Department of Computer Science & Engineering

MINI PROJECT REPORT

(2020-21)

PDF MAKER Application

Institute of Engineering & Technology

Submitted by

Vaishnavi Chaurasia	(181500777)
Pratibha Sharma	(181500499)
Kapil Kumar	(181500308)

Github Project- PDF Maker

Supervised by

Mr. Neeraj Khanna

(Technical Trainer)

Department of computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,
Mathura – 281406

DECLARATION

We hereby declare that the work which is being presented in the Mini Project Report entitled "***PDF MAKER***" carried out by us in BTech (Computer Science & Engineering) in the 5th Semester is genuine and has been done by us under the general supervision of our supervisor Mr. Neeraj Khanna. It has not been submitted to any other university or degree program by us.

We also confirm that the report is only prepared for our academic requirements not for any other purpose.

Signatures

1. Vaishnavi Chaurasia
2. Pratibha Sharma
3. Kapil Kumar

Department of computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,
Mathura – 281406

ACKNOWLEDGEMENT

We would like to express our special thanks of gratitude to **Mr. Neeraj Khanna sir** who gave us this wonderful opportunity and guided us throughout this project. Throughout the journey of this project, our mentor helps us a lot in every aspect. I offer my sincere appreciation for the learning opportunity provided by GLA Management.

Our Completion of this Project Could have not been accomplished without the Support of our Teammates, friends, and Family.

Further on, we want to thank the students of this program who made this project joyful and always efficient**.**

Department of computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,
Mathura – 281406

Content

Abstract

"**PDF MAKER**" is an Android Application that converts images into PDF File Document format. This application is developed by using the *Android Studio IDE* with the k*otlin* And *XML* Language. This Application is made for the sake of users like Students, Teachers, and other professionals. It helps us to arrange the images or documents in a single place in a mannered way.

There are a Navigation Drawer and Buttons in which users can perform assigned functions respectively. Users can access gallery images of their device, the camera built on to our application to create a PDF file from them. In this project, we build an application that is capable of scanning images from gallery and camera and convert them into a specific document format called PDF. It allows multiple images to be compressed into a single document. Users can even change the language of the application.

These PDFs are secure and easy to carry, use, share, and upload to any platform or social sites. kotlin and XML language makes it possible to build this application. There are many languages available to build this application. Kotlin is more concise than other languages. It's easier to read and understandable for developers. Android Studio is the platform to

develop such applications. It is the official integrated development environment (IDE) for Google's Android operating system, built on

JetBrains IntelliJ IDE software, and designed specifically for Android development. In this application, we have used some predefined functions for performing different tasks like creating pdf, accessing images from the camera and gallery, and viewing all pdf files in one place.

For the development of this application, we used kotlin as it is way better than java in the field of application development because it's easy to use and more understandable. This application is made for the ease of the common user to share their data very easily.

Introduction

General Introduction to Topic

In this digital era, everyone wants to store their data which should consume less amount of storage as well as retrieval of data should take less time. They want their files to be in a compressed format that can be easily shared as well as uploaded. Nowadays everyone wants to share their data without any discrepancies and that is possible with our android application so We aim to fulfill these requirements with our android application.
For example, if a student wants to send the image of pages of the same book, if he will send each page it will confuse the sender as well as the receiver and it will take much time to download each page and put it in a sequence. Through this application, users can store their images in one PDF and can share them easily within less time and in a sequenced manner. Our application will be used for professional purposes. It will help users to secure their documents in one place. Whereas if we are talking about assignment solutions given by college students that data must be in a pdf file because this is the only file format which is supported by their portals and it's very easy to upload the data using pdf files.

Literature Survey

Introduction

We wanted to make an application that is capable of scanning images and convert them into PDF files. It allows multiple images to be compressed into a single document. These PDF's are secure and easy to use, share, and upload to any platform or social sites. This app can resize images, rotate images to left/right, and many more features are there that we will describe later.

Problem Statements

We have created a mobile-based Android Application. We have developed a single-platform free application that can convert image files into PDF files. It can only run on android mobiles.

We planned to develop this app due to some problems:-

1. **Storage**
    a. Users want fewer storage files in their mobile phones so they prefer compressed documents in spite of a bunch of images.

2. Accuracy

a. Compressed documents (PDF) are very helpful in professional as well as Student Life.

   i. As a student, the books issued in the library can not be carried by them outside the campus after the time limit of 10-15 days, so they prefer PDF's instead of books.

   ii. As a professional guy, it's not easy to remember every information related to their job or to carry important documents handy so they prefer compressed documents like PDF as it is secure and lightweight.

3. Redundancy

a. Users want one-click share and download in spite of sharing individual images of similar content or type one by one.

4. Consistency

Users want their important documents to be in one place whenever they search for them. So, our application would be the best platform to put all important images in a document and create different folders for them

5. Security issues

Data must be stored on a secured platform. One of the security features can be a password-protected file. So, PDF is a trustworthy document that provides a facility to protect their data by using their own passwords.

6. Shareable

a) Users can share their PDFs through Whatsapp, Bluetooth, Easyshare, Instagram, and other shareable devices.

Implementation Idea

This idea has been implemented by others.

Analysis

During the research and analysis of our Project, we learned about new concepts, application features, and implementation ideas. We explored a pre-implemented app named Image to Pdf Converter. Based on our research, we came to know about many useful features of the pre-implemented application which are very useful.

Some Features are listed below:-

- Select one or more images from gallery
- Capture image from camera in our application
- Change Language of Application
- Application works offline
- Easy to use

Implementation of our Project

While studying and analyzing the pre-implemented application, and having looked at it practically we gathered some of the basic features which are either not implemented properly or they do not have considered in their application.

We had also worked on the design of the application and many other features that will be beneficial for the users. Our Application is compatible with the current software. There are some new features.

- Available in both languages - English and Hindi
- See all your created PDF in this Application

Future Scope

We can add many more features to our application when we require them.

- Put Password to PDF - Users can put a password to their document to protect it from other users. It will be an optional feature. There will be no restriction on the password.
- Different Image layouts - Users can choose different layouts of their image while scanning to make it clear and visible.
- Change Image quality in pixels - Users can change the quality of an image in pixels (unit) before converting it into PDF.
- QR Code Scanner - User can scan a QR code i.e Quick Response Code and know every information about the product. For example, A student can scan his book and get every detail about that book.

Software Requirement specification (SRS)

Introduction

***Purpose*:** The purpose of our document is to specify an android application that can convert an image into a PDF file.

Scope: This document enumerates the requirements of application software. This application will run offline. This application can change a simple image into a lightweight compressed document. This application is reusable, flexible, and easy to use.

Overview

In this digital era, everyone wants to store their data which should consume less amount of storage as well as retrieval of data should take less time. They want their files to be in a compressed format that can be easily shared as well as uploaded. We aim to fulfill these requirements with our android application. For example, if a student wants to send the image of pages of the same book, if he will send

page it will confuse the sender as well as the receiver and it will take much time to download each page and put it in a sequence. Through this application, users can store their images in one PDF and can share it easily within less time and in a sequenced manner.

Customer

Everyone can use this application without any age restrictions.

Development Responsibility

Our Team members worked on this android application development.

Functional Requirements

- Users can select multiple images from the gallery
- Users can scan image through camera and it can be displayed in gallery
- PDF are with no watermark
- Our application will work offline

- User can easily share and open the PDF
- Our application is available in both languages - English and Hindi
- Users can choose the application theme (Dark and Light)
- User can see all created PDF through this Application

Platform

Our Application is an android based mobile application.

Non Functional Requirement

- Usability Requirement

The application can be easily accessed by using android mobile. Since all users are familiar with the process. The application is user-friendly.

- Availability Requirement

This application can be easily available. It can be operated by users at any time 24/7.

- Accuracy

The application will be developed by considering the real-time issues in the functionality and the operations.

- Performance Requirement

The application will perform its functions according to the inputs given by the user.

- Reliability Requirement

The application will be reliable due to the importance of data stored in it. No user will face any issue in the provided features.

- Efficiency Requirement-

The application will be efficient enough in its role.

Hardware Requirement

- Android Mobile

- Laptop - We need a Laptop to perform the coding part of the application.

- USB cable - USB cable is used to run our application and install it to our device.

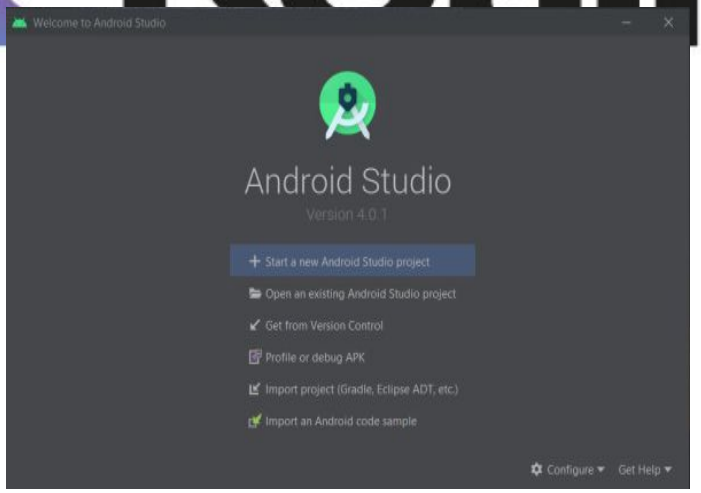
Software Requirement

We will develop our android application by using *Android Studio*. It is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains IntelliJ IDE software, and designed specifically for Android development.

Languages Used

Markup Language - XML (Extensible Markup Language)

Programming Language - Kotlin



Implementation Of Application

Part-1: Designed User InterFace wireframes to display the basic layout of an application.

