Ps. Underlined text means I need to remember it.

https://github.com/aegerita/Spot-A-Flower

Journal 1

March 16 Monday

Logo

At first I used Adobe Spark to design the logo. With various fonts and icons. So at first the logo looks like this with the name of the product and some leaves surrounding it.

However, after consideration, it would look really weird as an app icon. So I designed another one with symbols instead of words.

Instead of Adobe Illustrator, I actually used Vectormator on ipad pro. Because the Apple Pencil was very handy to do artwork, the logo was done in an hour. And I reused the colour scheme.

I used a lot of gradient filler to achieve a sense of value. I didn't realize it would cause problems. I stored the logo as a SVG file in order to use it as a vector asset in android studio. However, android studio does not support SVG files with gradient colours.

I tried changing it to monotone colours, but it's too ugly to work with. As a result, I just used it as an image asset and hope it wouldn't look weird on a phone. It is now a 512*512 png. BUT I can always go back to the original SVG file and save it to a bigger sized png.

GitHub

Created a github repository at https://github.com/aegerita/Spot-A-Flower. Created a new android project and linked it to the repository. It has been a while since I used git, so this was a great reminder to review my basic skills.

March 17 Tuesday

Logo

Uploaded the logo image to the project, set it as the app icon, and put it on an ImageView on the main page of the app. Looks amazing, I'm very proud of my work.

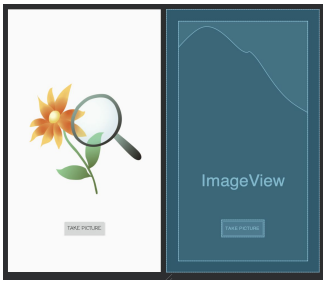
For the icon, I used a white background. No problem there.

For the main page, it was discussed that there should be a logo in the centre to fill the space. I added an ImageView, constrained to the edge of the phone and left save space.

Camera Button

There must be a button to take photos for image recognition.

I initially constrained it to the bottom of the logo Image. After testing it with different screen sizes, it does not work well (sometimes out of screen). As a result, I also linked it to the edge of the screen, but at 85%



downward. Although it sometimes covers the logo, who cares about it anyway.

It should have been an image button with a camera icon on it. But now it only has the word "take photo". If it looks ugly in the future, I'll draw a camera icon for it.

When the button is clicked, it must go to the default android activity for taking pictures, and pass the taken picture to the neural network. I used codes from this official tutorial. [Take photos]

```
// take photo when button is clicked
cameraButton.setOnClickListener { it: View!
    Intent(MediaStore.ACTION_IMAGE_CAPTURE).also { takePictureIntent ->
        takePictureIntent.resolveActivity(packageManager)?.also { it: ComponentName
            startActivityForResult(takePictureIntent, REQUEST_IMAGE_CAPTURE)
        }
    }
}
```

When the permission is not granted, the camera button will no longer take photos. It would simply remind the user to change the permission. I tried making it to check the permission status everytime it is clicked, but it would make a permission loop and crash the app. So I would later add to the user manual that they should restart the app if they change permission status.

```
// if the permission is not granted
override fun onRequestPermissionsResult(
    requestCode: Int,
    permissions: Array<String>,
    grantResults: IntArray
) {
    when (requestCode) {
        CAMERA_REQUEST -> {
            if ((grantResults.isEmpty() || grantResults[0] == PackageManager.PERMISSION_DENIED)) {
                //cameraButton.isEnabled = false
                cameraButton.setOnClickListener { it: View!
                    Toast.makeText(
                        context: this@MainActivity,
                        text: "Pls Allow Camera Access",
                        Toast.LENGTH_SHORT
                    ).show()
                }
            }
        }
    }
    return
}
```

Since the neural network is under construction, the taken photos are stored as imageBitmap, and replaces the logo simply to see if it worked. It did work despite the fact that I only wrote 10% of the codes.

```
// save the photo after it's taken
// for now just replace the logo, later pass it to the neural network
override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
    super.onActivityResult(requestCode, resultCode, data)
    if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == Activity.RESULT_OK) {
        val extras = data?.extras
        val imageBitmap = extras!!["data"] as Bitmap?
        logo.setImageBitmap(imageBitmap)
    }
}
```

Camera Permission

[Request App Permissions] Used this to ask for user permission. Modified the codes for my needs.

```
val CAMERA_REQUEST = 1
val REQUEST_IMAGE_CAPTURE = 1
```

At the beginning I didn't know what that meant. So I replaced it with 1 everywhere. It was a MISTAKE. Found the mistake after being confused for half an hour.

```
// ask for permission if haven't got one
if (ContextCompat.checkSelfPermission( context: this, Manifest.permission.CAMERA)
    != PackageManager.PERMISSION_GRANTED
) {
    ActivityCompat.requestPermissions(
        activity: this, arrayOf(Manifest.permission.CAMERA), CAMERA_REQUEST
    )
}
```

This code is copied from the tutorial. It works despite me confused by it. Maybe I should look at the kotlin library before coding.

After the break, I should ask the client whether he would like to save the photo after taking it with the app. If that function is requested, I could implement it rather easily. But not now since it's a bit boring to do.

Team

No response is given yet. Judging by the fact that now is still in March break, maybe I should take it easy and rest a bit. Later, I should start setting up the menu and results page and different activities. We have already discussed the design proof, so I would be fine to do it alone.

Journal 2

Mar 23, Monday

Toolbar

Trying to add the toolbar by using the support package, didn't work for some reason. Instead refractorred the whole project to use the AndroidX.

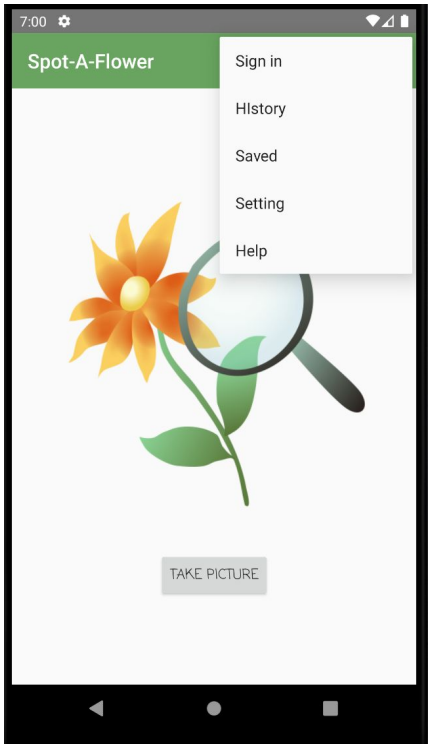
```
<androidx.appcompat.widget.Toolbar
    android:id="@+id/toolbar"
    android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
    android:background="#69A460"
    android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
    app:layout_constraintTop_toTopOf="parent"
    app:popupTheme="@style/Theme.AppCompat.Light" />
```

In the onCreate method of the main activity, I wrote setSupportActionBar(toolbar) so that the main activity could access the toolbar.

The app looks so much prettier, which is my main motivation to work on this project.

The colour should be stored as the prime colour of this project, luckily I remember.

Menu



Under the resources folder, I created the menu as a xml file and added it to the top bar, so that the user could access it whenever they want.

The menu is supposed to fulfill all the functions described in the project plan. However, it doesn't look nice.

So the next step would be fixing this menu to a sliding navigation window, which can be accessed by both clicking the top bar and swiping from left to right.

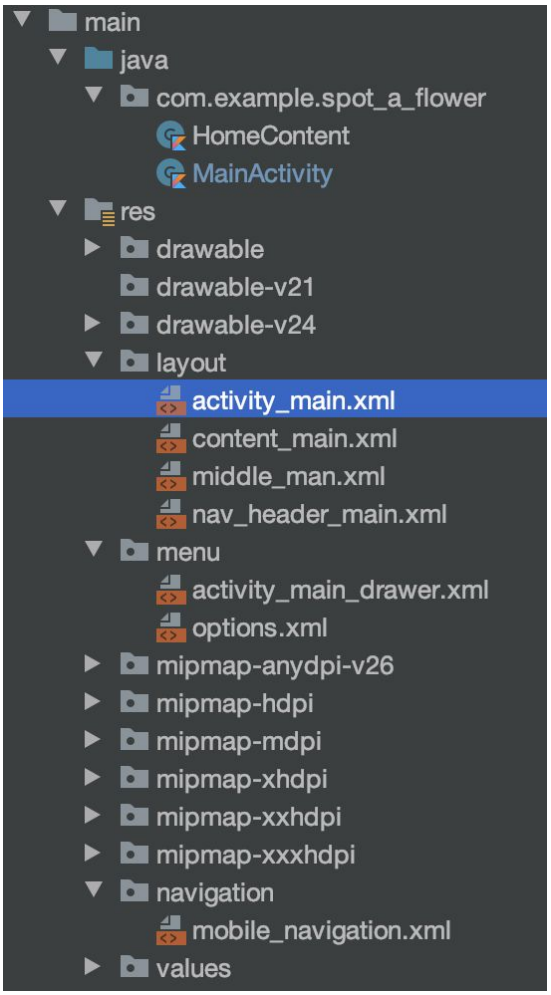
Menus Used this link to learn.

```
// add menu
override fun onCreateOptionsMenu(menu: Menu): Boolean {
    menuInflater.inflate(R.menu.options, menu)
    return true
}

// when menu items are clicked
override fun onOptionsItemSelected(item: MenuItem): Boolean {
    // Handle item selection
    return when (item.itemId) {
        // TODO
        R.id.saved -> {
            true
        }
        R.id.gallery -> {
            true
        }
        else -> super.onOptionsItemSelected(item)
    }
}
```

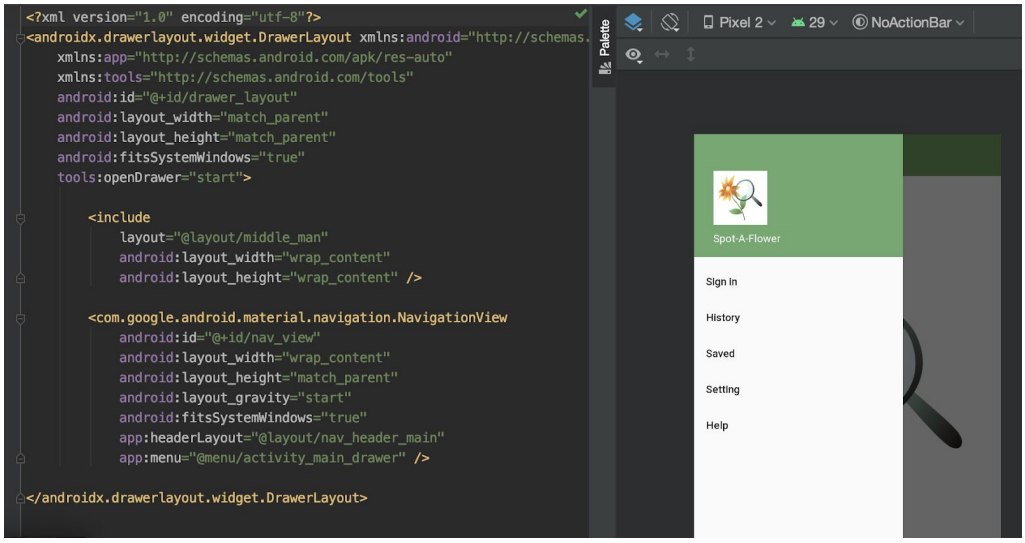
Navigation Slider (technically after mid night so Mar 24)

The navigation bar is a replacement for the menu above. Because it just looks better for the user to understand what is going on in this app. It was very difficult to understand at first. Navigation Although I have this link, it wasn't much help. Because the main activity was a constraint layout whole thing with a toolbar, camera button and everything in it, I had no idea how to get things out from there and include them in a whole new drawer layout.

The solution is, I made a new project with default drawer activity, and then I explored the file structure in that project. Then, I transferred a bunch of codes from there to my project. It surprisingly worked after a day of debugging.



This is what the file looks like currently. I only get a sense of how it works, but it works anyway. I'll try to explain it here.



This thing is now the new main activity. It has a navigation view with header and menu drawer (see the files, sorry) Rename it later since it is confusing which is which. And the original main page is included in the middle_man and content_main.

I originally put it all in content_main, but later when I tested it, I learned that the button and the toolbar can't be in the NavHostFragment to function, so I created a middle_man. I put the toolbar and cameraButton in there, while putting the logo to a navigation host fragment. Like this.

