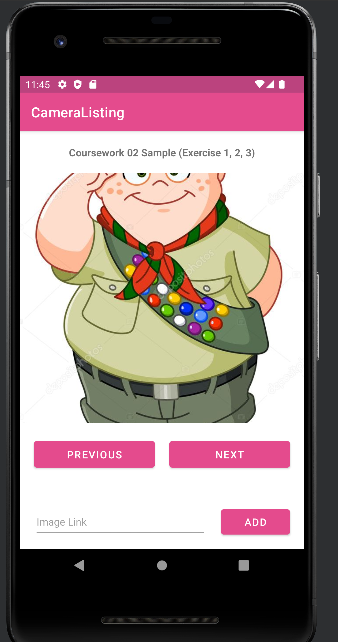
I/ Logbook 1- Previous function

|  |  |
| --- | --- |
| Student name | Phung Ngoc Tan |
| Who did you work with? Note that for  logbook exercises you are allowed to  work with one other person as long as  you give their name and login id and  both contribute to the work. | Name:  Login id: |
| Which Exercise is this? Tick as  appropriate. | ✓ |
| How well did you complete the  exercise? Tick as appropriate. |  I did it but I feel I should have done better   |
| Briefly explain your answer to question | I only able to implement the basic requirements, didn’t look into the layout or improvement which I can do better if I had more time |

1. Image



Picture 1 General Interface

This is how the interface works, we had the main center where the image will be add into, under it will be two buttons for two functions which is previous and next, the next button will move to the next image whethers the previous button will move backwards, the add button will give the user the ability to add image to a blank file

1. Activity.xml for previous function

<Button  
 android:id="@+id/btnPrevious"  
 android:layout\_width="0dp"  
 android:layout\_height="50dp"  
 android:layout\_marginStart="20dp"  
 android:layout\_marginTop="20dp"  
 android:layout\_marginEnd="10dp"  
 android:text="Previous"  
 android:textSize="15sp"  
 app:layout\_constraintEnd\_toStartOf="@+id/screenDividerHorizontal"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/imageView" />

This code demonstrate how the button was design, they given the id as well as the desinate width and height. I also adjust the margin from start,top,end. The text descibed what will be in the button which is previous. The layout lines will put the button to the wished place which is topbottom under the image view

1. How previos button works

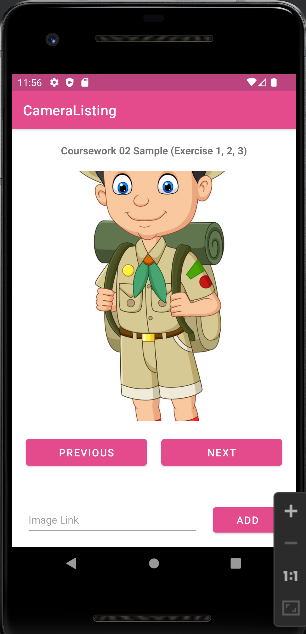
protected void previousImage() {  
 --\_currentIndex;  
 loadImage();  
}

The previous function will move the index backwards from the current index “ --currentindex “” thn it will load the image of that page

II/ Logbook-Next function

|  |  |
| --- | --- |
| Student name | Phung Ngoc Tan |
| Who did you work with? Note that for  logbook exercises you are allowed to  work with one other person as long as  you give their name and login id and  both contribute to the work. | Name:  Login id: |
| Which Exercise is this? Tick as  appropriate. | ✓ |
| How well did you complete the  exercise? Tick as appropriate. |  I did it but I feel I should have done better   |
| Briefly explain your answer to question | I only able to implement the basic requirements, didn’t look into the layout or improvement which I can do better if I had more time |

1. Image



This is how the interface works, we had the main center where the image will be add into, under it will be two buttons for two functions which is previous and next, the next button will move to the next image whethers the previous button will move backwards, the add button will give the user the ability to add image to a blank file. In here by clicking next we had move to another page and another image was loaded