A wireframe is a static blueprint of an application layout. It displays the flow of activities in an application. It helps a developer to understand the flow of information as well as the sequence of the layout.

There are many wireframe tools that help to design layout in a pictorial way. These tools contain some elements which help in designing our layout. They also have the facility to show the flow of activities through linking of screens of Android mobile.

Some of the common elements are:-

- Android Mobile
- About
- Buttons
- Header
- Navigation Bar
- All pdf's List.

Part-2: Implement User Interface by using Android Studio and XML Files

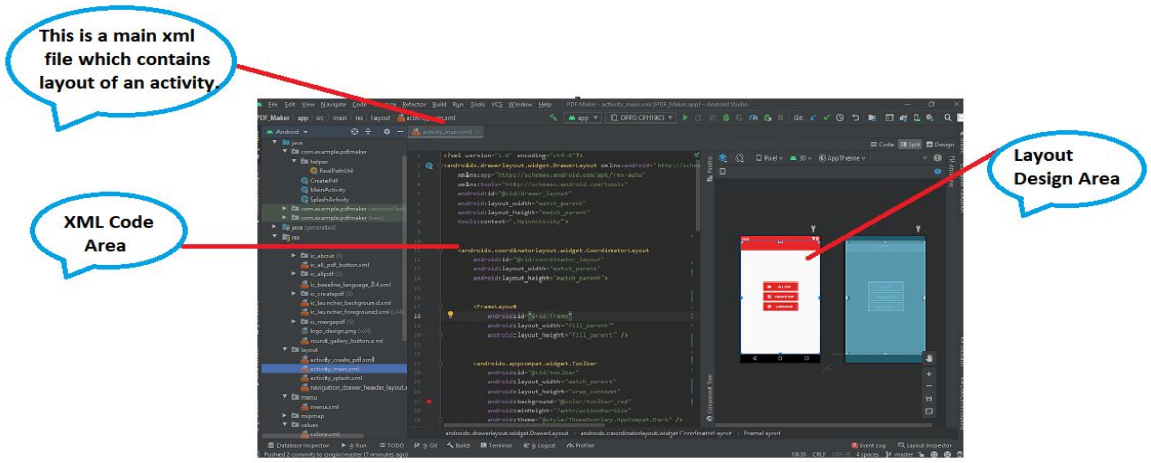
With the help of WireFrames i.e the blueprint of layout It will be easy to implement the actual User Interface of the application. By using XML we can design our layout.
Android Studio IDE is used to make the layout of our application.

Firstly, we made xml file by using following steps :-

Step-1 : Create xml file

Right click on res folder → Android Resource File → Write File name (any) & Resource File as layout → Click Ok

Step-2 : Write XML code by using tags in the code area or directly design layout through the design tab in the right hand side.



Layout Code of our Application

➢ activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">


    <androidx.coordinatorlayout.widget.CoordinatorLayout
        android:id="@+id/coordinator_layout"
        android:layout_width="match_parent"
        android:layout_height="match_parent">


        <FrameLayout
            android:id="@+id/frame"
            android:layout_width="fill_parent"
```

```
                android:layout_height="fill_parent" />


    <android.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/toolbar_red"
        android:minHeight="?attr/actionBarSize"
        android:theme="@style/ThemeOverlay.AppCompat.Dark" />



</androidx.coordinatorlayout.widget.CoordinatorLayout>

<com.google.android.material.navigation.NavigationView
    android:id="@+id/navigation_view"
    android:layout_width="match_parent"

    android:layout_height="match_parent"
    android:layout_gravity="start"
    app:headerLayout="@layout/navigation_drawer_header_layout"
    app:menu="@menu/menu" />

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:orientation="vertical"
    android:layout_centerVertical="true"
    android:id="@+id/mainXmlLinearLayout"
    >

    <Button
        android:layout_width="180dp"
        android:layout_height="50dp"
        android:layout_marginBottom="10dp"
        android:text="@string/all_pdf"
        android:textColor="#FFFFFF"
        android:textSize="18sp"
        android:drawableLeft="@drawable/ic_allpdf"
        android:background="@color/toolbar_red"
        android:textAllCaps="true"
        android:textStyle="bold|italic"
        android:paddingLeft="20dp"
```

```
                android:paddingStart="20dp"
            tools:ignore="RtlSymmetry"
            android:drawableStart="@drawable/ic_allpdf"
            android:onClick="openFolder"/>

        <Button
            android:layout_width="180dp"
            android:layout_height="50dp"
            android:layout_marginBottom="10dp"
            android:text="@string/create_pdf"
            android:textColor="#ffffff"
            android:background="@color/toolbar_red"
            android:textSize="18sp"
            android:textAllCaps="true"
            android:textStyle="bold|italic"
            android:paddingLeft="20dp"
            android:paddingStart="20dp"
            tools:ignore="RtlSymmetry"

            android:id="@+id/createPDFButton"
            android:drawableLeft="@drawable/ic_createpdf"
            android:drawableStart="@drawable/ic_createpdf"
            android:onClick="gotoCreatePdfActivity"/>

        <Button
            android:layout_width="180dp"
            android:layout_height="50dp"
            android:layout_marginBottom="10dp"
            android:id="@+id/Switch_Language"
            android:textColor="#ffffff"
            android:background="@color/toolbar_red"
            android:textSize="18sp"
            android:textAllCaps="true"
            android:textStyle="bold|italic"
            android:paddingLeft="20dp"
            android:paddingStart="20dp"
            tools:ignore="RtlSymmetry"
            android:drawableStart="@drawable/ic_baseline_language_24"
            android:drawableLeft="@drawable/ic_baseline_language_24"
            android:text="@string/switch_language" />

    </LinearLayout>
</androidx.drawerlayout.widget.DrawerLayout>
```

- Here, we have used Drawer Layout as the root of the layout. The child views are Coordinator Layout, Frame Layout and Toolbar.
- Drawer Layout → It is the root ViewGroup of the Activity Layout
- Navigation View → It is used for the setup of the Navigation drawer for our application with header and menu.
- Linear Layout → The views inside this will be either horizontal or vertical in constraints.
- Buttons → On Clicking these buttons respective functions will be performed.

Header layout of the Navigation drawer

➢ **Navigation_drawer_header_layout.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="300dp"
    android:background="@color/toolbar_red"
    android:orientation="vertical">

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/logo_design"
        android:layout_marginLeft="50dp"
        android:layout_marginStart="50dp"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
                android:text="@string/about"
        android:layout_marginLeft="50dp"
        android:layout_marginStart="50dp"
        android:textSize="25sp"
        android:textColor="@color/white"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/white"
        android:text="@string/aboutApp"
        android:layout_marginLeft="50dp"
        android:layout_marginStart="50dp"
        android:layout_marginTop="10dp"
        android:textStyle="italic"/>

</LinearLayout>
```

- First Activity of our application is Splash Activity. It is a constant screen which appears for a specific amount of time, generally shows for the first time when the app is launched. The Splash screen is used to display some basic introductory information such as the company logo, content, etc just before the app loads completely.

➢ **Activity_splash.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".SplashActivity">

    <ImageView
        android:id="@+id/splash_background"
        android:layout_width="wrap_content"
```

```
                android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/logo_design"></ImageView>


</androidx.constraintlayout.widget.ConstraintLayout>
```

- The Second Activity Layout of our Application with two buttons allows the user to select images from gallery and capture image from gallery and the image will automatically saved to phone gallery respectively.

➢ Activity_create_pdf.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".CreatePdf">


   <ImageButton
       android:id="@+id/cameraButton"
       android:layout_width="104dp"
       android:layout_height="104dp"
```

```
                android:layout_marginTop="200dp"
        android:background="@drawable/round_gallery_button"
        app:layout_constraintBottom_toTopOf="@+id/galleryButton"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.113"
        app:srcCompat="@android:drawable/ic_menu_camera"
        android:onClick="CameraButton"/>

    <ImageButton
        android:id="@+id/galleryButton"
        android:layout_width="100dp"
        android:layout_height="104dp"
        android:layout_marginBottom="244dp"
        android:background="@drawable/round_gallery_button"

        android:onClick="convertToPdf"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.504"
        app:layout_constraintStart_toStartOf="parent"
        app:srcCompat="@android:drawable/ic_menu_gallery" />

    </androidx.constraintlayout.widget.ConstraintLayout>
```

➢ Strings.xml (English)

```
<resources>
    <string name="app_name" translatable="false">PDF MAKER</string>

    <!--open drawer string-->
    <string name="open_nav_drawer" translatable="false">Open Navigation
Drawer</string>
    <!-- close drawer string-->
    <string name="close_nav_drawer" translatable="false">Close Navigation
Drawer</string>
    <string name="all_pdf">ALL PDF</string>
    <string name="pdf_maker" translatable="false">PDF MAKER</string>
    <string name="create_pdf">CREATE PDF</string>
```

```
    <string name="merge_pdf">MERGE PDF</string>
   <string name="Language">Language Changed</string>
   <string name="switch_language">Language</string>
   <string name="aboutApp">PDF MAKER is an Android Application that converts
images into PDF File Document format.</string>
   <string name="about">About</string>
</resources>
```

➢ Strings.xml (Hindi)

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <string name="all_pdf">सभी पीडीएफ देखें</string>
   <string name="create_pdf">पीडीएफ बनाएं</string>
   <string name="switch_language">भाषा स्विच करें</string>
   <string name="merge_pdf">पीडीएफ विलय</string>
   <string name="Language">भाषा स्विच</string>
    <string name="aboutApp">" PDF MAKER एक एंड्रॉयड एप्लीकेशन है जो छवियों को PDF
 फ़ाइल दस्तावेज़ प्रारूप में परिवर्तित करता है।"</string>
   <string name="about">एप के बारे में</string>
</resources>
```

➢ Styles.xml