To summarize, activity_main includes middle_man and navigation; middle_man includes toolbar, camera button and this fragment. This fragment references the navGraph, which goes back to HomeContent class and content_main layout.

```
<fragment
    android:id="@+id/nav_host_fragment"
    android:name="androidx.navigation.fragment.NavHostFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:defaultNavHost="true"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:navGraph="@navigation/mobile_navigation"
    tools:layout="@layout/content_main" />
```

The class home content is put there like this.

```
class HomeContent : Fragment() {
    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? {
        return inflater.inflate(R.layout.content_main, container,  attachToRoot: false)
    }
}
```

I did it because that is what is done in the default drawer activity. So that the navigation can be accessed from the main page in the mobile_navigation.xml, like this.

```
<?xml version="1.0" encoding="utf-8"?>
<navigation 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/mobile_navigation"
    app:startDestination="@id/homeContent">
    <fragment
        android:id="@+id/homeContent"
        android:name="com.example.spot_a_flower.HomeContent"
        android:label="content_main"
        tools:layout="@layout/content_main" />
</navigation>
```

At last, the navigation must be added to the toolbar when the main activity is created.

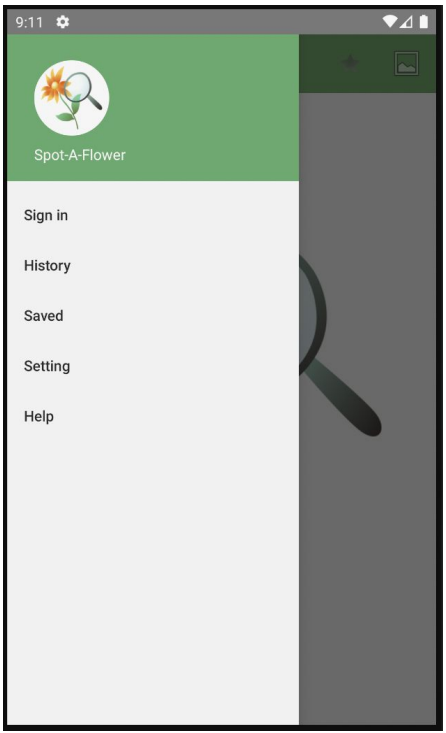
```
// add navigation to toolbar
val navController : NavController  = findNavController(R.id.nav_host_fragment)
val appBarConfiguration : AppBarConfiguration  = AppBarConfiguration(navController.graph, drawer_layout)
toolbar.setupWithNavController(navController, appBarConfiguration)
```

So there we are, the navigation is now working. It can be called both from the toolbar and swiping.

Mar 24, Tuesday

Menu

After yesterday the navigation started working, the menu needed to be changed to other functions.



Looks closer (sorry no saved picture), it is now 2 icons: favourite and gallery. I plan on changing the menu depending on user activity. When the user is on the main page, only the gallery appears to let the user choose a photo from storage. When the user searched for the flower, only favourites appeared to let the user save the flower.

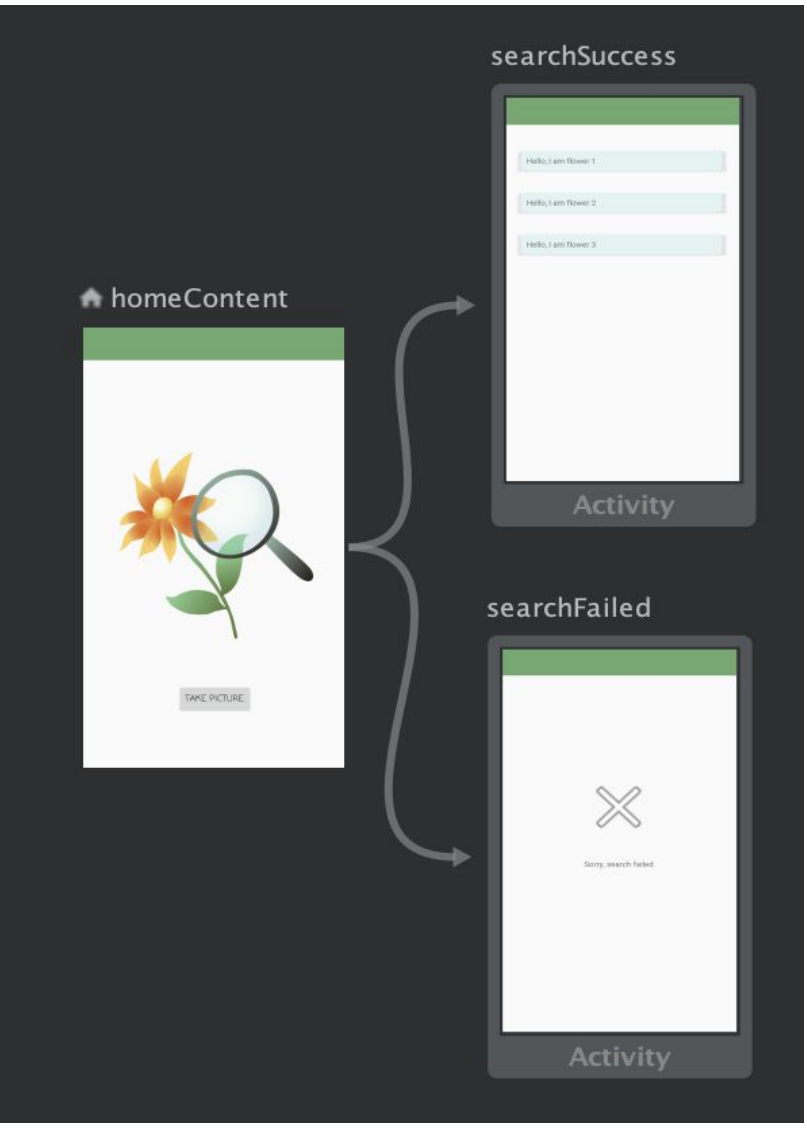
Anyway, now it is just two ugly icons. Draw the icon to a heart (clicked and not clicked) and a gallery because now it is too ugly.

And today I fixed a bunch of bugs and some style errors.

Mar 25, Wednesday

Call Search Results

Today I plan on adding two pages: one when the search is successful and one when the search is unsuccessful. There must be info passed to the success page so it can show the species of the flower.



It currently looks like this. When a photo is taken, the main activity calls the neural network (currently using math.random() to determine whether fail or succeed), then passes the result to searchSuccess page which shows the flower, or the searchFailed page.

This process is shown by the codes below.

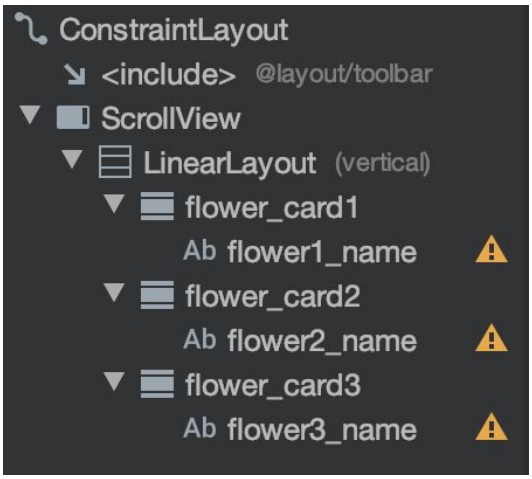
```
// call neural network to determine result
private fun searchFlower(imageBitmap: Bitmap?) {
    if (Math.random() < 0.5) {
        searchSuccess()
    } else
        searchFailed()
}

// go to failed page
private fun searchFailed() {
    val intent = Intent( packageContext: this, SearchFailed::class.java)
    startActivity(intent)
}

// go to success page
private fun searchSuccess() {
    val intent = Intent( packageContext: this, SearchSuccess::class.java)
    intent.putExtra( name: "flower1", value: "red flower")
    intent.putExtra( name: "flower2", value: "yellow flower")
    intent.putExtra( name: "flower3", value: "blue flower")
    startActivity(intent)
}
```

Toolbar

By the way, I took out the toolbar to a single toolbar layout xml file, so that different activities can use the same toolbar.

Search Success Page

This is the current layout page for search success. I plan on showing the flowers as cards after the search, which links to a details page for specific flowers. The neural network passes the flower name, and the activities take out information from the database to show in the card and detail page.



Anyway, the card view requires the layout be like this. Including the toolbar, the activity has a scroll view which shows the cards linearly.

Then, the page needs to be able to up to the parent view (home). As a result, the following code is written to go back to home. The same goes for the search fails page.

The menu is changed on this page. To search again, the user must go back to home, so the gallery ico is cancelled. This is just a test to see if it works, actually, all icons shouldn't show on this page. Gallery show in home page to allow user to choose photo from gallery (haven't made), Favorite show in details page to allow user to save flowers.

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

    // my_child_toolbar is defined in the layout file
    setSupportActionBar(findViewById(R.id.toolbar))

    // Get a support ActionBar corresponding to this toolbar and enable the Up button
    supportActionBar?.setDisplayHomeAsUpEnabled(true)
}

override fun onPrepareOptionsMenu(menu: Menu): Boolean {
    menuInflater.inflate(R.menu.options, menu)
    menu.findItem(R.id.gallery).isVisible = false
    return true
}
```

And I did this in the manifest.xml to link it to the parent home activity.

```
<activity
    android:name=".SearchSuccess"
    android:parentActivityName=".MainActivity" />

<activity
    android:name=".SearchFailed"
    android:parentActivityName=".MainActivity" />
```

Search Fail Page

Same process is done, except that this page has only a big cross sign and a text showing "Sorry, the search failed".

Plans

Later, I plan to work on the detail page, and the gallery function.

Mar 26, Thursday (after midnight)

Permission

Since I want to be able to read and write images in storage, I added these permissions to manifest.

```
        <uses-permission android:name="android.permission.CAMERA" />
+       <uses-permission android:name="android.permission.INTERNET" />
+       <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
+       <uses-permission android:name="ANDROID.PERMISSION.READ_EXTERNAL_STORAGE" />
```

```
// ask for camera permission if haven't got one
if (ContextCompat.checkSelfPermission(
        context: this, arrayOf(
            Manifest.permission.CAMERA,
            Manifest.permission.READ_EXTERNAL_STORAGE,
            Manifest.permission.WRITE_EXTERNAL_STORAGE
        ).toString()
    )
    != PackageManager.PERMISSION_GRANTED
) {
    ActivityCompat.requestPermissions(
        activity: this, arrayOf(
            Manifest.permission.CAMERA,
            Manifest.permission.READ_EXTERNAL_STORAGE,
            Manifest.permission.WRITE_EXTERNAL_STORAGE
        ), permissionRequest
    )
}
// if the permission is not granted
override fun onRequestPermissionsResult(
    requestCode: Int,
    permissions: Array<String>,
    grantResults: IntArray
) {
    when (requestCode) {
        permissionRequest -> {
            if (grantResults[0] == PackageManager.PERMISSION_DENIED) {
                cameraButton.setOnClickListener { it: View!
                    Toast.makeText(
                        context: this@MainActivity,
                        text: "Pls Allow Camera Access",
                        Toast.LENGTH_SHORT
                    ).show()
                }
            }
            if (grantResults[1] == PackageManager.PERMISSION_DENIED) {
                galleryPermitted = false
                invalidateOptionsMenu()
            }
```

Instead of checking permissions individually, I put them together like this.

So after the permission is checked, I can deal with the results one by one.

Menu

The camera button will only show alert when the camera is not granted. I designed the menu to do the same thing when the gallery is not granted, however, menu options can not change functions like that.

Because I am smart, I made a "fake" gallery menu option that shows up when the permission is not granted. The original options will set to be invisible, so the user will only see the button changes function, actually it is the button that changes.

When the permission is not granted, the method calls invalidateOptionMenu, which refreshes the menu. The menu then shows depending on the boolean variable galleryPermitted. Like the code below on the left.

Oops…

Holy cow, why didn't i set the function of the gallery option to change according to the boolean instead of refreshing the whole menu…

And I should have done the same thing with the camera…

Hh, it surprisingly didn't work with checking permission every time, I don't think it can check permission when the app is running. Strange. Turns out I have to make a boolean.

I changed the menu thingy like this, yet the camera thing remained like before.

Nice.