1. Activity.xml of next function

<Button  
 android:id="@+id/btnNext"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:layout\_marginStart="10dp"  
 android:layout\_marginEnd="20dp"  
 android:text="Next"  
 android:textSize="15sp"  
 app:layout\_constraintBottom\_toBottomOf="@+id/btnPrevious"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toEndOf="@+id/screenDividerHorizontal"  
 app:layout\_constraintTop\_toTopOf="@+id/btnPrevious" />

This code demonstrate how the button was design, they given the id as well as the desinate width and height. I also adjust the margin from start,top,end. The text descibed what will be in the button which is previous. The layout lines will put the button to the wished place which is topbottom under the image view

1. How this button works

protected void nextImage() {  
 ++\_currentIndex;  
 loadImage();  
}

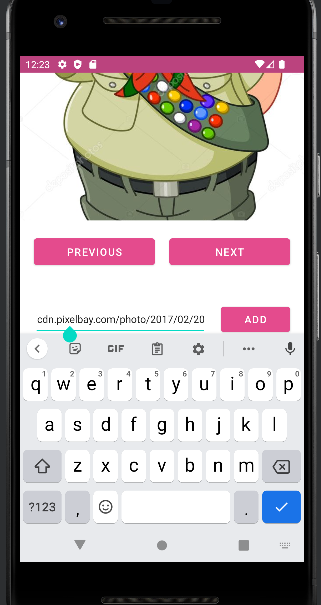
Not like the previous button the button will go forward instead from the currentindex ( “ currentindex++”), then it will load the image on the page.

From my interface , we can see that the imagewas changed due to the fact I click to the next button, it move to the new index and the image will load

III/ Logbook- Add function

1.Image

In order for use to add image to the appilication, this function is added



A provide image linked

1. Acticity.xml

<EditText  
 android:id="@+id/etImageLink"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:layout\_marginEnd="20dp"  
 android:ems="10"  
 android:hint="Image Link"  
 android:inputType="textPersonName"  
  
 android:textSize="15sp"  
 app:layout\_constraintBottom\_toBottomOf="@+id/btnAdd"  
 app:layout\_constraintEnd\_toStartOf="@+id/btnAdd"  
 app:layout\_constraintStart\_toStartOf="@+id/btnPrevious"  
 app:layout\_constraintTop\_toTopOf="@+id/btnAdd" />

This is how the image link bar was coded in layout

<Button  
 android:id="@+id/btnAdd"  
 android:layout\_width="100dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="15dp"  
 android:text="Add"  
 android:textSize="15sp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="@+id/btnNext" />

This is how the add button was implemented

1. How add function run

protected void addImage() {  
 String imageURL = etImageLink.getText().toString();  
  
 \_imageList.add(imageURL);  
 writeURLToFile(imageURL);  
  
 etImageLink.setText("");  
  
 Toast.*makeText*(this, "Added successfully.", Toast.*LENGTH\_SHORT*).show();  
}

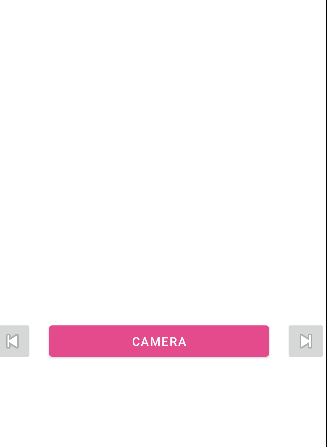
The image will add through URL and the pop up added successfully to notice the use that the image was added

protected void writeURLToFile(String url) {  
 try {  
 OutputStreamWriter outputStreamWriter = new OutputStreamWriter(openFileOutput(*\_FILE\_NAME*, *MODE\_APPEND*));  
 outputStreamWriter.write(url);  
 outputStreamWriter.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 Toast.*makeText*(this, "File not found.", Toast.*LENGTH\_SHORT*).show();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 private void getImageListFromFile(ArrayList<String> imageList) {  
 try {  
 InputStream inputStream = openFileInput(*\_FILE\_NAME*);  
  
 if (inputStream != null) {  
 InputStreamReader inputStreamReader = new InputStreamReader(inputStream);  
 BufferedReader bufferedReader = new BufferedReader(inputStreamReader);  
  