```
<resources>
   <!-- Base application theme. -->
   <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
       <!-- Customize your theme here. -->
```

```
                <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/toolbar_red</item>
        <item name="colorAccent">@color/colorAccent</item>

        <item name="windowActionBar">false</item>
        <item name="windowNoTitle">true</item>
    </style>

    <!-- Splash Application Theme -->
    <style name="SplashTheme" parent="Theme.AppCompat.Light.NoActionBar">
        <!--No Action Bar on Splash Screen-->
        <item name="windowActionBar">false</item>
        <!-- No Title Display on Splash Screen-->
        <item name="windowNoTitle">true</item>
    </style>
</resources>
```

➢ **colors.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#6200EE</color>
    <color name="colorPrimaryDark">#3700B3</color>
    <color name="colorAccent">#03DAC5</color>
    <color name="splash_screen_white">#fff</color>

    <!-- Toolbar color-->
    <color name="toolbar_red">#F3E61F1F</color>

    <!-- Button background color-->
    <color name="button_background_Color">#F3E61F1F</color>
    <color name="white">#FFFFFFFF</color>
</resources>
```

➢ **dimens.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <dimen name="nav_header_height">50dp</dimen>
</resources>
```

- Icons used

➢ **Drawable → Ic_all_pdf_button.xml**

```
<vector android:height="34dp"
   android:tint="#FFFFFF"
   android:viewportHeight="24"
   android:viewportWidth="24"
   android:width="34dp"
   xmlns:android="http://schemas.android.com/apk/res/android">
   <path
       android:fillColor="@android:color/white"


 android:pathData="M20,2L8,2c-1.1,0 -2,0.9 -2,2v12c0,1.1 0.9,2 2,2h12c1.1,0
2,-0.9 2,-2L22,4c0,-1.1 -0.9,-2 -2,-2zM11.5,9.5c0,0.83 -0.67,1.5
-1.5,1.5L9,11v2L7.5,13L7.5,7L10,7c0.83,0 1.5,0.67 1.5,1.5v1zM16.5,11.5c0,0.83
-0.67,1.5 -1.5,1.5h-2.5L12.5,7L15,7c0.83,0 1.5,0.67
1.5,1.5v3zM20.5,8.5L19,8.5v1h1.5L20.5,11L19,11v2h-1.5L17.5,7h3v1.5zM9,9.5h1v-1
L9,8.5v1zM4,6L2,6v14c0,1.1 0.9,2 2,2h14v-2L4,20L4,6zM14,11.5h1v-3h-1v3z" />
</vector>
```

➢ **Drawable → ic_baseline_language_24.xml**

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
  android:width="24dp"
  android:height="24dp"
  android:tint="#FFFFFF"
  android:viewportWidth="24"
  android:viewportHeight="24">
  <path
      android:fillColor="@android:color/white"
      android:pathData="M11.99,2C6.47,2 2,6.48 2,12s4.47,10
9.99,10C17.52,22 22,17.52 22,12S17.52,2
11.99,2zM18.92,8h-2.95c-0.32,-1.25 -0.78,-2.45 -1.38,-3.56 1.84,0.63
3.37,1.91 4.33,3.56zM12,4.04c0.83,1.2 1.48,2.53
1.91,3.96h-3.82c0.43,-1.43 1.08,-2.76 1.91,-3.96zM4.26,14C4.1,13.36
4,12.69 4,12s0.1,-1.36 0.26,-2h3.38c-0.08,0.66 -0.14,1.32 -0.14,2 0,0.68
0.06,1.34 0.14,2L4.26,14zM5.08,16h2.95c0.32,1.25 0.78,2.45 1.38,3.56
```

```
            -1.84,-0.63 -3.37,-1.9 -4.33,-3.56zM8.03,8L5.08,8c0.96,-1.66
      2.49,-2.93 4.33,-3.56C8.81,5.55 8.35,6.75 8.03,8zM12,19.96c-0.83,-1.2
      -1.48,-2.53 -1.91,-3.96h3.82c-0.43,1.43 -1.08,2.76
      -1.91,3.96zM14.34,14L9.66,14c-0.09,-0.66 -0.16,-1.32 -0.16,-2 0,-0.68
      0.07,-1.35 0.16,-2h4.68c0.09,0.65 0.16,1.32 0.16,2 0,0.68 -0.07,1.34
      -0.16,2zM14.59,19.56c0.6,-1.11 1.06,-2.31 1.38,-3.56h2.95c-0.96,1.65
      -2.49,2.93 -4.33,3.56zM16.36,14c0.08,-0.66 0.14,-1.32 0.14,-2 0,-0.68
      -0.06,-1.34 -0.14,-2h3.38c0.16,0.64 0.26,1.31 0.26,2s-0.1,1.36
      -0.26,2h-3.38z" />
      </vector>
```

➢ **Drawable → ic_createpdf**

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
   android:width="24dp"
   android:height="24dp"
   android:viewportWidth="24"
   android:viewportHeight="24"
   android:tint="#FFFFFF"
   android:alpha="0.8">
 <path
     android:fillColor="@android:color/white"
     android:pathData="M19,3L5,3c-1.11,0 -2,0.9 -2,2v14c0,1.1 0.89,2
2,2h14c1.1,0 2,-0.9 2,-2L21,5c0,-1.1 -0.9,-2
-2,-2zM17,13h-4v4h-2v-4L7,13v-2h4L11,7h2v4h4v2z"/>
</vector>
```

➢ **Drawable → round_gallery_button.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
```

```
                <item>

        <shape android:shape="oval">
            <solid android:color="@color/button_background_Color"></solid>
            <size android:height="180dp" android:width="180dp"></size>
        </shape>

    </item>


</selector>
```

Part-3: Implementation of Logic of our application

By using programming language we can implement the basic logic of our application. Here we use Kotlin as our programming language. As we all know, logic is the main building block of an application, if you don't know the logic you can not proceed further. The logic depends on the features of our application.

- Select image from gallery
- Convert them into PDFs
- View all created PDF's
- Capture an image using camera
- Switch Language in Hindi or English

➢ **Com.example.pdfmaker → helper → RealPathUtil.kt**

URI stands for Uniform Resource Identifiers a compact sequence of characters that identifies a particular resource. All URI follows some predefined syntax rules. For Example - http://

This is a helper Object. Here, we tried to get the real path of the image from URI in different APIs of Android Version. API stands for Application Programming Interface which helps two applications to communicate with each other. Every Android device has a different version corresponding to a different API level.
That's why we have made three different functions for different API level as mentioned below:-

getPathFromURI_API19 → For API level 19
getRealPathFromURI_API11to18 → For API level 11 to 18
getRealPathFromURI_BelowAPI11→ For API level below 11

```
package com.example.pdfmaker.helper
```

```
import android.annotation.SuppressLint
import android.content.Context
import android.content.CursorLoader
import android.net.Uri
import android.provider.DocumentsContract
import android.provider.MediaStore

object RealPathUtil {

    /** Annotation by Android Lint Tool **/
    @SuppressLint("NewApi")

            /** API-19 **/
    fun getPathFromURI_API19(context: Context, uri: Uri): String? {
        var filePath: String? = ""
        if (DocumentsContract.isDocumentUri(context, uri)) {
            // Will return "image:x*"
            val documentId = DocumentsContract.getDocumentId(uri)
            // Split at colon, use second item in the array
            val idArraySplit = documentId.split(":").toTypedArray()
            if (idArraySplit.size == 2) {
                val id = idArraySplit[1]
                val column = arrayOf(MediaStore.Images.Media.DATA)

                // where id is equal to
                val sel = MediaStore.Images.Media._ID + "=?"
                      // Media provider to access all Image type files on the
External Storage
                // contentResolver used to access data in content provider
                val cursor = context.contentResolver.query(
                      MediaStore.Images.Media.EXTERNAL_CONTENT_URI,      // The
content:// style URI for the "primary" external storage volume
                    column, sel, arrayOf(id), null
                )
                val columnIndex = cursor!!.getColumnIndex(column[0])
```

```
                        // move the cursor to first row
                if (cursor.moveToFirst()) {
                    filePath = cursor.getString(columnIndex)
                }
                // cursor is closed
                cursor.close()
            }
        } else {
            filePath = uri.path
        }
        return filePath
    }


    @SuppressLint("NewApi")

            /** API - 11 to 18 **/
     fun getRealPathFromURI_API11to18(context: Context?, contentUri: Uri?):
String? {
        val projection =
            arrayOf(MediaStore.Images.Media.DATA)  // projection is an array of
columns that should be included for each row retrieved.
        var result: String? = null
        // loader that queries the ContentResolver and returns a Cursor
         val cursorLoader = CursorLoader(context, contentUri, projection, null,
null, null)
        val cursor = cursorLoader.loadInBackground()
        // when cursor not equal null
        if (cursor != null) {
            // column index
                                                    val     column_index     =
cursor.getColumnIndexOrThrow(MediaStore.Images.Media.DATA)
            // cursor will move to first row
            cursor.moveToFirst()
            result = cursor.getString(column_index)
```

```
                        }
        return result
    }

    /** Below API - 11 **/

fun getRealPathFromURI_BelowAPI11(context: Context, contentUri: Uri?): String
{
        val projection = arrayOf(MediaStore.Images.Media.DATA)
          val cursor = context.contentResolver.query(contentUri!!, projection,
null, null, null)
                                                    val       column_index       =
cursor!!.getColumnIndexOrThrow(MediaStore.Images.Media.DATA)
        cursor.moveToFirst( return cursor.getString(column_index)}}
```

- Our Application consists of three Activities which users will go through while creating a PDF document.