```
override fun onPrepareOptionsMenu(menu: Menu): Boolean {
    menu.clear()
    menuInflater.inflate(R.menu.options, menu)
    menu.findItem(R.id.gallery).isVisible = galleryPermitted
    menu.findItem(R.id.falseGallery).isVisible = !galleryPermitted
    menu.findItem(R.id.saved).isVisible = false
    return true
}
```

```
// Handle item selection
return when (item.itemId) {
    R.id.gallery -> {
        if (galleryPermitted){
            val intent = Intent(Intent.ACTION_PICK)
            intent.type = "image/*"
            startActivityForResult(intent, requestGalleryPhoto)
        } else {
            Toast.makeText(
                context: this@MainActivity,
                text: "Pls Allow Access to Photo Storage",
                Toast.LENGTH_SHORT
            ).show()
        }
        true
    }
```

Get Gallery Photo

```
} else if (requestCode == requestGalleryPhoto) {
    if (data != null) {
        val selectedImage: Uri? = data.data
        val imageBitmap : Bitmap! =
            MediaStore.Images.Media.getBitmap(this.contentResolver, selectedImage)
        //logo.setImageURI(selectedImage)
        searchFlower(imageBitmap)
    } else {
        Toast.makeText( context: this,  text: "Failed!", Toast.LENGTH_SHORT).show()
    }
}
```

Copied code from somewhere on stackexchange. Works well. I don't know whether the neural network needs bitmap or uri so I kept both.

Store Photo to Gallery

```
MediaStore.Images.Media.insertImage(
    contentResolver, imageBitmap,
    title: "Flower",  description: "from Spot-A-Flower"
)
```

When the camera activity is finished, the program should save the photo to the gallery. This code doesn't need permission, however, the photo is depreciated. I will change this inthe future.

Mar 27, Friday

Store Photo to Gallery

Today I spent 4 hours searching for a method to save a clear image to the gallery. Seeing the following contrast made my headache.

I've tried everything.

Everything.

It all looks like craps.

The following are some of my attempts, in addition to some which I deleted because nothing works.

Then out of nowhere, I tried putting the bitmap on to the logo imageView to see what it looks like. It also looks awful.

Then I realized that the problem is due to the camera passing in a bad quality bitmap, not saving it to the gallery.

```
fun getImageContent(parent: File): ContentValues? {
    val image = ContentValues()
    image.put(MediaStore.Images.Media.TITLE, "spot-a-flower")
    image.put(MediaStore.Images.Media.DISPLAY_NAME, "flower")
    image.put(MediaStore.Images.Media.DESCRIPTION, "App Image")
    image.put(MediaStore.Images.Media.DATE_ADDED, System.currentTimeMillis())
    image.put(MediaStore.Images.Media.MIME_TYPE, "image/jpg")
    image.put(MediaStore.Images.Media.ORIENTATION, 0)
    image.put(
        MediaStore.Images.ImageColumns.BUCKET_ID, parent.toString()
            .toLowerCase().hashCode()
    )
    image.put(
        MediaStore.Images.ImageColumns.BUCKET_DISPLAY_NAME, parent.name
            .toLowerCase()
    )
    image.put(MediaStore.Images.Media.SIZE, parent.length())
    image.put(MediaStore.Images.Media.DATA, parent.absolutePath)
    return image
}
```

```
private fun SaveImage(segg: Bitmap) {
    var fOut: OutputStream? = null
    val generator = Random()
    var n = 10000
    n = generator.nextInt(n)
    val fileName = "Image-$n.png"
    val appDirectoryName = "Spot-A-Flower"
    val imageRoot = File(
        Environment.getExternalStoragePublicDirectory(
            Environment.DIRECTORY_PICTURES
        ), appDirectoryName
    )
    imageRoot.mkdirs()
    val file = File(imageRoot, fileName)
    try {
        fOut = FileOutputStream(file)
    } catch (e: FileNotFoundException) {
        e.printStackTrace()
    }
    segg.compress(Bitmap.CompressFormat.PNG,  quality: 100, fOut)
    try {
        Toast.makeText(
             context: this,
            file.absolutePath,
            Toast.LENGTH_LONG
        ).show()
        fOut?.flush()
        fOut?.close()
    } catch (e: IOException) {
        e.printStackTrace()
    }
    val values = ContentValues()
    values.put(MediaStore.Images.Media.TITLE, "flower")
    values.put(MediaStore.Images.Media.DESCRIPTION, "from spot-a-flower")
    values.put(MediaStore.Images.Media.DATE_TAKEN, System.currentTimeMillis())
    values.put(
        MediaStore.Images.ImageColumns.BUCKET_ID,
```

```
val filename: File
try {
    val path1 = Environment.getExternalStorageDirectory()
        .toString()
    Log.i("in save()", "after mkdir")
    val file = File("$path1/flower")
    if (!file.exists()) file.mkdirs()
    filename = File(
        file.absolutePath.toString() + "/" + "flower"
                + ".jpg"
    )
    Log.i("in save()", "after file")
    val out = FileOutputStream(filename)
    Log.i("in save()", "after outputstream")
    imageBitmap.compress(Bitmap.CompressFormat.JPEG, 100, out)
    out.flush()
    out.close()
    Log.i("in save()", "after outputstream closed")
    //File parent = filename.getParentFile();
    val image: ContentValues = getImageContent(filename)
    val result = contentResolver.insert(
        MediaStore.Images.Media.EXTERNAL_CONTENT_URI, image
    )
    Toast.makeText(
        applicationContext,
        "File is Saved in  $filename", Toast.LENGTH_SHORT
    ).show()
} catch (e: Exception) {
    e.printStackTrace()
}*/
```

Anyway, It works like this: (all copied from [camera/photo basics](), should have looked into it) This method makes a place for storing the camera photo.

```
@Throws(IOException::class)
private fun createImageFile(): File {
    // Create an image file name
    val timeStamp: String = SimpleDateFormat( pattern: "yyyyMMdd_HHMMSS", Locale.CANADA).format(Date())
    val storageDir: File? = getExternalFilesDir(Environment.DIRECTORY_PICTURES)
    return File.createTempFile(
         prefix: "JPEG_${timeStamp}_", /* prefix */
         suffix: ".jpg", /* suffix */
        storageDir /* directory */
    ).apply { this: File
        // Save a file: path for use with ACTION_VIEW intents
        currentPhotoPath = absolutePath
    }
}
```

```
// take photo when button is clicked
cameraButton.setOnClickListener { it: View!
    Intent(MediaStore.ACTION_IMAGE_CAPTURE).also { takePictureIntent ->
        // Ensure that there's a camera activity to handle the intent
        takePictureIntent.resolveActivity(packageManager)?.also { it: ComponentName
            // Create the File where the photo should go
            val photoFile: File? = try {
                createImageFile()
            } catch (ex: IOException) {
                Toast.makeText( context: this,  text: "Can't Save Image", Toast.LENGTH_SHORT).show()
                null
            }
            // Continue only if the File was successfully created
            photoFile?.also { it: File
                val photoURI: Uri = FileProvider.getUriForFile(
                    context: this,
                    authority: "com.example.spot_a_flower.fileProvider",
                    it
                )
                takePictureIntent.putExtra(MediaStore.EXTRA_OUTPUT, photoURI)
                startActivityForResult(takePictureIntent, requestImageCapture)
            }
        }
    }
```

Then, when the camera is being called, it adds the camera intent to store the photo uri to the file. So the image is nice and clear.

Then when the camera is finished, it gets decoded again from the file to a simple bitmap, and then the bitmap is again stored to the Pictures folder.

Everything worked out at the end. But it's not like I could sleep before it works out as a workaholic, so…

Nice.

```
if (requestCode == requestImageCapture) {
    // get full size image
    val options = BitmapFactory.Options()
    options.inPreferredConfig = Bitmap.Config.ARGB_8888
    val imageBitmap : Bitmap!  = BitmapFactory.decodeFile(currentPhotoPath

    // save image to Pictures
    if (storagePermitted) {
        MediaStore.Images.Media.insertImage(
            contentResolver, imageBitmap,
            title: "Flower_" + SimpleDateFormat( pattern: "yyyyMMdd_HHMMSS",
            description: "from Spot-A-Flower"
        )
```

Permission

It seems that the reading and writing permission is asked together in the app, which means that I wouldn't need to deal with the situation separately when the user does not grant them.

So I made the saving picture thingy dependent on the same boolean with the reading storage one. (they are the same thing now)

If the user did not grant access to storage, there would just be a toast popping up. It is not an essential function, so I don't try to stop the user from taking pictures.

Journal 5

Mar 30, Monday

Overview

I thought again of my search result page, I found that it would be repetitive to show 3 different card views separately. I should have made a cardview model and added it 3 times to the search result activity.

Recyclerview According to this, I need to make a recycler view fragment to achieve that.

However, I thought about it again, I now have questions about the neural network. It is highly possible that it just gives out the most probable results instead of showing the possibility for each flower. So listing top 3 results may need extra work and stuff.

I could certainly directly go into the first flower details instead of a page listing top 3 results. It would be much easier.

However, if I implemented the results list, I could easily add a search function that let the user search flower by both photo and name. The display fragment would be the same if this is going to be implemented.

Since my teammate currently has no progress made in the break, this issue shall be discussed later. To be fair, the coronavirus might negatively affect our project even more, so maybe I should just cut all of the necessary functions.

No matter whether I am going to do the search results, the detailed flower information page must be done anyway. I will do that first, and then the page for each menu option.

Excuse

I got into Waterloo SE today!!! So I am too excited to work

Mar 31, Tuesday

Overview

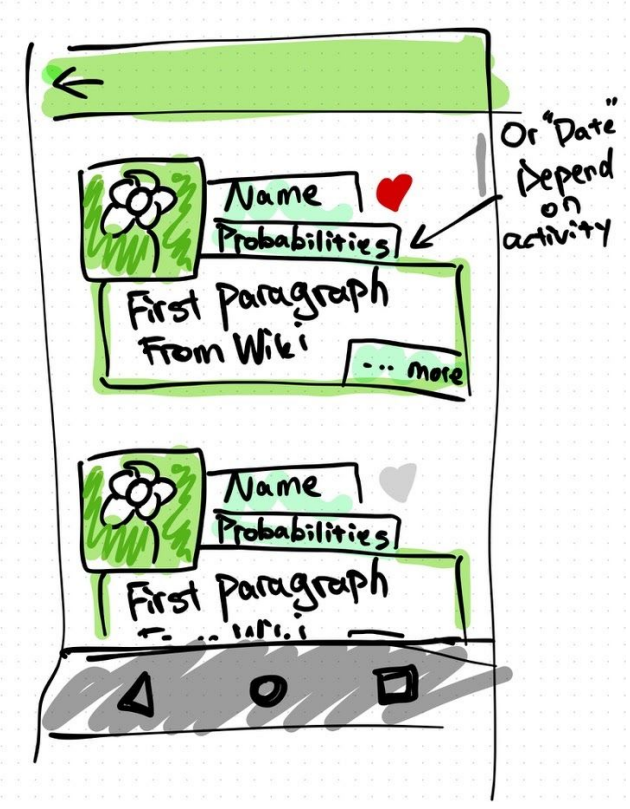
Reconsidering the thing I discussed yesterday. I missed a very important point. The history and the saved flowers page should also show the flower in a never ending card view style. So no matter what the search result page should look like, i need to do the recycler view anyway….

Feel like I have wasted a day.

CardView

Currently I am thinking of looking like this with photos and information laid out.

The heart button controls the "favorite" function, save it to the user database.



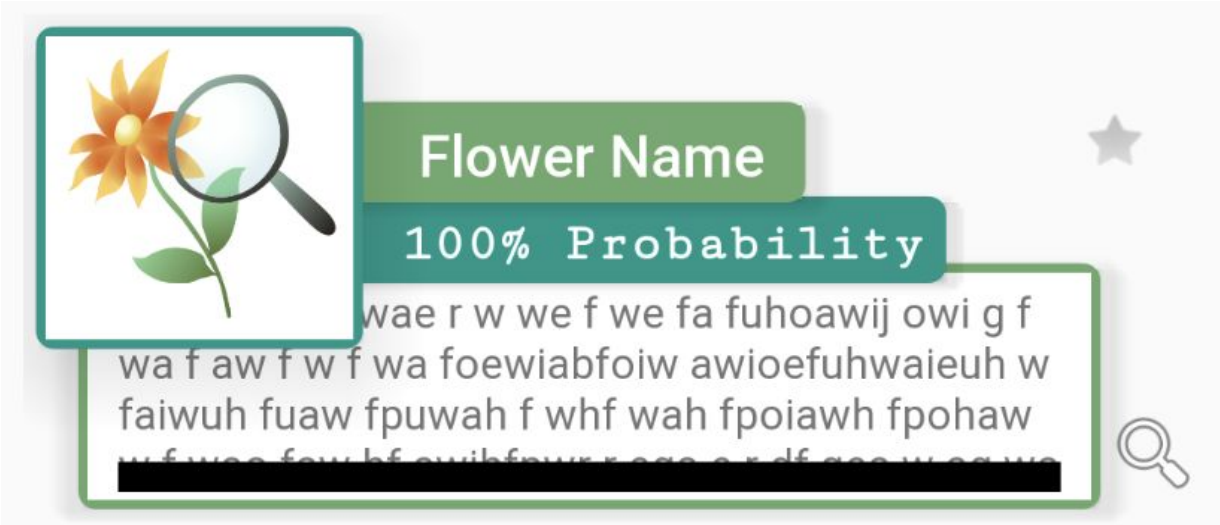
The "more" button will open the wikipedia link in the default android web browser (although i don't know how yet, hopefully as easy as the android default camera)

Mar 30 - 31, Wednesday - Thursday

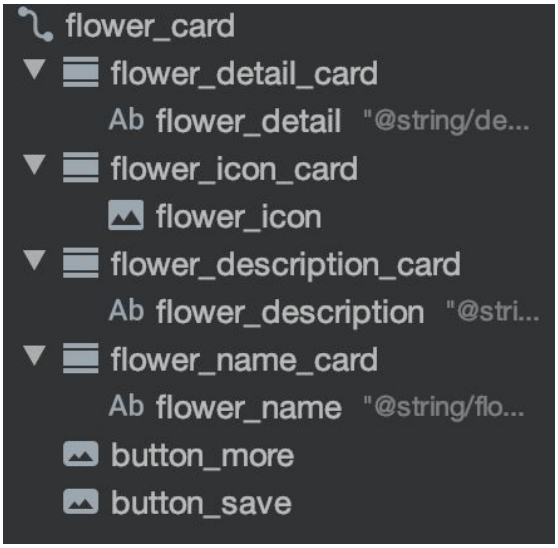
Recycler View

Since the android studio has a default fragment of recycler view I can add, I added it to the project. It has a fragment, a view adapter and a dummy object. I can't write something about something I don't understand, so I will get back to this section once I learned this on my own. But anyway, it can now list a bunch of the following flower cards on the search result page.

Flower Card



This is my current flower card. Anyone who sees this would understand my effort on making this beautiful. Adjusting the layout padding of each side of each element takes massive time. Everything matches up perfectly. But no one will know the padding is all "1.3dp", "2.7dp", "1.6dp", "18.5dp" with so much accuracy and arts… This took 2 days.



This is the structure of the flower card. There are 4 cards of flower information (icon, name detail, description) to create the floating effect with shadow. The 2 buttons are for saving the flower to the user database and to go to the detail page of the flower (or wikipedia depending on later discussion).

The actual thing actually looks like this because the description text view has set the ellipsize to "end" and the fadingEdge to 16dp. Thereby the user would know that the description is now ended, so they would press the button to see more.

The two buttons need to be drawn. The buttons now do not have a click listener which needs to be implemented in the recycler somehow. When the save is clicked, it needs to change the image to let the user know it is clicked.

Apr 1, Friday

Navigation Drawer