 String url = "";  
 while ((url = bufferedReader.readLine()) != null) {  
 imageList.add(url);  
 }  
  
 inputStream.close();  
 }  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 Toast.*makeText*(this, "File not found.", Toast.*LENGTH\_SHORT*).show();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

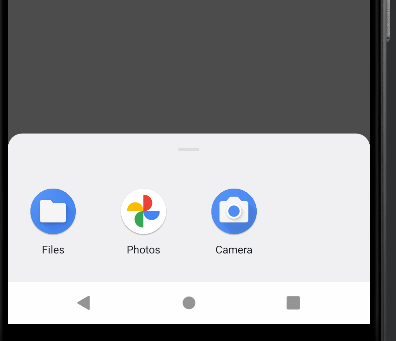
This code demonstrate how the function take the URL that was added and convert it to image which will display on the image view after being added

IV/ Logbook 4- Camera

|  |  |
| --- | --- |
| Student name | Phung Ngoc Tan |
| Who did you work with? Note that for  logbook exercises you are allowed to  work with one other person as long as  you give their name and login id and  both contribute to the work. | Name:  Login id: |
| Which Exercise is this? Tick as  appropriate. | ✓ |
| How well did you complete the  exercise? Tick as appropriate. |  I did it but I feel I should have done better   |
| Briefly explain your answer to question | I only able to implement the basic requirements, didn’t look into the layout or improvement which I can do better if I had more time |



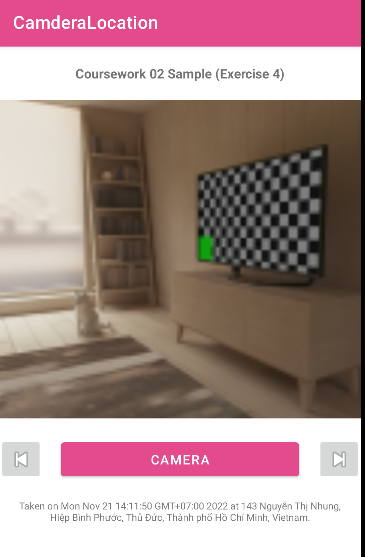
This is the front screen where there is a camera function and 2 button that can move forward and backward to view image



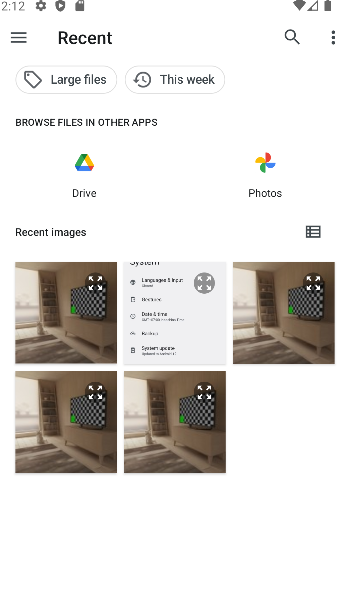
When clicking to the camera, it will take a shot



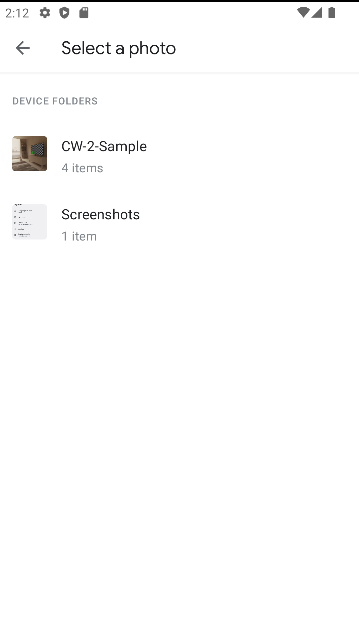
Click to the camera to take a picture



The picture has been taken and also show the location



Click to the file icon to see all the available file



Click to the photo icon to view all the photo

1. **Code and how it works**

protected void nextImage() {  
 ++\_currentIndex;  
 loadImage();  
 }  
  
 protected void previousImage() {  
 --\_currentIndex;  
 loadImage();  
 }  
}

This code is used for move forward and backward images

<Button  
 android:id="@+id/btnTakePicture"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:layout\_marginStart="20dp"  
 android:layout\_marginEnd="20dp"  
 android:text="Camera"  
 android:textSize="15sp"  
 app:layout\_constraintBottom\_toBottomOf="@+id/btnPrevious"  
 app:layout\_constraintEnd\_toStartOf="@+id/btnNext"  
 app:layout\_constraintStart\_toEndOf="@+id/btnPrevious"  
 app:layout\_constraintTop\_toTopOf="@+id/btnPrevious" />  
  