Splash Activity → Main Activity → CreatePDF Activity

First Activity → Splash Screen Activity

```
package com.example.pdfmaker

import android.content.Intent
import android.os.Bundle
import android.os.Handler
import android.os.Looper
import android.appcompat.app.AppCompatActivity

/*
Splash Screen is the First Activity of our Application
*/
class SplashActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
```

```
                super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_splash)

        // Handler used to send and process Message
        Handler(Looper.getMainLooper()).postDelayed({
            /* Intent to start the Main Activity*/
            startActivity(Intent(this, MainActivity::class.java))
            finish()
            // 1000ms = 1sec Duration time of Splash Screen
        }, 1000)
    }}
```

Second Activity → Main Screen Activity

```
package com. Example.pdf maker

import android.content.Context
import android.content.Intent
import android.content.res.Configuration
import android.net.Uri
import android.os.Bundle
import android.os.Environment
import android.view.MenuItem
import android.view.View
import android.widget.Button
import android.widget.FrameLayout
import androidx.appcompat.app.ActionBarDrawerToggle
import androidx.appcompat.app.AlertDialog
import android.appcompat.app.AppCompatActivity
import android.appcompat.widget.Toolbar
import androidx.coordinatorlayout.widget.CoordinatorLayout
import androidx.core.view.GravityCompat
import androidx.drawerlayout.widget.DrawerLayout
import com.google.android.material.navigation.NavigationView
import java.util.*

class MainActivity : AppCompatActivity() {

    // lateinit is used as promise that the variable will be initialised later
    private lateinit var toolbar: Toolbar
    lateinit var drawerLayout: DrawerLayout
    lateinit var coordinatorLayout: CoordinatorLayout
```

```
                lateinit var frameLayout: FrameLayout
    lateinit var navigationView: NavigationView
    lateinit var languageButton: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        /****  id of Views are initialised to the variables  ****/
        toolbar = findViewById(R.id.toolbar)
        drawerLayout = findViewById(R.id.drawer_layout)
        frameLayout = findViewById(R.id.frame)
        coordinatorLayout = findViewById(R.id.coordinator_layout)
        navigationView = findViewById(R.id.navigation_view)
        languageButton = findViewById(R.id.Switch_Language)

        // function call
        setUpToolbar()

        /**** Tie the functionality of DrawerLayout and the framework ActionBar
****/
        val actionBarDrawerToggle = ActionBarDrawerToggle(
            this@MainActivity,
            drawerLayout,
            R.string.open_nav_drawer,
            R.string.close_nav_drawer
        )

        drawerLayout.addDrawerListener(actionBarDrawerToggle)
        actionBarDrawerToggle.syncState()

        /** Language Change on Button Click **/
        languageButton.setOnClickListener {
            showchangelanguage()
        }

    }

    /****    Toolbar Setup    ****/
    private fun setUpToolbar() {
        setSupportActionBar(toolbar)
        // Title set on toolbar
        supportActionBar?.setTitle("PDF MAKER")
        // Get a support ActionBar corresponding to this toolbar and enable the
Up button
```

```
                supportActionBar?.setHomeButtonEnabled(true)
        supportActionBar?.setDisplayHomeAsUpEnabled(true)
    }

    override fun onOptionsItemSelected(item: MenuItem): Boolean {
        val id = item.itemId
        if (id == android.R.id.home) {
            drawerLayout.openDrawer(GravityCompat.START)

        }
        return super.onOptionsItemSelected(item)
    }


/** Language **/

    fun showchangelanguage() {
        val listlanguage = arrayOf("Hindi", "English")
        val mBuilder = AlertDialog.Builder(this@MainActivity)
        mBuilder.setTitle("Choose Language")
        mBuilder.setSingleChoiceItems(listlanguage, -1) { dialog, which ->
            if (which == 0) {
                setLocate("hi")
                recreate()
            } else if (which == 1) {
                setLocate("en")
                recreate()
            }
            dialog.dismiss()
        }
        val mDialog = mBuilder.create()
        mDialog.show()
    }

    fun setLocate(Lang: String) {
        val locale = Locale(Lang)
        Locale.setDefault(locale)
        val config = Configuration()
        config.locale = locale
                        baseContext.resources.updateConfiguration(config,
baseContext.resources.displayMetrics)

                        val    editor    =    getSharedPreferences("settings",
Context.MODE_PRIVATE).edit()
        editor.putString("My_Lang", Lang)
```

```
                editor.apply()
    }

    fun gotoCreatePdfActivity(view: View) {
        //kotlin code to go to next activity
        startActivity(Intent(this, CreatePdf::class.java))
    }

    fun openFolder(view: View) {
        val intent = Intent(Intent.ACTION_VIEW)

val mydir = Uri.parse(Environment.getExternalStorageState())
        intent.setDataAndType(mydir, "*/*")
        startActivity(intent)}}
```

Third Activity → CreatePdf Activity

```
package com.example.pdfmaker
import android.Manifest
import android.app.Activity
import android.content.Intent
import android.content.pm.PackageManager
import android.graphics.BitmapFactory
import android.graphics.pdf.PdfDocument
import android.net.Uri
import android.os.Build
import android.os.Bundle
import android.os.Environment
import android.os.StrictMode
import android.provider.MediaStore
import android.util.Log
import android.view.View
import android.widget.Toast
import androidx.annotation.RequiresApi
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.ActivityCompat
import com.example.pdfmaker.helper.RealPathUtil
import java.io.*
class CreatePdf : AppCompatActivity() {
```

```
                    // Image URI
    private var file: Uri? = null
    // PDF document
    var pdfDocument: PdfDocument? = null
Val                                  directory                                    =
Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DCIM)
        .toString() + "/Camera/"
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_create_pdf)


 /** Request Permission to Read and Write to External Storage **/
        ActivityCompat.requestPermissions(
             this,arrayOf(Manifest.permission.READ_EXTERNAL_STORAGE,          //
Request permission to read
                Manifest.permission.WRITE_EXTERNAL_STORAGE            // Request
permission to write
            ), PackageManager.PERMISSION_GRANTED
        )
        // Function call
        createFilesDirectory()
        ActivityCompat.requestPermissions(
            this,
                                          arrayOf(Manifest.permission.CAMERA,
Manifest.permission.WRITE_EXTERNAL_STORAGE),
            PackageManager.PERMISSION_GRANTED
        )
        val builder = StrictMode.VmPolicy.Builder()
        StrictMode.setVmPolicy(builder.build())
    }
    fun CameraButton(view: View?) {
        val file = "$directory$randomName.jpg"
        val newFile = File(file)
        try {
```

```
                    newFile.createNewFile()
        } catch (e: IOException) {
            e.printStackTrace()
        }
        val outputFileUri = Uri.fromFile(newFile)
        val cameraIntent = Intent(MediaStore.ACTION_IMAGE_CAPTURE)
        cameraIntent.putExtra(MediaStore.EXTRA_OUTPUT, outputFileUri)
        startActivity(cameraIntent)
    }
    private fun createFilesDirectory() {
        // Directory for all files
          val rootPath = File(Environment.getExternalStorageDirectory(), "PDF
MAKER Files")
        // If path does not exist
        if (!rootPath.exists()) {
            // will make a directory for it
            rootPath.mkdirs()
        }
         //Creates a new File instance from a parent abstract pathname and a
child pathname string.
        val dataFile = File(rootPath, "test file")
         // if current state of the primary "external" storage device != state
at which media is present and mounted
        if (Environment.getExternalStorageState() != Environment.MEDIA_MOUNTED)
{
            // Toast is a message in the form of popup to the user
                    Toast.makeText(this, "Cannot use External Storage",
Toast.LENGTH_SHORT)
            finish()
            return
        }
        try {
            val mOutput = FileOutputStream(dataFile, false)
            val data = "DATA"
            mOutput.write(data.toByteArray())
```

```
                    mOutput.close()
            } catch (e: FileNotFoundException) {
                e.printStackTrace()
            } catch (e: IOException) {
                e.printStackTrace()
            }
            try {
                val mInput = FileInputStream(dataFile)
                val data = ByteArray(128)
                mInput.read(data)
                mInput.close()
            } catch (e: FileNotFoundException) {
                e.printStackTrace()
            } catch (e: IOException) {
                e.printStackTrace()
            }
            dataFile.delete()
        }
        // object that is common to all instances of this class like static in java
        companion object {
            // Assume any positive integer type Number
            const val IMAGE_PICK_CODE = 1
        }
        /** Check whether Permission granted by user **/
        @RequiresApi(Build.VERSION_CODES.KITKAT)
         public override fun onActivityResult(requestCode: Int, resultCode: Int,
data: Intent?) {
            Log.d("onActivityResult: ", "Activity result came out")
            // Multiple Image selection from phone gallery
            val clipData = data!!.clipData
            //Iff user permits
             if (requestCode == IMAGE_PICK_CODE && resultCode == Activity.RESULT_OK
&& clipData != null) {
                // creates a document
                pdfDocument = PdfDocument()
```

```
                        // loop till data (image) counter ends
            for (i in 0 until clipData.itemCount) {
                // uri of each data (image)
                file = clipData.getItemAt(i).uri
                // converted to bitmap (an image file format)
                val bitmap = BitmapFactory.decodeFile(uriToFilename(file))
                // create a page description
                     val pageInfo = PdfDocument.PageInfo.Builder(bitmap.width,
bitmap.height, i + 1).create()
                // start a Page
                val startPage = pdfDocument!!.startPage(pageInfo)
                // draw something on the page
                startPage.canvas.drawBitmap(bitmap, 0f, 0f, null)
                // finish Page
                pdfDocument!!.finishPage(startPage)
            }
            // location of PDF document and a random name to it
                                                    val     pdfFileDir     =
"/storage/emulated/0/MY_PDF_CONVERTER/$randomName.pdf"
            Log.d("onActivityResult: ", pdfFileDir)
            val pdfFiles = File(pdfFileDir)
            try {
                pdfDocument!!.writeTo(FileOutputStream(pdfFiles))
            } catch (e: IOException) {
                e.printStackTrace()
            }
            // document closed
            pdfDocument!!.close()
        }

        // if single image is picked
        else if (requestCode == IMAGE_PICK_CODE && resultCode == RESULT_OK) {
            file = data.data
            // Repeat the same process for one image
            /** data (image) --> Bitmap --> Page --> PDF Document **/
```