```
// set navigation drawer
val navigationView : NavigationView! = findViewById<NavigationView>(R.id.nav_view)
navigationView.setNavigationItemSelectedListener { menuItem ->
    when (menuItem.itemId) {
        R.id.account -> {
            drawer_layout.closeDrawer(GravityCompat.START)
            true ^setNavigationItemSelectedListener
        }
        R.id.history -> {
            drawer_layout.closeDrawer(GravityCompat.START)
            val intent = Intent( packageContext: this, FlowerSearch::class.java)
            intent.putExtra( name: "Parent",  value: "history")
            startActivity(intent)
            true ^setNavigationItemSelectedListener
        }
        R.id.saved -> {
            drawer_layout.closeDrawer(GravityCompat.START)
            val intent = Intent( packageContext: this, FlowerSearch::class.java)
            intent.putExtra( name: "Parent",  value: "saved")
            startActivity(intent)
            true ^setNavigationItemSelectedListener
        }
        R.id.setting -> {
            drawer_layout.closeDrawer(GravityCompat.START)
            true ^setNavigationItemSelectedListener
        }
        R.id.help -> {
            drawer_layout.closeDrawer(GravityCompat.START)
            true ^setNavigationItemSelectedListener
        }
        else -> {
            drawer_layout.closeDrawer(GravityCompat.START)
            false ^setNavigationItemSelectedListener
        }
    }
}
```

This is the code in the Main activity that sets the listener to each option in the navigation drawer.

When the history and saved option is chosen, the app will showcase the flower in the recycler view. So they start the search success activity, only that the flower dataset would come from the user database.

Flower Search

Now, whenever the flower search is called, it creates a dataset of flower class. Depending on what the intent message is coming from, the dataset can change. So that later when the user database is implemented, the flower dataset can be the corresponding set.

The flower class is currently:

```
// flower class
data class Flower(
    val name: String,
    val detail: String,
    val description: String
) {
    override fun toString(): String = "$name: $detail\ndescription"
}
```

The parameters match with the flower card, so the UI can show the text according to the flower.

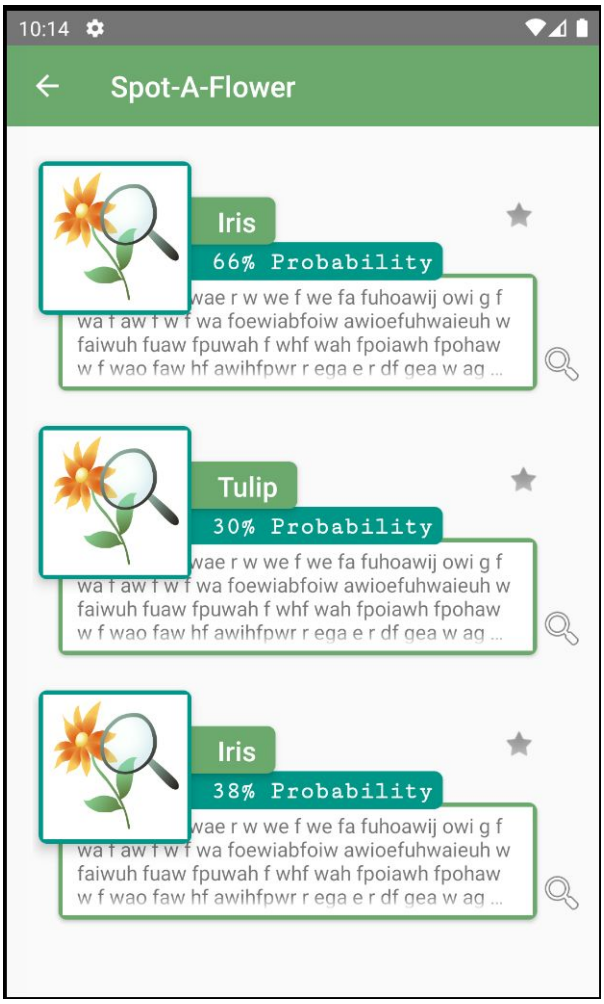
These flower objects in the dataset are passed into the recycler view.

```
// call the recycler view
viewManager = LinearLayoutManager( context: this)
viewAdapter = RecyclerViewAdapter(myDataset)

recyclerView = findViewById<RecyclerView>(R.id.flower_list).apply { this: RecyclerView!
    setHasFixedSize(true)
    // use a linear layout manager
    layoutManager = viewManager
    // specify an viewAdapter (see also next example)
    adapter = viewAdapter
}
```

Recycler View

```
class RecyclerViewAdapter(
    private val Flowers: MutableList<FlowerSearch.Flower>
) : RecyclerView.Adapter<RecyclerViewAdapter.ViewHolder>() {

    // Create new views (invoked by the layout manager)
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
        val view : View! = LayoutInflater.from(parent.context)
            .inflate(R.layout.fragment_flowers, parent, attachToRoot: false)
        return ViewHolder(view)
    }

    // Replace the contents of a view (invoked by the layout manager)
    override fun onBindViewHolder(holder: ViewHolder, position: Int) {
        // - get element from your data set at this position
        // - replace the contents of the view with that element
        val flower : FlowerSearch.Flower = Flowers[position]
        holder.name.text = flower.name
        holder.detail.text = flower.detail
        holder.description.text = flower.description
    }

    // Return the size of your data set (invoked by the layout manager)
    override fun getItemCount(): Int = Flowers.size

    // view holder, declare the UI component
    inner class ViewHolder(val flowerCard: View) : RecyclerView.ViewHolder(flowerCard) {
        val name: TextView = flowerCard.flower_name
        val detail: TextView = flowerCard.flower_detail
        val description: TextView = flowerCard.flower_description
        val icon: ImageView = flowerCard.flower_icon
    }
}
```

The adapter then is created. For each card, the UI component is stored in the view holder, then changed individually to the flower's info in the on bind view holder. So the UI can show a recycled view of the flower cards. Like this:



Name is generated randomly for now. Didn't expect two of "Iris" tho.

It can also be scrolled down.

Others

I also added fading animation to the buttons.

```
holder.moreButton.setOnClickListener { it: View!
    holder.moreButton.startAnimation(AlphaAnimation(1.0f, 0.2f))
    println("go to the supposed website")
}
```

Other than that, I tried to improve the project by referencing the main activity directly to the nav-host fragment, cutting out the middleman. I failed.

Journal 4

Apr 8, Wednesday

I am seriously thinking of abandoning this project.

I know I definitely do not have the ability to work as 2 people in an already pretty tight schedule. Impossible to catch up to all the testing, evaluations and such. Not like I don't want to, but I just can't work that much with better things to do.

https://github.com/aegerita/Spot-A-Flower

All my efforts and hard work is here, and in all my previous 3 journals.

Please allow me to treat this as a personal project, instead of school work.

At least let me work on my own term, so that I might still finish this in my own time.

Apr 12, Sunday

OMG! Jerry just applied saying he is still interested in doing the project!!

I am really glad my previous rant is useless. And I am glad he is still alive.

He said he has a lot of English stuff to catch up. Me too, in some sense.

BUT I am not worried about the future of the project anymore!

That being said, I didn't do a lot of work this week. However, I am confident my previous work can cover this week. I am way ahead of my schedule in the project plan.

Journal 5

Apr 15, Wednesday

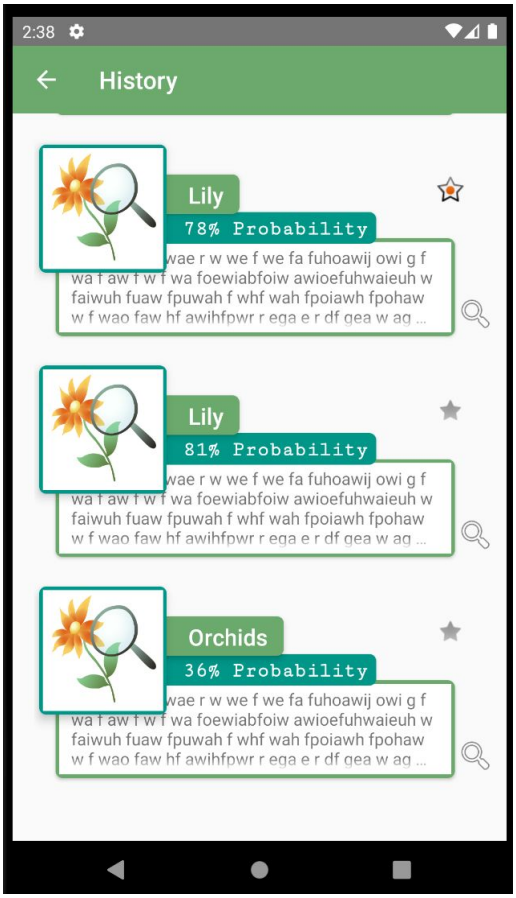
Scrollview layout

```
android:clipToPadding="false"
android:paddingTop="8dp"
android:paddingBottom="48dp"
android:scrollbarStyle="outsideOverlay"
```

Very simple codes in the xml file to padd the top and bottom of the scrollview.

Thereby there would be space before and after the flower list.

I designed it to show exactly three flowers for a Pixel 2 phone. For it to look pretty!



Toolbar

Like the picture on the left. The toolbar now changes title according to the name of the activity.

```
supportActionBar?.title = "History"
```

Using the code on top.

Open Webview

As other information about the flower, the link to the wikipedia is added as a parameter.

```
val link = "https://en.wikipedia.org/wiki/${name}"
```

The string is passed into the flower object.

