Mobile Vision API on the images retrievable from the External Storage.

Describe any corner cases in the UX.

The app has 4 major layouts:

1. The main camera view layout.
2. The gallery grid layout.
3. The full screen image view layout.
4. The OCR capture layout

User can launch the gallery grid layout and the OCR capture layouts by clicking the floating action buttons on the main camera view layout. If the user clicks on a particular image in the gallery, he/she will view the full screen image view layout. These layouts consist of a back imagebutton which can be used to return to the respective previous activity. These back buttons have a similar functionality to the default back button of the android device.

Describe any libraries you'll be using and share your reasoning for including them.

1. OpenCV Android v3.1.0 for capturing and manipulation of images.

Google Services:

2. Google Zxing for barcode detection and image processing.
3. Google Mobile Vision Text API to see and understand text using OCR.
4. Google Analytics to measure user interaction with the app.

5. Android-Universal-Image-Loader for loading, caching and displaying images in the gallery grid layout.
6. FABToolbar for implementing a Floating Action Button transforming into toolbar.
7. Drag-select-recyclerview for Google Photos style multi-selection for RecyclerViews.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

- Configure libraries
- Integrate OpenCV with Android Studio
- Handle permissions requests for Camera, External Storage and Internet in the Android Manifest file.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for DocumentScannerActivity.
- Build UI for GalleryGridActivity.
- Build UI for FullScreenViewActivity.
- Build UI for OCRCaptureActivity.

Task 3: Implement the Document Scanning functionalities of the app

- Set up the user interaction to manually show or hide the system UI(i.e. status bar and navigation/system bar).
- Enable the Camera View.
- Create a surface using the best camera of the device for scanning documents.
- Detect edges of the document and perform perspective correction with the help of the surface created.
- Create a PreviewFrame using a Heads Up Display to highlight the document to be scanned.
- Process the PreviewFrame to check if it contains a QR Code.

Task 4: Handle image processing when user captures an image of the document

- Separate the PreviewFrame i.e. the scanned document in the image from its background using Background Reduction.
- Detect contours and lines using Hough Transformation.
- Detect which corner corresponds to which other corner, and then apply a perspective transformation to get just the page from the whole image.
- Enhance the document image using Adaptive Mean Threshing.

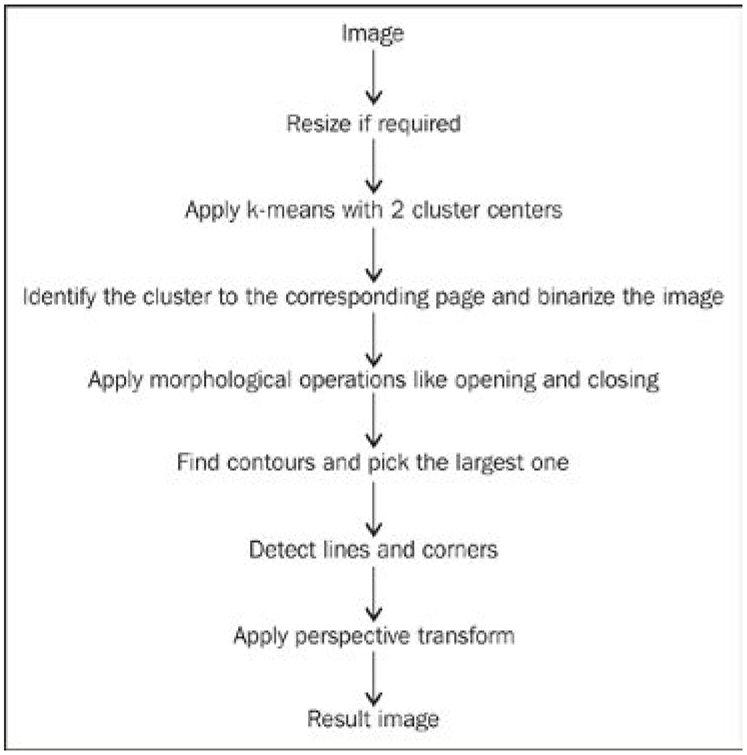


Image
Resize if required
Apply k-means with 2 cluster centers
Identify the cluster to the corresponding page and binarize the image
Apply morphological operations like opening and closing
Find contours and pick the largest one
Detect lines and corners
Apply perspective transform
Result image

Task 5: Post Document Processing

- Save the resulting image of the scanned document in a folder created in the External Storage of the device.
- Implement a Gallery Grid Activity that displays all the scanned documents by retrieving them from the external storage using the Universal Image Loader.
- Implement Full screen view of the scanned document when an image is clicked from the Gallery.
- Extend Android ImageView to include pinch zooming, panning, fling and double tap zoom.
- Enable sharing of the scanned document images via social media, email attachment or sharing the doc link.

Task 6: Implement the OCR functionalities of the app

- Set up the transition from DocumentScannerActivity to the OCRCaptureActivity via intents using a floating action button available on the main camera view.
- Integrate the Mobile Vision API in the app; Add the respective Google Play Services dependencies.
- Set up the TextRecognizer and CameraSource . Once it's initialized, a TextRecognizer can be used to detect text in all types of images.
- Create the OcrDetectorProcessor to read text straight from the camera.
- Draw the detected text on top of the camera preview.
- Make the app to save the detected text to an editable text files. Create folder and respective files in the external storage for this purpose and write to them the recognized text.

Task 7: Create an App Widget

- Create a quickly accessible app widget for the app.
- Add three main buttons to the widget.
- Implement OnClickListeners for the buttons to launch the 3 main activities of the app:
    1. DocumentScannerActivity
    2. OCRCaptureActivity
    3. GalleryGridAcrivity

Task 8: Integrate Google Analytics in the app

- Integrate Google Analytics to track and measure user interaction with the app
- Add the dependency for Google Play Services.
- Get a configuration file and add to the app project
- Add screen tracking.

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"