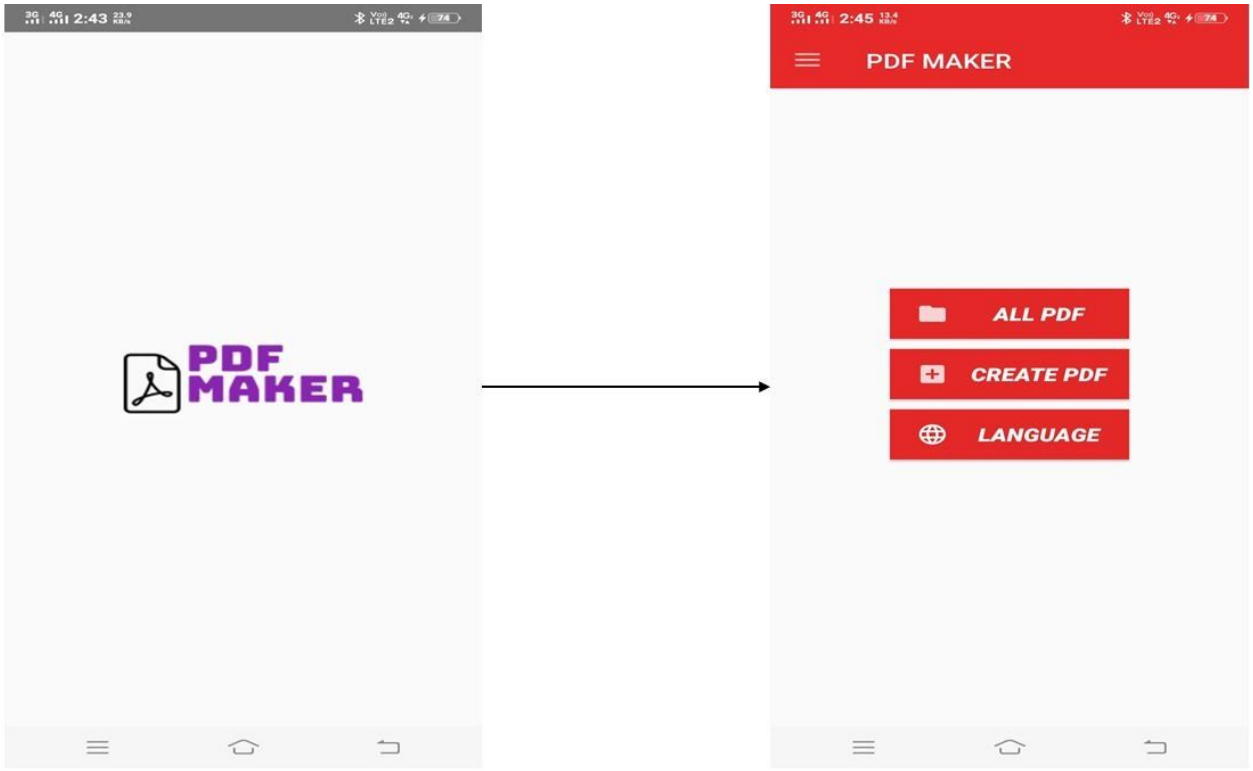
```
                                                    val     bitmap     =
BitmapFactory.decodeFile(uriToFilename(file))
            val pdfDocument = PdfDocument()
                    val  pageInfo  =  PdfDocument.PageInfo.Builder(bitmap.width,
bitmap.height, 1).create()
            val startPage = pdfDocument.startPage(pageInfo)
            startPage.canvas.drawBitmap(bitmap, 0f, 0f, null)
            pdfDocument.finishPage(startPage)
                            val   pdfFileDir   =   "/storage/emulated/0/PDF   Maker
Files/$randomName.pdf"
            Log.d("onActivityResult: ", pdfFileDir)
            val pdfFiles = File(pdfFileDir)
            try {
                pdfDocument.writeTo(FileOutputStream(pdfFiles))
            } catch (e: IOException) {
                e.printStackTrace()
            }
            pdfDocument.close()
        }
        super.onActivityResult(requestCode, resultCode, data)
    }

    // function to convert image uri to filename
    private fun uriToFilename(uri: Uri?): String? {
        var path: String? = null
        path = if (Build.VERSION.SDK_INT < 11) {
            // function call from RealPathUtil Object
            RealPathUtil.getRealPathFromURI_BelowAPI11(this, uri)
        } else if (Build.VERSION.SDK_INT < 19) {
            RealPathUtil.getRealPathFromURI_API11to18(this, uri)
        } else {
            RealPathUtil.getPathFromURI_API19(this, uri!!)
        }
        // Path created
        return path
```

```
                }

    // Random name assigned to PDF file name
    private val randomName: String
        private get() {
            val AlphaNumericString = ("ABCDEFGHIJKLMNOPQRSTUVWXYZ"
                    + "0123456789"
                    + "abcdefghijklmnopqrstuvxyz")
            val sb = StringBuilder(5)
            for (i in 0..4) {
                val index = (AlphaNumericString.length
                        * Math.random()).toInt()
                sb.append(AlphaNumericString[index])
            }
            return sb.toString()
        }


    override fun onRestart() {
        super.onRestart()
        super.onResume()
                        Toast.makeText(this,   "PDF   Created   Successfully",
Toast.LENGTH_SHORT).show()
    }


    fun convertToPdf(view: View?) {
        val intent = Intent()
        intent.type = "image/*"
        intent.putExtra(Intent.EXTRA_ALLOW_MULTIPLE, true)
        intent.action = Intent.ACTION_GET_CONTENT
         startActivityForResult(Intent.createChooser(intent, "Select Picture"),
IMAGE_PICK_CODE)
    }
```