```
// open link in browser
holder.moreButton.setOnClickListener {  it: View!
    holder.moreButton.startAnimation(AlphaAnimation(1.0f, 0.2f))
    val i = Intent(Intent.ACTION_VIEW, Uri.parse(flower.link))
    context.startActivity(i)
    //TODO
    println("save this to history")
}
```

When the button is clicked, it quickly does an animation, and then does the intent thing so that the android phone goes open the link in the default browser.

When the link is clicked, it is a signal to save this particular flower to history (user database). So that the user can access it in the history page. It will be done after the user database is established.

Save Button

```
// change star and save to database
holder.saveButton.setOnClickListener { it: View!
    holder.saveButton.startAnimation(AlphaAnimation(1.0f, 0.2f))
    if (holder.saveButton.tag == 1) {
        holder.saveButton.tag = 0
        holder.saveButton.setImageResource(android.R.drawable.star_off)
        //TODO
        println("cancel storing " + flower.name)
    } else {
        holder.saveButton.tag = 1
        holder.saveButton.setImageResource(android.R.drawable.star_on)
        //TODO
        println("store " + flower.name)
    }
}
```

I learned from stackoverflow that the button can have a tag. Depending on the current tag (status) of the button, different action is being made. It is whether storing this flower to a user database or unstoring. And the image needs to be changed when the button is clicked to change.

Of course, before clicking the button, it needs to initialize with the corresponding tag, according to the user database.

```
// initialize save button
if (flower.isSaved) {
    holder.saveButton.tag = 1
    holder.saveButton.setImageResource(android.R.drawable.star_on)
} else holder.saveButton.tag = 0
```

All the things about the search results page are set up. Now I can put my focus on the setting page and the account system.

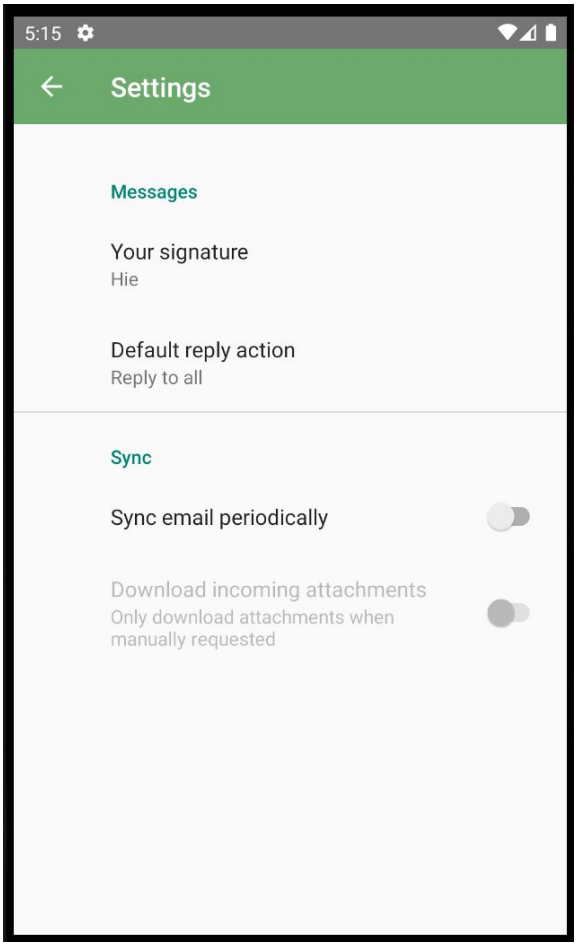
Apr 16, Thursday

Click flower

One last thing about the search result, because I found that the search button is super redundant. Why not just open the link when the whole flower card is clicked? That would be way more convenient!

So I deleted the moreButton and placed the click listener to the item view instead.

Settings



This is the current setting. I added the default setting activity to my project, and adjusted the layout a bit.

It added an activity class, a layout xml file, and a root_preference xml file.

```
<androidx.constraintlayout.widget.ConstraintLayout xmlns:an
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical">

    <include
        layout="@layout/toolbar"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <FrameLayout
        android:id="@+id/settings"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        android:layout_marginTop="56dp"
        android:paddingTop="16dp" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

The layout is just a normal layout file. I added the toolbar for my project. Obviously.

In the activity, the program has the following codes:

```
supportFragmentManager
    .beginTransaction()
    .replace(R.id.settings, SettingsFragment())
    .commit()
```

By my understanding, the class fills the setting layout with the following root_preference xml file.

It is the place that edits the menu options, like the top right phone.

```
class SettingsFragment : PreferenceFragmentCompat() {
    override fun onCreatePreferences(savedInstanceState: Bundle?, rootKey: String?) {
        setPreferencesFromResource(R.xml.root_preferences, rootKey)
    }
}
```

It is actually really convenient. I thought I had to make each option individually. Never mind that.

https://developer.android.com/guide/topics/ui/settings

This link has all the information I need to create a successful menu. I will study that and use it in my project.

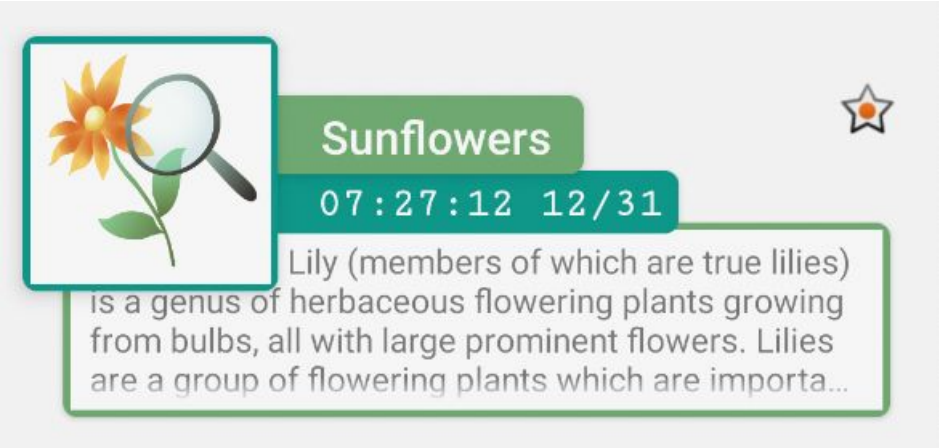
Flower Details

```
val detail : String! = if (scenario == "history" || scenario == "saved") {
    val date = Date((Random().nextDouble() * 60 * 60 * 24 * 365).toLong()
    val sdf = SimpleDateFormat( pattern: "hh:mm:ss MM/dd", Locale.CANADA)
    sdf.format(date)
} else {
    (Math.random() * 100).toInt().toString() + "% Probability"
}
```

I know i said no more search result works, but here we go. The flower detail now changes depending on activity. When it is in history or saved, it shows the timestamp (currently random, need database). When it is in the search result, it still shows probability.

```
val description : String = "                              " + getString(R.string.description)
val link = "https://en.wikipedia.org/wiki/${name}"
val isSaved : Boolean = if (scenario == "saved") true else Math.random() > 0.5
```

The description needs to jump through the space of the flower icon. So I added a bunch of space to push the word "Lily" outside of the picture. BTW I changed the default description about Lily to make it more enjoyable for me to program.



The wikipedia link is easy. Just put the name of the flower after the wikipedia default link.

And now all of the flowers in the saved page are saved by default.

I know I will change some of these after the user database is set up. However, most codes can be reused by that time.

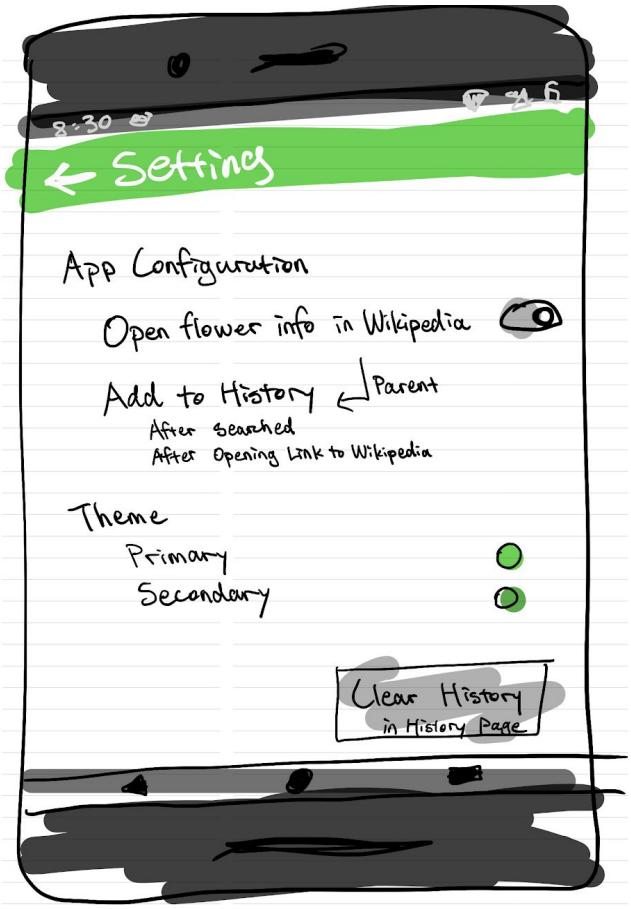
Apr 17, Friday

Settings Menu

The menu is not as big a workload as I thought it would be.

https://incipia.co/post/app-development/what-should-your-app-include-in-settings/

From this website, I checked the menu options of other application, and sketched my ideas:



My software doesn't have a lot of customizable configuration. The only thing I can currently think of is the open link thing in the search result page, because I just did it.

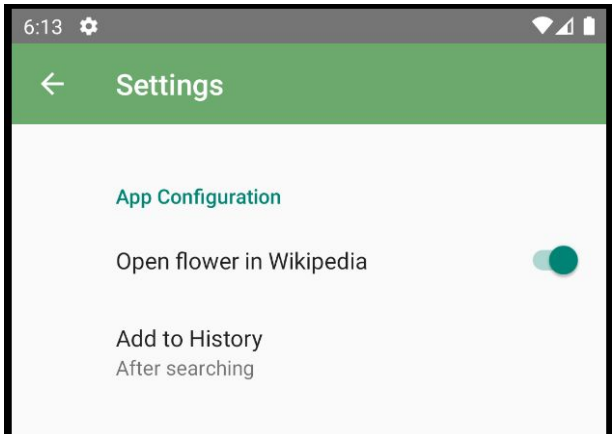
The user can choose whether the app opens the link when the flower is clicked.

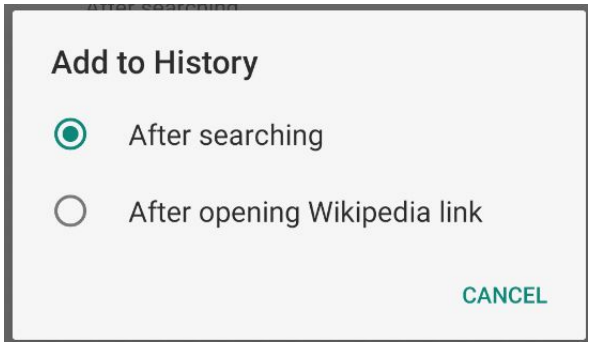
If the user allows it, then the user can choose when the flower is stored in the history database. Whether it is when the flower shows up in the search result, or only when the user opens the link.

The default is after searching.

So the first preference is a parent of the second preference.

```
<SwitchPreferenceCompat
    app:defaultValue="true"
    app:key="openWiki"
    app:title="Open flower in Wikipedia" />

<ListPreference
    app:defaultValue="true"
    app:dependency="openWiki"
    app:entries="@array/history_setting_entries"
    app:entryValues="@array/history_setting_values"
    app:key="addHistoryWhen"
    app:title="Add to History"
    app:useSimpleSummaryProvider="true" />
```





Now the codes are exactly like I said. Perfect!

BTW, the list preference shows this dialog right here to allow the user to choose the options.

Besides the app configuration, I also want to make a theme colour picker in the setting. I should try that if I want to do it.

Apply Setting Preference

Now that the options are set up, I need to implement the app to make the changes.

Learned from stackoverflow that the preference is stored by pairs according to the key I wrote.

```
val sharedPreferences = PreferenceManager.getDefaultSharedPreferences(this)
```

Using this code, and a bunch of if statements, I did the following changes.

If the user chooses to open the link, add the click listener for the flower item. Then if the user chooses to save the flower when opening the link, store the flower in the click listener.