<ImageButton  
 android:id="@+id/btnPrevious"  
 android:layout\_width="50dp"  
 android:layout\_height="50dp"  
 android:layout\_marginTop="20dp"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/screenDividerVertical"  
 app:srcCompat="@android:drawable/ic\_media\_previous" />  
  
<ImageButton  
 android:id="@+id/btnNext"  
 android:layout\_width="50dp"  
 android:layout\_height="0dp"  
 app:layout\_constraintBottom\_toBottomOf="@+id/btnPrevious"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintTop\_toTopOf="@+id/btnPrevious"  
 app:srcCompat="@android:drawable/ic\_media\_next" />

These code dedicated to the layout of the 3 main buttons , the camera, the next and previous

protected void takePicture() {  
 // Ask for camera permissions.  
 if (!allPermissionsGranted\_CAMERA()) {  
 ActivityCompat.*requestPermissions*(this, *REQUIRED\_PERMISSIONS\_CAMERA*, *REQUEST\_CODE\_PERMISSIONS\_CAMERA*);  
 return;  
 }  
  
 Intent imageCaptureIntent = new Intent(MediaStore.*ACTION\_IMAGE\_CAPTURE*);  
  
 Intent fileIntent = new Intent(Intent.*ACTION\_GET\_CONTENT*);  
 fileIntent.setType("image/\*");  
  
 Intent galleryIntent = new Intent(Intent.*ACTION\_PICK*, MediaStore.Images.Media.*EXTERNAL\_CONTENT\_URI*);  
 galleryIntent.setType("image/\*");  
  
 Intent chooserIntent = Intent.*createChooser*(imageCaptureIntent, "Select Image");  
 chooserIntent.putExtra(Intent.*EXTRA\_INITIAL\_INTENTS*, new Intent[]{fileIntent, galleryIntent});  
  
 startActivityForResult(chooserIntent, *REQUEST\_CODE\_CAMERA*);  
}

This code is use when the camera take the picture

protected Uri saveImage(Bitmap bitmap) {  
 String fileName = new SimpleDateFormat("yyyyMMddHHmmss", Locale.*US*).format(new Date());  
 String path = Environment.*DIRECTORY\_PICTURES* + File.*separator* + "CW-2-Sample";  
  
 ContentValues contentValues = new ContentValues();  
 contentValues.put(MediaStore.MediaColumns.*RELATIVE\_PATH*, path);  
 contentValues.put(MediaStore.MediaColumns.*DISPLAY\_NAME*, fileName);  
 contentValues.put(MediaStore.MediaColumns.*MIME\_TYPE*, "image/png");  
  
 ContentResolver resolver = getContentResolver();  
 Uri imageUri = resolver.insert(MediaStore.Images.Media.*EXTERNAL\_CONTENT\_URI*, contentValues);  
  
 try (ParcelFileDescriptor parcelFileDescriptor = resolver.openFileDescriptor(imageUri, "w")) {  
 FileDescriptor fileDescriptor = parcelFileDescriptor.getFileDescriptor();  
  
 try (OutputStream stream = new FileOutputStream(fileDescriptor)) {  
 // Perform operations on "stream".  
 bitmap.compress(Bitmap.CompressFormat.*PNG*, 100, stream);  
 }  
  