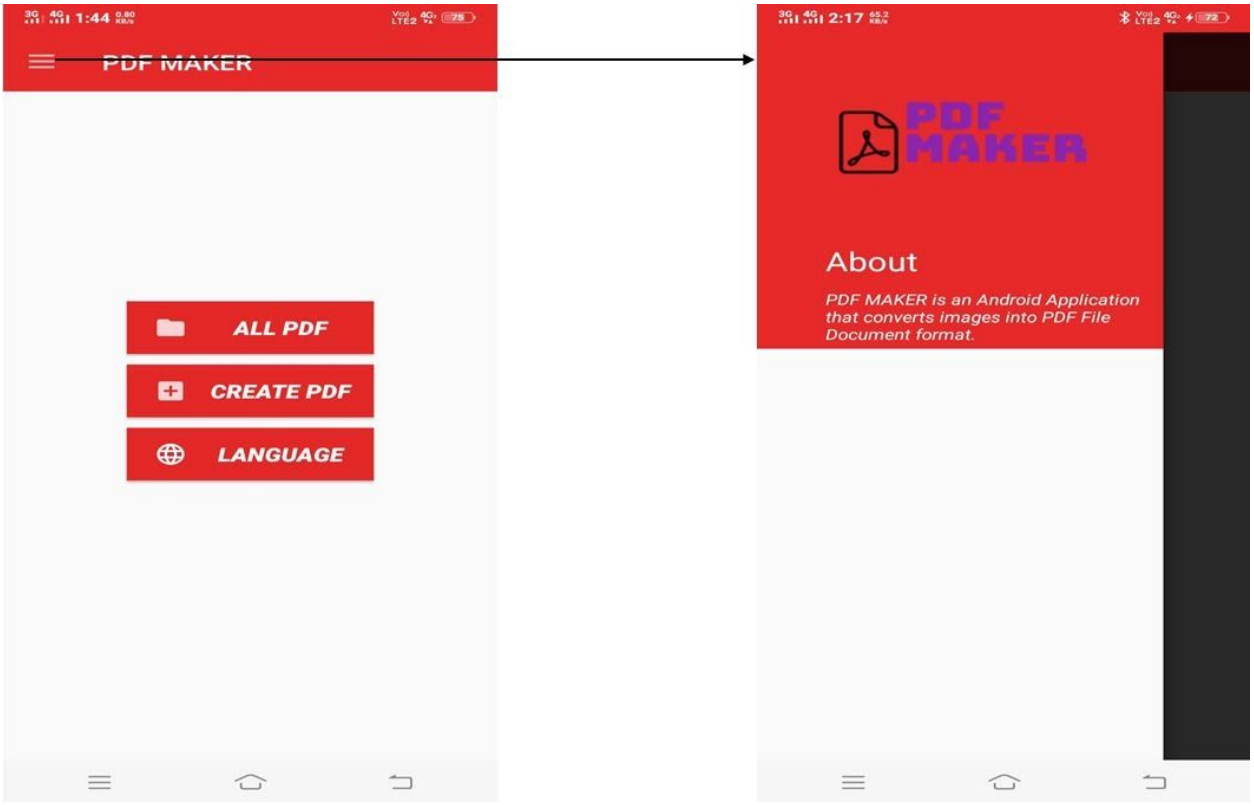
}

Working of Application

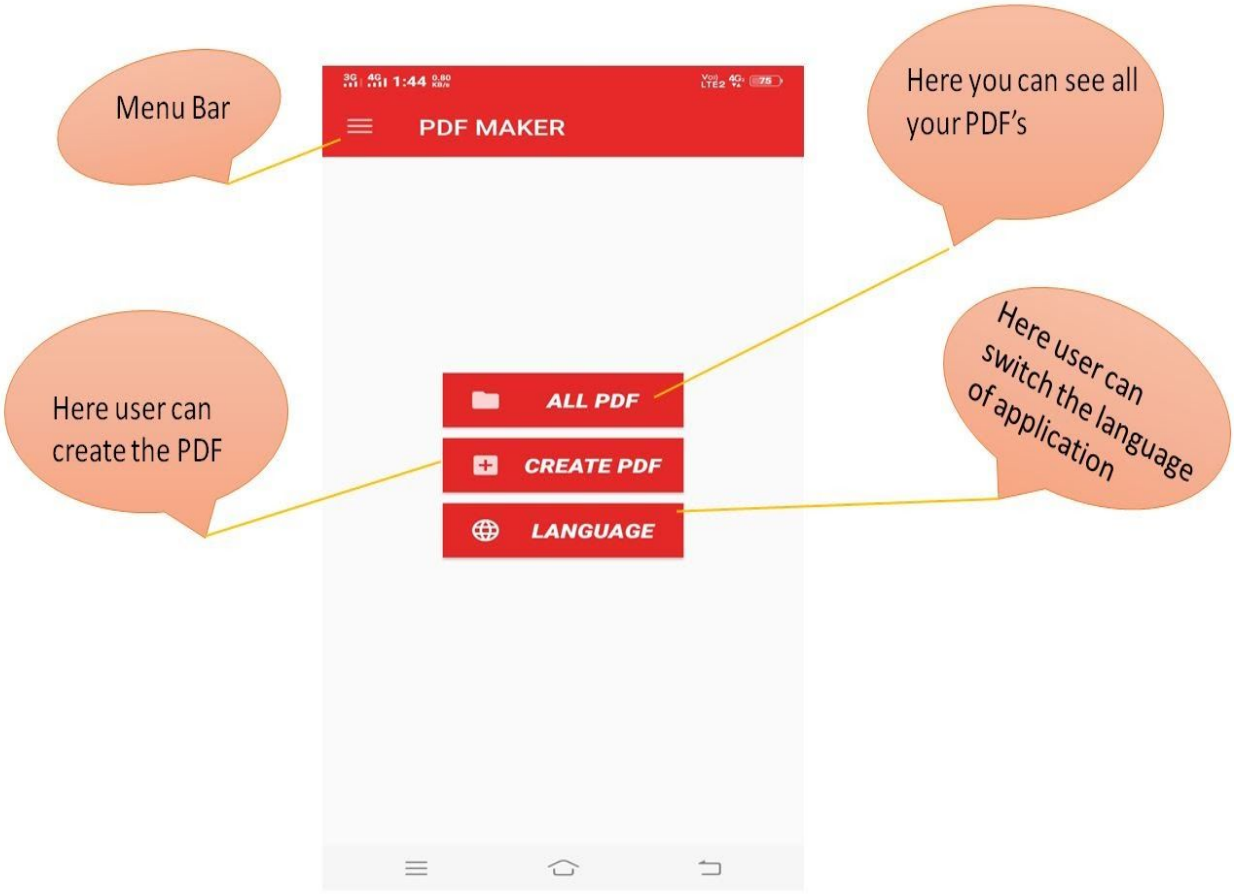
Now we will explain the working of our application entitled “PDF MAKER “. Initially, the user has to go through the first splash screen activity of our application. The splash screen is the first screen that is visible to the user. The background process of the application runs through this screen and to keep the interest of the user developers put their application logo or title on to it. It displays to the user for a couple of seconds.



In our application, there is a navigation drawer which can be opened by a click on the hamburger icon on the top left-hand side of the main screen. On click, on the icon, the drawer slides to the right-hand side overlapping the main screen. Generally, it is used to keep the clickable menu items one after the other (optional). On top of the screen, there is a toolbar that is red in color displays the title of our application. It can also contain clickable items (optional). At the center of the main screen, there are three buttons with specific text and icons to it. When the user will press these buttons they will automatically perform their respective functionality.



Let's start with the functionality of the *first button (All PDF)*. It is used to view all PDFs created by our application. When the user will click on this button a popup will appear which shows the External storage contents of their android device.

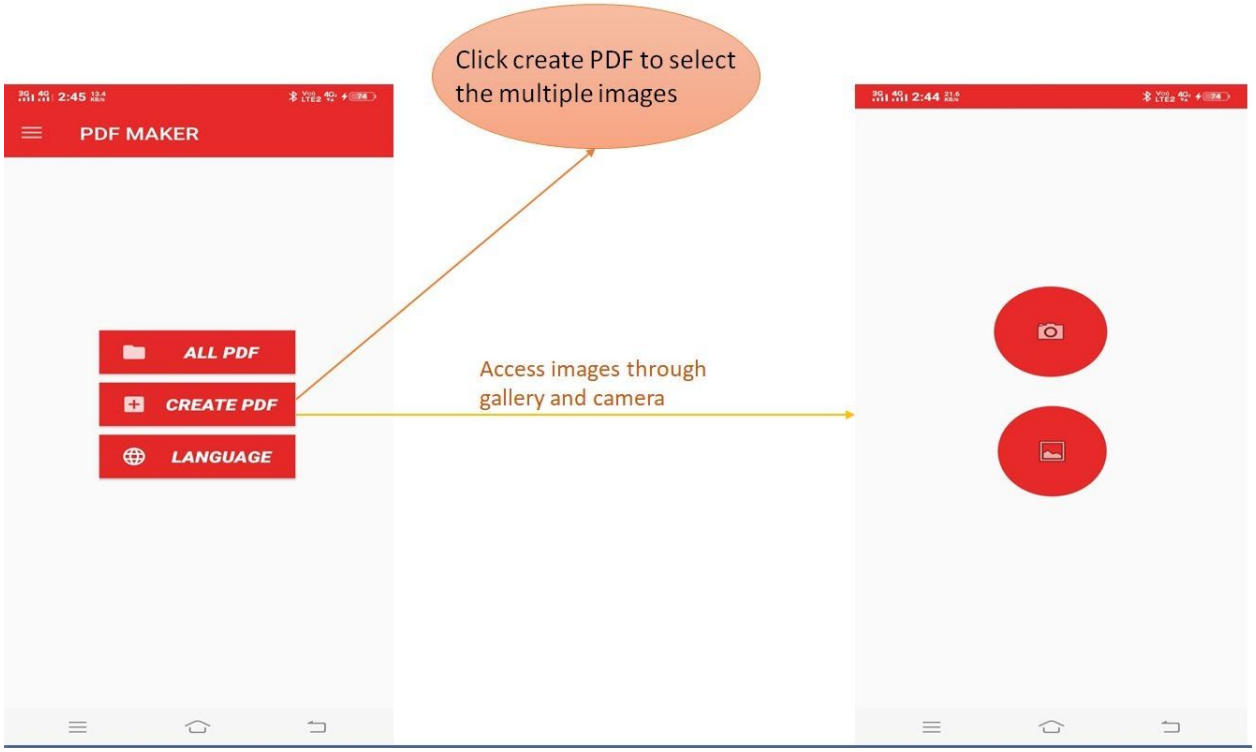


The procedure to view PDF files is:-

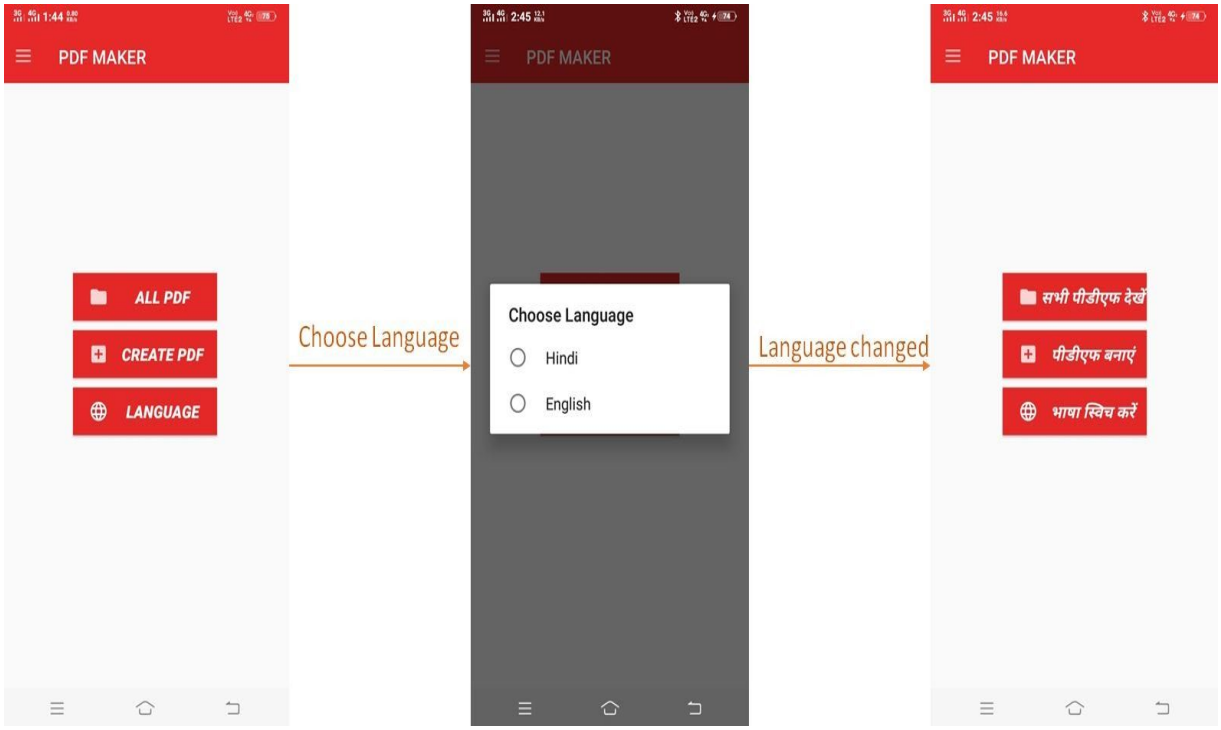
All PDF button → Files → External storage directory→"PDF MAKER Files" folder → View your created PDF