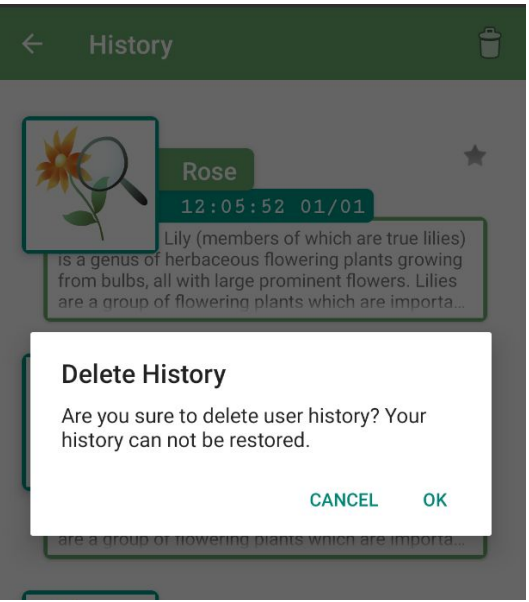
```
if (sharedPreferences.getBoolean( key: "openWiki",  defValue: true)) {
    holder.itemView.setOnClickListener { it: View!
        // when the flower is clicked, open link in browser
        context.startActivity(Intent(Intent.ACTION_VIEW, Uri.parse(flower.link)))
        // if the user choose to save to history when open link
        if (sharedPreferences.getString( key: "addHistoryWhen",  defValue: "search") == "link")
            // TODO save the flower to history when open wiki link
            println("save ${flower.name} to history")
        }
    }
}
```

If the user chooses to save the flower when searching, store the flower in the search result page before passing to the adapter, and make sure it is from search result, not from save or history.

```
val sharedPreferences : SharedPreferences! = PreferenceManager.getDefaultSharedPreferences( c
if ((scenario != "history" && scenario != "saved")
    && (sharedPreferences.getString( key: "addHistoryWhen",  defValue: "search") == "search"
            || sharedPreferences.getBoolean( key: "openWiki",  defValue: false))
) {
    // TODO save the flower to history when search
    println("save all of these flowers to history")
}
```

Delete History

I thought of adding a deleting history button in the setting. However, it is nicer to have it directly in the history page. So I added an option in the toolbar of the history page, and then added this code.



The alert dialog looks pretty nice. It is really

```
// make a alert dialog when user want to delete history
override fun onOptionsItemSelected(item: MenuItem): Boolean {
    return when (item.itemId) {
        R.id.delete_history -> {
            AlertDialog.Builder( context: this)
                .setTitle("Delete History")
                .setMessage(
                    "Are you sure to delete user history? " +
                            "Your history can not be restored. "
                )
                // when user confirms, delete history
                .setPositiveButton(
                    android.R.string.yes
                ) { _, _ ->
                    //TODO delete user history in database
                    println("delete user history in database")
                    // change the viewAdapter accordingly
                    myDataset.clear()
                    viewAdapter.notifyDataSetChanged()
                }
                .setNegativeButton(android.R.string.no,  listener: null)
                //.setIcon(android.R.drawable.ic_dialog_alert)
                .show()
            true
```

similar to the JOptionPane in java, which I still remember somehow.

But I need to remember that, whenever I change the dataset, I need to both notify the online database, and notify the viewAdapter to refresh.

After the database is done, I should add a "scroll down to refresh" function.

Navigation

After doing that successfully, when I tested the application on the emulator, I found a bug. The up button wouldn't let me return to the main page from the setting page. After a while I found that I forgot to add the activity to the manifest…

```
<activity
    android:name=".SettingsActivity"
    android:parentActivityName=".MainActivity" />
```

I should check my previous journal before implementing similar things.

But through my efforts to find solution to this bug, i came across to another method of adding the navigation drawer to the toolbar: actionBarDrawerToggle

This means a lot. This means I can finally cut out the middle man!!

Cut Out the Middleman

I moved everything from the main_content and the middle_man to the main activity. Now it is a drawer layout including the navigation bar and a constraint layout with toolbar camera button and the logo in it.

```
// add navigation to toolbar by toggle
val actionBarDrawerToggle =
    ActionBarDrawerToggle(
        activity: this,
        findViewById(R.id.drawer_layout),
        toolbar,
        R.string.open_drawer,
        R.string.close_drawer
    )
findViewById<DrawerLayout>(R.id.drawer_layout).addDrawerListener(actionBarDrawerToggle)
actionBarDrawerToggle.syncState()
```

And then I added this to the main activity, which replaced this:

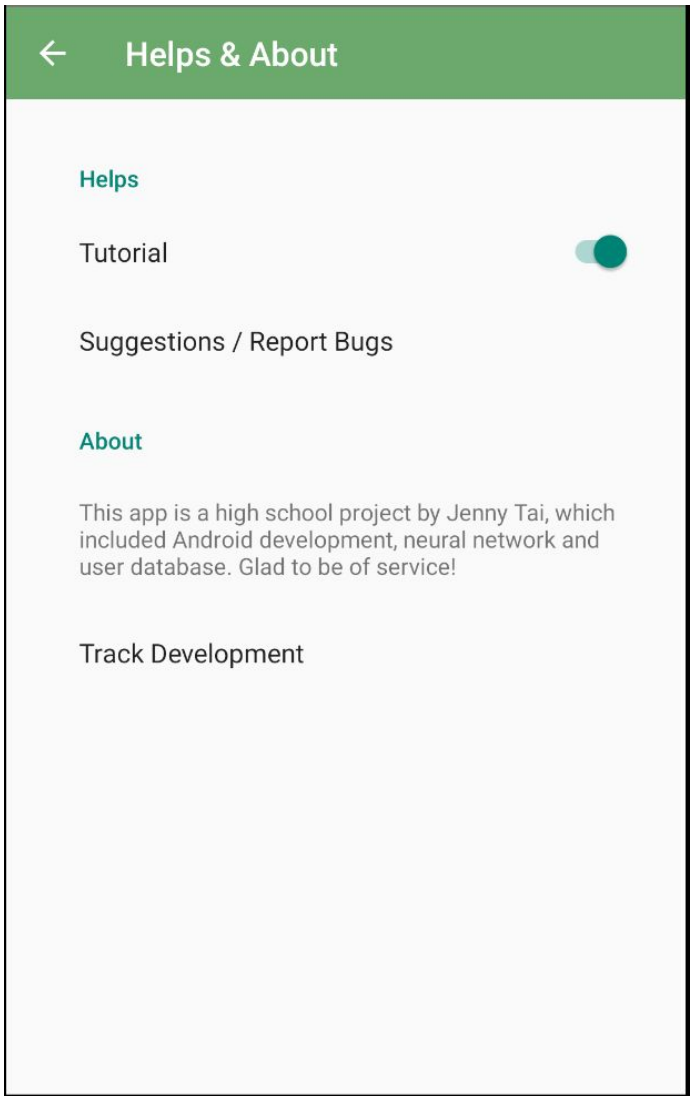
```
// add navigation to toolbar
//val navController = findNavController(R.id.nav_host_fragment)
//val appBarConfiguration = AppBarConfiguration(navController.graph, drawer_layout)
//toolbar.setupWithNavController(navController, appBarConfiguration)
```

I never thought it would be so easy! Now I can delete all those annoying extra classes and the navigation map. I am sure it has its purpose, but I don't understand and I don't need it.

So I get rid of it! This is a great process, right here!

Apr 18, Saturday

Helps and About Page



Because the setting preference is so convenient, I decided that I would make the help page by the same method.

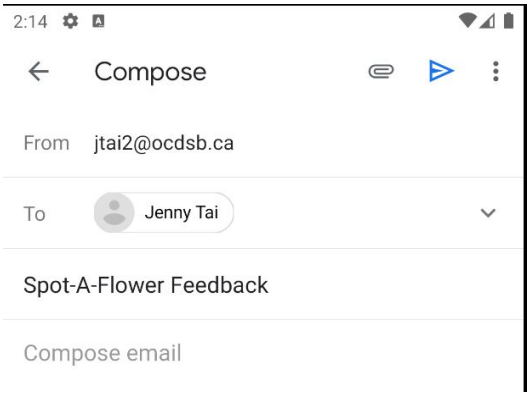
By the same method as the search page (passing in intent string extra to identify the parent activity), I made the setting class able to present the setting page and the help page.

```
val preferenceFrag: PreferenceFragmentCompat =
    if (intent.getStringExtra( name: "Parent") == "setting")
        SettingsFragment()
    } else {
        HelpsFragment()
    }
supportFragmentManager.beginTransaction().replace(R.id.setti
```

```
<PreferenceCategory
    app:iconSpaceReserved="false"
    app:title="Helps">
    <SwitchPreferenceCompat
        app:defaultValue="true"
        app:key="tutorial"
        app:iconSpaceReserved="false"
        app:title="Tutorial" />

    <Preference
        app:iconSpaceReserved="false"
        app:title="Suggestions / Report Bugs">
        <intent
            android:action="android.intent.action.VIEW"
            android:data="mailto:jennytai3221@gmail.com">
            <!--suppress AndroidElementNotAllowed -->
            <extra
                android:name="android.intent.extra.SUBJECT"
                android:value="Spot-A-Flower Feedback" />
        </intent>
    </Preference>

</PreferenceCategory>
```

The code for the help preference is on the right. The tutorial is now a switch button.

Currently the plan is that it opens the tutorial mode, and when the user finishes the tutorial, the button automatically switches back, waiting for the next click.



However, the tutorial could also be an animated view or something else. In that case this would include an intent to call that activity. Still deciding what is the tutorial.

The suggestion and bug report automatically write me an email in the default android email app. Like the left in Gmail.

This function is achieved by the android:action thing. Grateful it is so simple.

Although it now sends messages to my personal email, I believe one day there will be a point to create an email for this app!

```
<PreferenceCategory
    app:allowDividerAbove="false"
    app:iconSpaceReserved="false"
    app:title="About">
    <Preference
        app:iconSpaceReserved="false"
        app:selectable="false"
        app:summary="This app is a high school project by Jenny Tai, wh

    <Preference
        app:iconSpaceReserved="false"
        app:title="Track Development">
        <intent
            android:action="android.intent.action.VIEW"
            android:data="https://github.com/aegerita/Spot-A-Flower" />
    </Preference>

</PreferenceCategory>
```

For the about page, I just put string information about the app.

And the "track development" thing. When it is clicked, it opens the Spot-A-Flower github in the default browser.

Nice little addition.

Setting Layout

You can see there is an app:iconSpaceReserved code in every preference. It is for reducing the padding space in the left for every preference.

I can not tolerate ugly layout in my software!!! So I added this to every single preference and then changed the padding space of the fragment. You can see the layout today is a big improvement compared to yesterdays.

Search Result Page Reconstruction

So, after the setting is done, the only thing remaining for the UI is the account system. So I thought I would go through all my codes and comment on the part I didn't comment on.

But then I found a better way for the search result page. I used to make it my MainActivity's job to decide whether the search fails or not. It should not be the case because the search should happen in the search result activity. So I have to move it.

```
// replaced the neural network with random number generator, for now
val constant: Int = if (Math.random() < 0.5) {
    8
} else
    0
```

For now this constant decides whether the dataset has 8 flowers or 0 flowers. The search activity would then decide whether the search failed.