 // Sync data with disk. It's mandatory to be able later to call writeExif.  
 fileDescriptor.sync();  
 } catch (IOException e) {  
 e.printStackTrace();  
 Toast.*makeText*(this, "Saving Image FAILED.", Toast.*LENGTH\_SHORT*).show();  
 return null;  
 }  
  
 return imageUri;  
}

This code is use for save image, after taking the image, this function will be executed and store the image

protected void startGPS() {  
 // Check permissions.  
 if (!allPermissionsGranted\_GPS()) {  
 ActivityCompat.*requestPermissions*(this, REQUIRED\_PERMISSIONS\_GPS, REQUEST\_CODE\_PERMISSIONS\_GPS);  
 return;  
 }  
  
 locationListener = location -> {  
 currentLocation = location;  
 Toast.*makeText*(MainActivity.this, "Get current location successfully.", Toast.*LENGTH\_SHORT*).show();  
 };  
  
 locationManager = (LocationManager) getSystemService(Context.*LOCATION\_SERVICE*);  
 locationManager.requestLocationUpdates(LocationManager.*GPS\_PROVIDER*, LOCATION\_REFRESH\_TIME, LOCATION\_REFRESH\_DISTANCE, locationListener);  
}  
  
protected void loadImage() {  
 if (\_imageList == null || \_imageList.size() == 0) {  
 Toast.*makeText*(this, "No Image !!!", Toast.*LENGTH\_SHORT*).show();  
 return;  
 }  
  
 Uri imageUri = \_imageList.get(\_currentIndex);  
  
 imageView.setImageURI(imageUri);  
  
 try {  
 if (android.os.Build.VERSION.*SDK\_INT* >= android.os.Build.VERSION\_CODES.*Q*) {  
 InputStream inputStream = getContentResolver().openInputStream(imageUri);  
 ExifInterface exifInterface = new ExifInterface(inputStream);  
  
 String tag\_address = "Not Found";  
  
 String tag\_date = exifInterface.getAttribute(ExifInterface.*TAG\_DATETIME\_ORIGINAL*);  
 String tag\_latitude = exifInterface.getAttribute(ExifInterface.*TAG\_GPS\_DEST\_LATITUDE\_REF*);  
 String tag\_longitude = exifInterface.getAttribute(ExifInterface.*TAG\_GPS\_DEST\_LONGITUDE\_REF*);  
  
 if (tag\_longitude != null && !tag\_longitude.isEmpty() &&  
 tag\_latitude != null && !tag\_latitude.isEmpty()) {  
 double latitude = Double.*parseDouble*(tag\_latitude);  
 double longitude = Double.*parseDouble*(tag\_longitude);  
  
 Geocoder gcd = new Geocoder(getBaseContext(), Locale.*getDefault*());  
 List<Address> addresses = gcd.getFromLocation(latitude, longitude, 1);  
  
 if (addresses.size() > 0) {  
 tag\_address = addresses.get(0).getAddressLine(0);  
 }  
 }  
  
 String description = "Taken on " + tag\_date + " at " + tag\_address + ".";  
 tvDescription.setText(description);  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
}

In order to get the GPS location for the image, this code will be executed

protected void writeExif(Uri uri) {  
 try (ParcelFileDescriptor imagePfd = getContentResolver().openFileDescriptor(uri, "rw")) {  
 if (android.os.Build.VERSION.*SDK\_INT* >= android.os.Build.VERSION\_CODES.*N*) {  
 ExifInterface exif = new ExifInterface(imagePfd.getFileDescriptor());  
  
 exif.setAttribute(ExifInterface.*TAG\_DATETIME\_ORIGINAL*, new Date().toString());  
 exif.setAttribute(ExifInterface.*TAG\_ARTIST*, "PNT");  
 exif.setAttribute(ExifInterface.*TAG\_IMAGE\_DESCRIPTION*, "This is from PNT' codes.");  
  
 if (null != currentLocation) {  
 exif.setAttribute(ExifInterface.*TAG\_GPS\_DEST\_LATITUDE\_REF*, String.*valueOf*(currentLocation.getLatitude()));  
 exif.setAttribute(ExifInterface.*TAG\_GPS\_DEST\_LONGITUDE\_REF*, String.*valueOf*(currentLocation.getLongitude()));  
 }  
  
 exif.saveAttributes();  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
}

Meanwhile this code will write the details GPS to the image