*** **PDF Maker Files** is the name of the folder where users can view all created PDFs by our application.

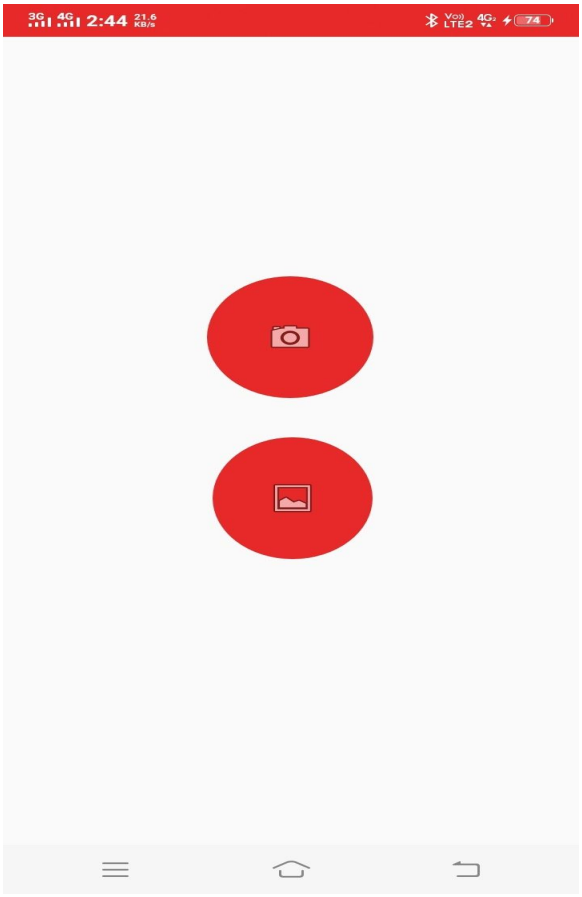
The functionality of the *second button* (Create PDF) is to proceed to the next activity (last activity) where the user can select an image or capture an image using a camera.



The functionality of the *third button (Language)* is to switch the language of our application. Our application allows users to change the language from English to Hindi or vice versa.



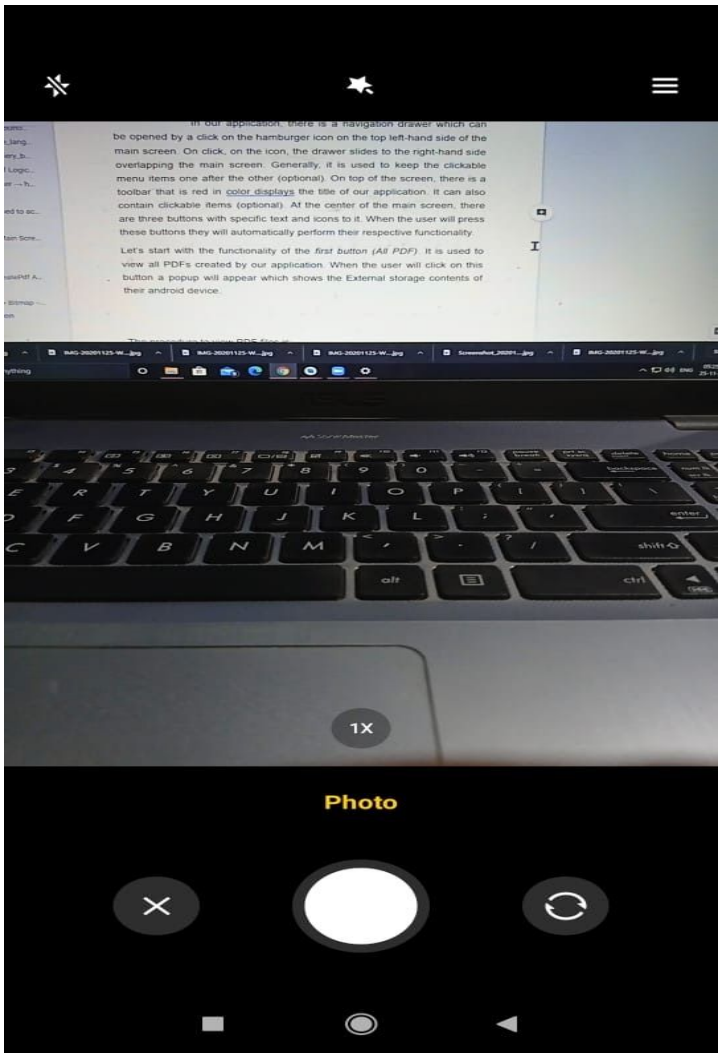
The last activity consists of two round red buttons which perform the main function for which this application is designed



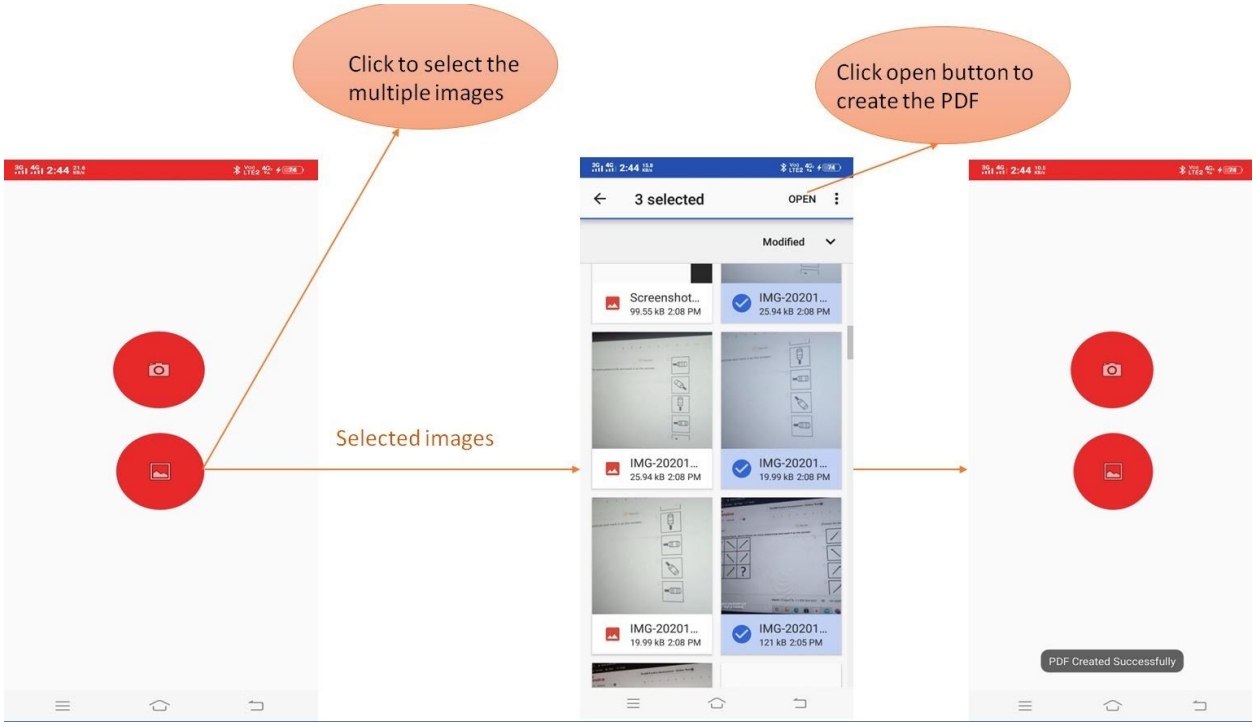
The functionality of the first button is to proceed to the camera built on our application to capture an image and save it to the phone gallery. The device camera is linked to our application. After capturing the image, the user can again create Pdf through the gallery access button.

.