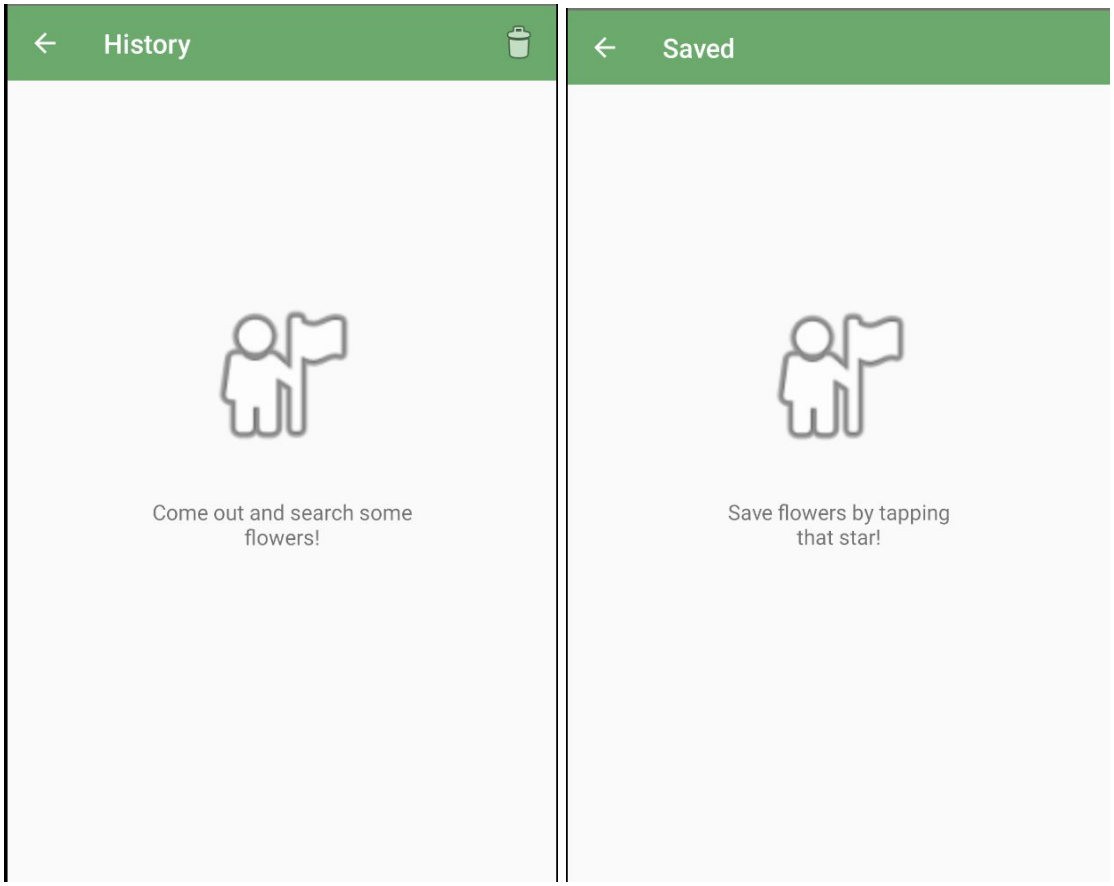
Look at the codes below: when there are flowers, it goes on as normal to the list of flowers. Otherwise, the code goes to a fail page, where messages are displayed according to scenario: search, history or saved.

This is fairly straightforward, because I used the exact same layout for the failed page, only changing the message.

BTW all the intent extra string is changed to the toolbar title. Because it is simpler.

```
// set up user interface
if (myDataset.size != 0) {
    // if dataset not empty, all use search success
    setContentView(R.layout.activity_search_success)
    viewManager = LinearLayoutManager( context: this)
    viewAdapter = RecyclerViewAdapter( context: this, myDataset)

    recyclerView = findViewById<RecyclerView>(R.id.flower_list).apply {
        setHasFixedSize(true)
        layoutManager = viewManager
        adapter = viewAdapter
    }
} else {
    // if dataset empty, all goes to fail page
    setContentView(R.layout.activity_search_failed)
    findViewById<TextView>(R.id.failText).text =
        when (scenario) {
            "History" -> {
                findViewById<ImageView>(R.id.failImage).setImageResourc
                "Come out and search some flowers!"
            }
            "Saved" -> {
                findViewById<ImageView>(R.id.failImage).setImageResourc
                "Save flowers by tapping that star!"
            }
            else -> "Sorry, search failed"
        }
}
// set up toolbar
setSupportActionBar(findViewById(R.id.toolbar))
supportActionBar?.setDisplayHomeAsUpEnabled(true)
supportActionBar?.title = scenario
```

So, it looks like the screen below:



Others look the same.

Why didn't I think of this when I was doing the paper prototype? Such a convenient and thoughtful UI to do when there is no saved flowers!

Delete History

Instead of notifying the change of the dataset to the recycler, the history should go straight to this failed page when deleted.

```
//TODO delete user history in database
println("delete user history in database")
// go back to failing page
setContentView(R.layout.activity_search_failed)
findViewById<ImageView>(R.id.failImage).setImageResou
findViewById<TextView>(R.id.failText).text =
    "Come out and search some flowers!"
// set up toolbar
setSupportActionBar(findViewById(R.id.toolbar))
supportActionBar?.setDisplayHomeAsUpEnabled(true)
supportActionBar?.title = "History"
```

So I wrote this in the original click listener. Basically redoing the top codes. The toolbar needs to be set again because the content view was changed.

Code Cleanup

After that, I returned to the initial task: cleaning and commenting the codes. The amount of redundant codes surprised me!

According to Github, I had 33 additions and 139 deletions in this

commit. And another 40 something in the previous one. It was a long process, but there is nothing to talk about because it is just commenting and moving things around.

Future

Anyway, the UI is all finished beside the user account thingy. I left it last because it is the least interesting thing for me… I am sure the paper (not actually paper) prototype can help me very much.

There is a thing I haven't figured out what to do. I need to change the UI a bit for anonymous and signed in versions. In java I just make a boolean static. I am not sure what to do here.

Apr 19, Sunday

Dark Theme Support

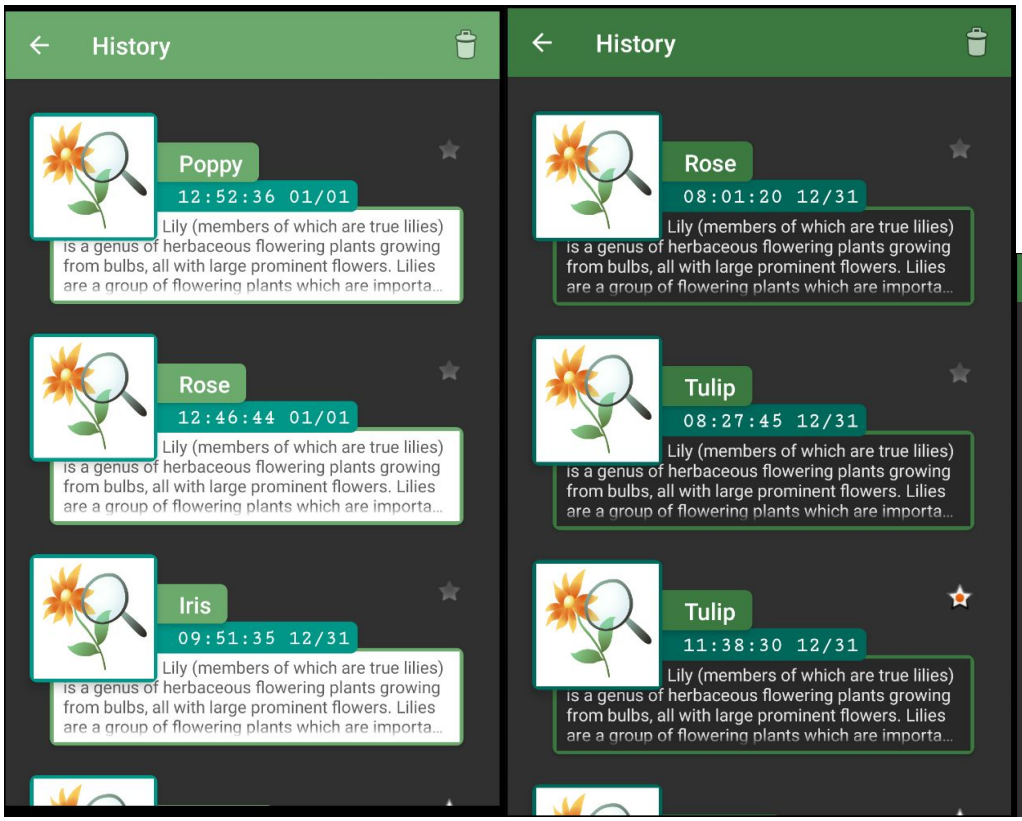
On my way to explore how to do the colour picker, I figured that it requires using a library and stuff like that. It is likely that I need to choose the colour for each component in programming manually. Because it cannot change the colour resource file, so… Not likely to be done.

So I gave up that idea. Instead, I can make my app to support the night mode, and make the user choose that in the setting. It would also be a nice idea.

All I did is change the theme in the manifest from "day" to "day night". So the default app colour would change with the android night mode.

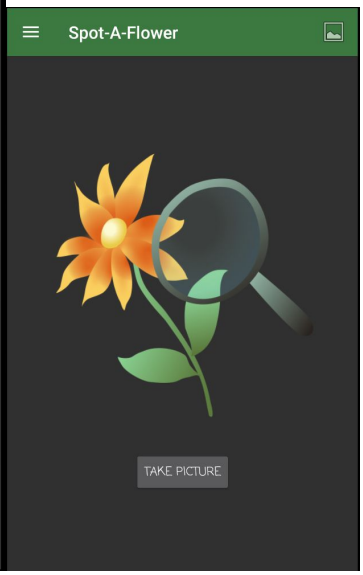
```
android:theme="@style/Theme.AppCompat.DayNight.NoActionBar">
```

However, it does create some problems. My original app colour does not look nice at all in the night mode.



Not a lot of people will care, but I care!

So I did what stack overflow told me: create a night mode colour resources file and change all the colours.



https://material.io/resources/color/#!/?view.left=0&view.right=1&primary.color=2E7D32&secondary.color=009688

I used this site to find an appropriate colour based on my current theme colour. And I flipped the colour for background and text.

```
<resources>
    <color name="colorPrimary">#6BA96D</color>
    <color name="colorPrimaryDark">#009688</color>
    <color name="colorBackground">#FFFFFF</color>
    <color name="colorDescription">#6B6B6B</color>
    <color name="colorTitle">#FFFFFF</color>
```

This is the normal colour resource. The background and description apply to the textview, the title is for the toolbar, navigation bar and the app icon. This part wouldn't change even in night mode (although I can make it do that, but it would be weird to have an app that changes icon colour in night mode).

```
<resources>
    <color name="colorPrimary">#3c7941</color>
    <color name="colorPrimaryDark">#00675b</color>
    <color name="colorBackground">#282828</color>
    <color name="colorDescription">#E6E6E6</color>
```

This the night mode colour resource. Notice that the colour is not exactly back or exactly white, to make it comfortable to human eyes.

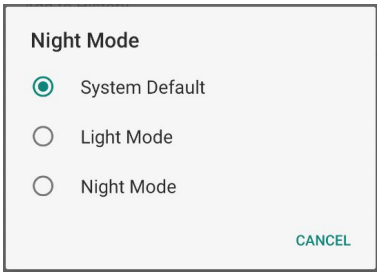
Night Mode Configuration

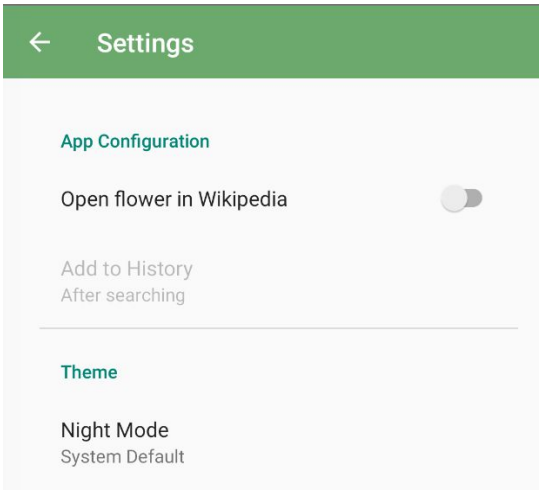
Now, after the app supports the night mode, I need to make a preference in settings that let the user choose the theme.

```
<ListPreference
    app:defaultValue="search"
    app:dependency="openWiki"
    app:entries="@array/history_setting_entries"
    app:entryValues="@array/history_setting_values"
    app:key="addHistoryWhen"
    app:title="@string/add_history"
    app:iconSpaceReserved="false"
    app:useSimpleSummaryProvider="true" />
```

Obviously, I made it a list preference. Just like how I've done it before.

This calls the string array resource to make all the options.





After that is done, I need to make a listener to apply the theme changes.

```
// changes the theme when the user chooses night mode
val theme: Preference? = findPreference( key: "theme")
theme!!.onPreferenceChangeListener =
    Preference.OnPreferenceChangeListener { _, newValue ->
        when (newValue) {
            "system" -> setDefaultNightMode(MODE_NIGHT_FOLLOW_SYSTEM)
            "light" -> setDefaultNightMode(MODE_NIGHT_NO)
            "night" -> setDefaultNightMode(MODE_NIGHT_YES)
        }
        true ^OnPreferenceChangeListener
    }
```

Copying from stackoverflow, I created this block of code in the onCreate method of the setting fragment. Thereby when the theme preference is changed, the new value of the preference determines the mode of the app.

In addition, I found that when the parent preference is set to false, the child preference only dims out and becomes disabled without changing the options back to default. That may confuse the user quite a bit because the summary still shows the previous choice without it being in effect.