The functionality of the second button is to select the images from the gallery. As soon as the user selects one or multiple images, the application

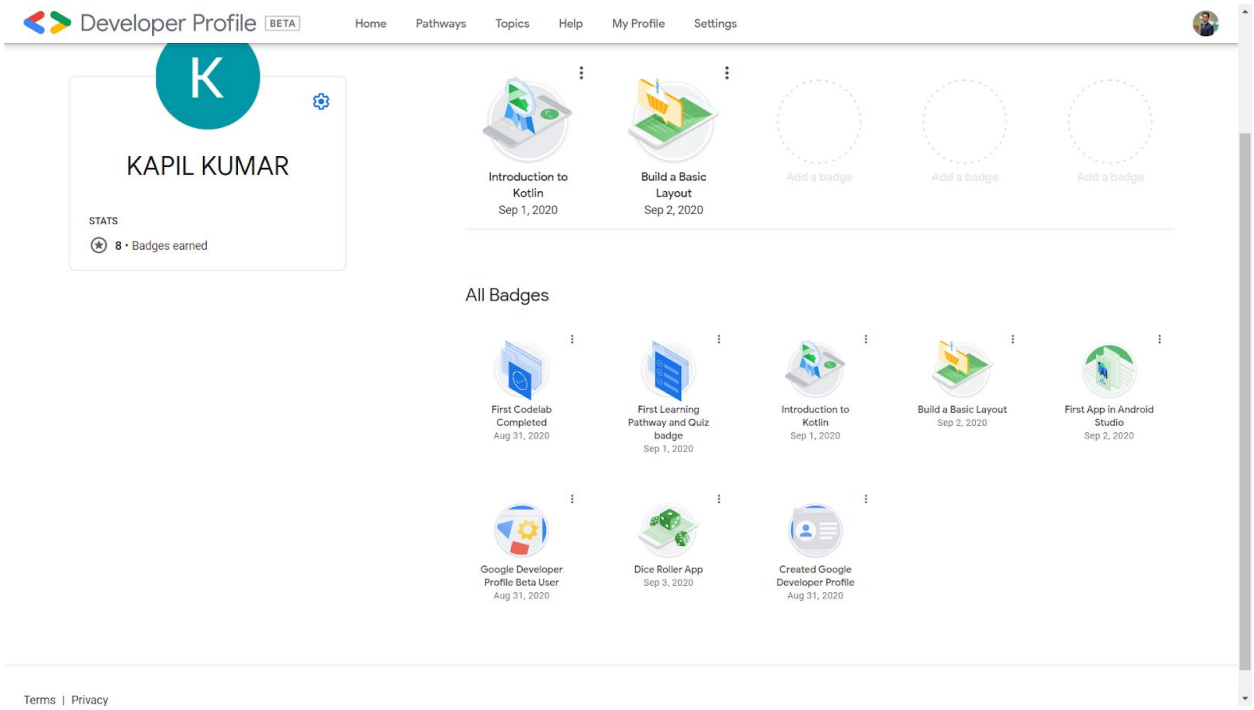


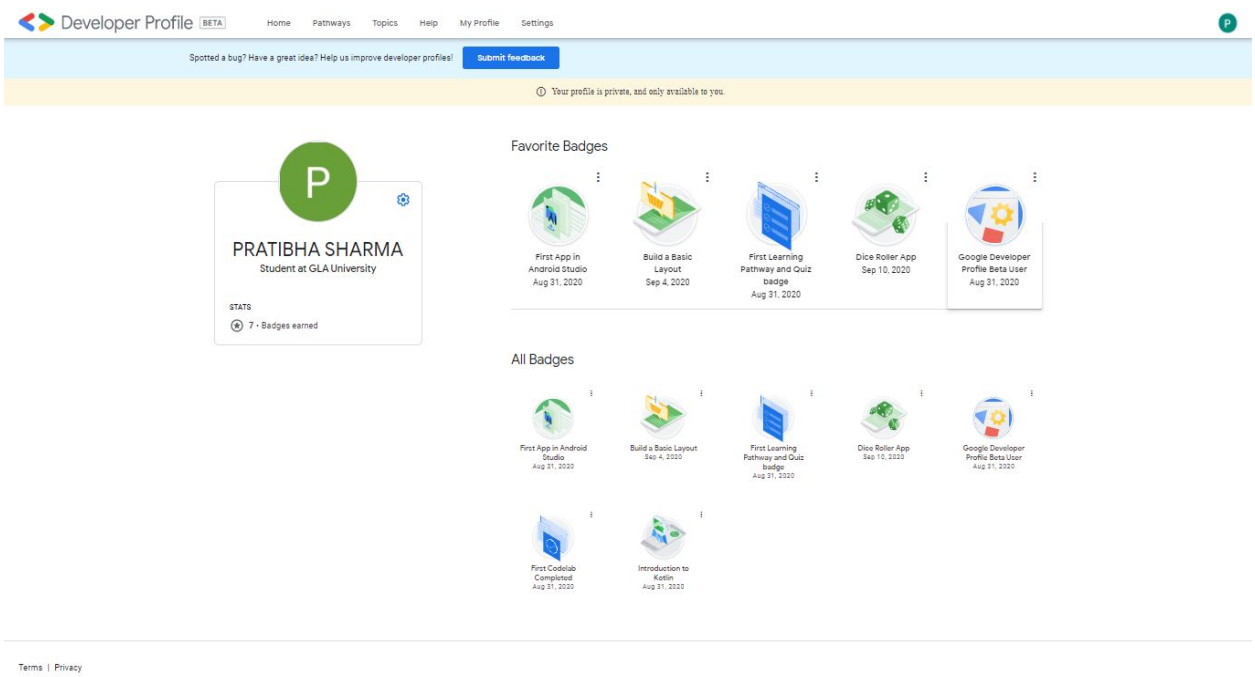
will automatically create a PDF and save it to the external storage folder named PDF Maker Files. The user can create a PDF of single as well as multiple images at once.

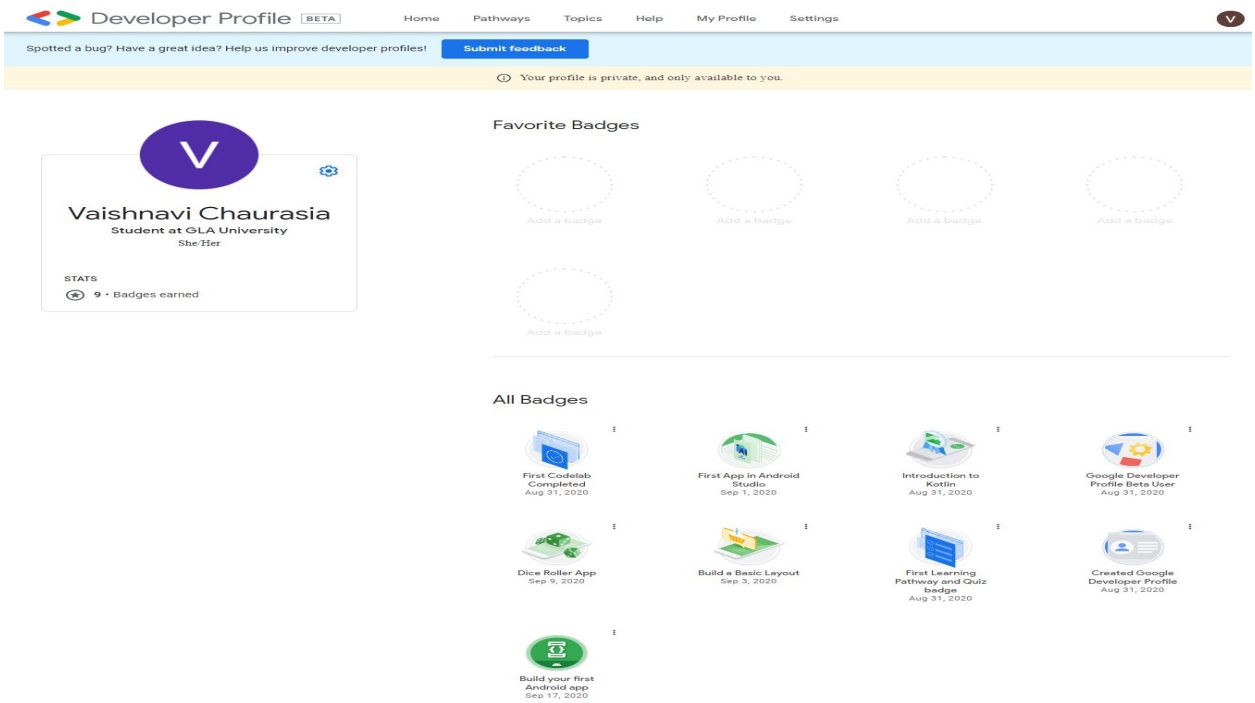

Click to select the multiple images
Click open button to create the PDF
Selected images
3 selected
OPEN
Modified
Screenshot...
99.55 kB 2:08 PM
IMG-20201...
25.94 kB 2:08 PM
IMG-20201...
19.99 kB 2:08 PM
IMG-20201...
121 kB 2:05 PM
PDF Created Successfully

Department of computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha, Mathura – 281406

Certificates (Badges)






Developer Profile BETA
Home
Pathways
Topics
Help
My Profile
Settings
Spotted a bug? Have a great idea? Help us improve developer profiles!
Submit feedback
Your profile is private, and only available to you.
V
Vaishnavi Chaurasia
Student at GLA University
She/Her
STATS
9 • Badges earned
Favorite Badges
Add a badge
Add a badge
Add a badge
Add a badge
Add a badge
All Badges
First Codelab Completed
Aug 31, 2020
First App in Android Studio
Sep 1, 2020
Introduction to Kotlin
Aug 31, 2020
Google Developer Profile Beta User
Aug 31, 2020
Dice Roller App
Sep 9, 2020
Build a Basic Layout
Sep 3, 2020
First Learning Pathway and Quiz badge
Aug 31, 2020
Created Google Developer Profile
Aug 31, 2020
Build your first Android app
Sep 17, 2020
Terms | Privacy

CONCLUSION

This Project is based on application development, in which we have learnt new technologies like kotlin and xml. We made an application named PDF Maker which converts images into PDF files. During the working period of this project we have drawn a conclusion that this project will be very useful for everyone. We have completed this project after facing so many errors and bugs. They gave us the strength to solve them efficiently and our mentor helped us a lot in making this project successful. We would like to thank our mentors and team mates for their continuous support throughout this journey.

REFERENCE

- Stack Overflow - Helpful in Error and Bug Fixing
- Kotlin course - Training Courses
- Wireframe tool Balsamiq - Wireframes to Go