So I decided to manually change it back to the default option when the parent preference is changed.

```
// set the history preference to default when disabled
val wiki: Preference? = findPreference( key: "openWiki")
wiki!!.onPreferenceChangeListener =
    Preference.OnPreferenceChangeListener { _, newValue ->
        if (newValue == false)
            findPreference<ListPreference>( key: "addHistoryWhen")!!.value = "search"
        true ^OnPreferenceChangeListener
    }
```

One day I might extract all the key strings to the resource file, but not today.

My app is looking prettier every day thanks to me!

User Account Boolean - isGuest

https://stackoverflow.com/questions/21810240/how-to-create-a-global-variable-in-android

According to this, user information should all be shared preference to survive app restart

I'll wait and see whether it works or not, haha.

Journal 6

Spot-A-Flower - Firebase

Because I am on the way to build the account system, so I thought I can just add the firebase configuration now. That way I would know what to expect and how to do related coding.

https://firebase.google.com/docs/android/setup

Followed the official instruction, which includes changing a bit configuration and manifest et cetera.

I am thinking about the login options. Maybe I should do the google login first bc that is what the codelad taught me. Or maybe I should just do both of them.

I asked Jerry and I'll see what he will respond.

Codelab - Learning Firebase

But after setting it up, I found that I have absolutely no idea how it works.

Fortunately I have found this code lab:
https://codelabs.developers.google.com/codelabs/firebase-android/#0

What you learn to do

- Allow users to sign in.
- Sync data using the Firebase Realtime Database.
- Store binary files in Firebase

Basically, I follow the instruction to build a messenger app, which includes the crucial functions of the database that I can use.

It provides all the lessons I need. So I am happy I found it to teach me how to do this.

https://github.com/firebase/codelab-friendlychat-android SO I cloned this project, and did the following steps:

Step 1-6

Boring stuffs: download and open projects, adding to the firebase console, initializing, and such boring stuff that I already did in my precious Spot-A-Flower app.

Step 7

Firebase Realtime Database Rules

Access to your Firebase Database is configured by a set of rules written in a JSON configuration language.

Go to your project in the Firebase console and select **Database**. Select the **Realtime Database** option (not Cloud Firestore). If prompted for security rules, with choices to start in either **test mode** or **locked mode**, choose **locked mode**. Once the default rules are established, select the **Rules** tab and update the rules configuration with the following:

```
{
  "rules": {
    ".read": "auth != null",
    ".write": "auth != null"
  }
}
```

Click "Publish" to publish the new rules.

For more information on how this works (including documentation on the "auth" variable) see the Firebase security documentation.

Here, I set up the realtime database in the firebase console (on the website). So, the database can only be accessed when the app is authorized.

I have to make an anonymous mode for the user to use the app normally with history and save without an account. There are 3 options: force the user to login, forbid the save functions without account, or make the anonymous mode.

Add the following to the `onCreate` method **after** `mUsername` has been initialized:

MainActivity.java

```
// Initialize Firebase Auth
mFirebaseAuth = FirebaseAuth.getInstance();
mFirebaseUser = mFirebaseAuth.getCurrentUser();
if (mFirebaseUser == null) {
    // Not signed in, launch the Sign In activity
    startActivity(new Intent(this, SignInActivity.class));
    finish();
    return;
} else {
    mUsername = mFirebaseUser.getDisplayName();
    if (mFirebaseUser.getPhotoUrl() != null) {
        mPhotoUrl = mFirebaseUser.getPhotoUrl().toString();
    }
}
```

Then, do this: set up the user: go to the sign page or get the existing user.

MainActivity.java

```
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.sign_out_menu:
                mFirebaseAuth.signOut();
                Auth.GoogleSignInApi.signOut(mGoogleApiClient);
                mUsername = ANONYMOUS;
                startActivity(new Intent(this, SignInActivity.class));
                finish();
                return true;
            default:
                return super.onOptionsItemSelected(item);
        }
    }
```

Now we have all of the logic in place to send the user to the sign-in screen when necessary. Next we need to implement the sign-in screen to properly authenticate users.

Do this to sign out the user.

SignInActivity.java

```
private void signIn() {
    Intent signInIntent = Auth.GoogleSignInApi.getSignInIntent(mGoogleApiClient);
    startActivityForResult(signInIntent, RC_SIGN_IN);
}
```

SignInActivity.java

```
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    // Result returned from launching the Intent from GoogleSignInApi.getSignInIntent(..
    if (requestCode == RC_SIGN_IN) {
        GoogleSignInResult result = Auth.GoogleSignInApi.getSignInResultFromIntent(data)
        if (result.isSuccess()) {
            // Google Sign-In was successful, authenticate with Firebase
            GoogleSignInAccount account = result.getSignInAccount();
            firebaseAuthWithGoogle(account);
        } else {
            // Google Sign-In failed
            Log.e(TAG, "Google Sign-In failed.");
        }
    }
}
```

Add the required `firebaseAuthWithGoogle` method to authenticate with the signed in Google account:

Do this to sign in the user…

SignInActivity.java

```
private void firebaseAuthWithGoogle(GoogleSignInAccount acct) {
    Log.d(TAG, "firebaseAuthWithGooogle:" + acct.getId());
    AuthCredential credential = GoogleAuthProvider.getCredential(acct.getIdToken(), null
    mFirebaseAuth.signInWithCredential(credential)
            .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    Log.d(TAG, "signInWithCredential:onComplete:" + task.isSuccessful())

                    // If sign in fails, display a message to the user. If sign in succe
                    // the auth state listener will be notified and logic to handle the
                    // signed in user can be handled in the listener.
                    if (!task.isSuccessful()) {
                        Log.w(TAG, "signInWithCredential", task.getException());
                        Toast.makeText(SignInActivity.this, "Authentication failed.",
                                Toast.LENGTH_SHORT).show();
                    } else {
                        startActivity(new Intent(SignInActivity.this, MainActivity.class
                        finish();
                    }
                }
            });
}
```

Do this to ensure the credential is passed and return the user to the main activity.

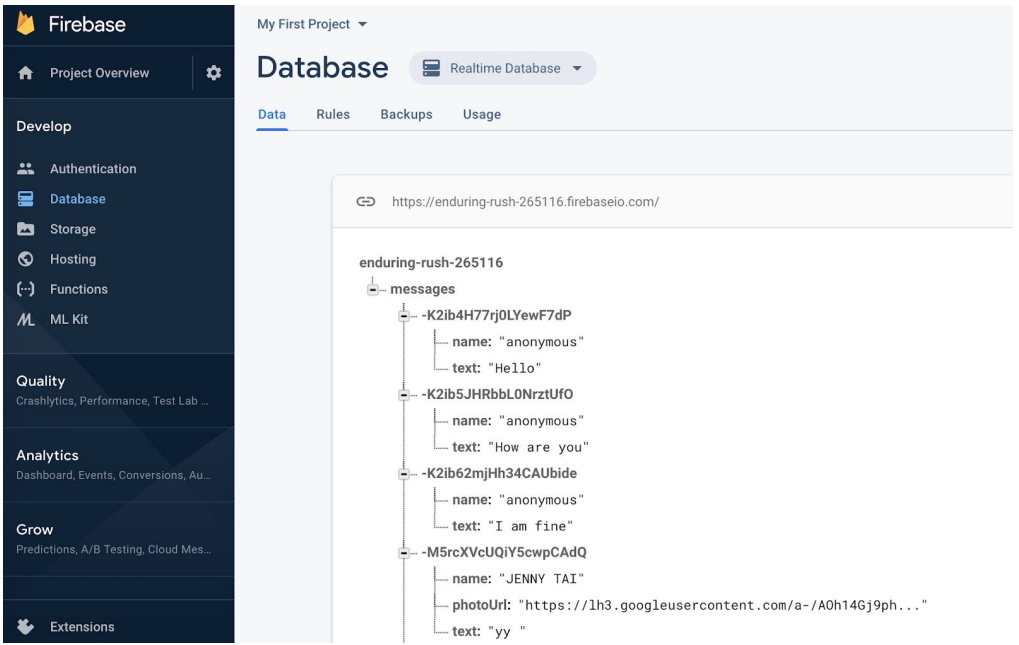
So this is all the steps about google authorization. I am pretty sure email would work similarly. So next week I'll make a login page and copy in these codes to see if it works, then work my way toward email/password login.

Step 8

By adding a realtime database in the firebase console, I am now able to upload and download user information from the firebase.

Then, the codelab includes a long list of codes without explanation…

I am trying so hard to understand it…



```
// New child entries
mFirebaseDatabaseReference = FirebaseDatabase.getInstance().getReference();
SnapshotParser<FriendlyMessage> parser = new SnapshotParser<FriendlyMessage>() {
    @Override
    public FriendlyMessage parseSnapshot(DataSnapshot dataSnapshot) {
        FriendlyMessage friendlyMessage = dataSnapshot.getValue(FriendlyMessage.class);
        if (friendlyMessage != null) {
            friendlyMessage.setId(dataSnapshot.getKey());
        }
        return friendlyMessage;
    }
};
```

For this part, by my understanding, it creates a parser that is able to parse the info in the database to the message class.

```
DatabaseReference messagesRef = mFirebaseDatabaseReference.child(MESSAGES_CHILD);
FirebaseRecyclerOptions<FriendlyMessage> options =
        new FirebaseRecyclerOptions.Builder<FriendlyMessage>()
                .setQuery(messagesRef, parser)
                .build();
```

Then, this copies the message section from the database, and builds "options" (unknown stuff) by the database and the parser.

```
mFirebaseAdapter = new FirebaseRecyclerAdapter<FriendlyMessage, MessageViewHolder>(options) {
    @Override
    public MessageViewHolder onCreateViewHolder(ViewGroup viewGroup, int i) {
        LayoutInflater inflater = LayoutInflater.from(viewGroup.getContext());
        return new MessageViewHolder(inflater.inflate(R.layout.item_message, viewGroup,  attachToRoot: false));
    }
```

The option is passed to the adapter. This is just normal adapter stuff.

```
@Override
protected void onBindViewHolder(final MessageViewHolder viewHolder,
                                int position,
                                FriendlyMessage friendlyMessage) {
    mProgressBar.setVisibility(ProgressBar.INVISIBLE);
    if (friendlyMessage.getText() != null) {
        viewHolder.messageTextView.setText(friendlyMessage.getText());
        viewHolder.messageTextView.setVisibility(TextView.VISIBLE);
        viewHolder.messageImageView.setVisibility(ImageView.GONE);
```

Then goes into the holder section. Somehow the message can be used directly. Like, is that it? All the things above just made it work? Somehow? Anyway, just put the information to the holder.

Making a progress bar may be a good idea. I have to try how long it takes my program to do the login thing. If it takes longer than a second i'll just add a progress bar.

```
} else if (friendlyMessage.getImageUrl() != null) {
    String imageUrl = friendlyMessage.getImageUrl();
    if (imageUrl.startsWith("gs://")) {
        StorageReference storageReference = FirebaseStorage.getInstance()
                .getReferenceFromUrl(imageUrl);
        storageReference.getDownloadUrl().addOnCompleteListener(
                new OnCompleteListener<Uri>() {
                    @Override
                    public void onComplete(@NonNull Task<Uri> task) {
                        if (task.isSuccessful()) {
                            String downloadUrl = task.getResult().toString();
                            Glide.with(viewHolder.messageImageView.getContext())
                                    .load(downloadUrl)
                                    .into(viewHolder.messageImageView);
                        } else {
                            Log.w(TAG,  msg: "Getting download url was not successful.",
                                    task.getException());
                        }
                    }
                });
```

Then this gets really really weird. I know it gets profile pictures from the database. When i tried using a google account, my profile picture is just stated as "https://lh3.googleusercontent.com/a-/AOh14Gj9ph4HxzqYiRjWq7UG1y0yOJYTcrx9ZywNQtFo=s96-c", which has nothing to do with"gs://". I have no idea why code would list that. Maybe I should just delete it.

```
    } else {
        Glide.with(viewHolder.messageImageView.getContext())
                .load(friendlyMessage.getImageUrl())
                .into(viewHolder.messageImageView);
    }
    viewHolder.messageImageView.setVisibility(ImageView.VISIBLE);
    viewHolder.messageTextView.setVisibility(TextView.GONE);
}
```

Anyway, normally, the code just replaces the imageview with the correct profile picture.

So this is the basic steps to download info from the database. To upload, just do "mFirebaseDatabaseReference.child(MESSAGES_CHILD).push().setValue(friendlyMessage);"

Then it pushes this message to the message section of the firebase database.

```
@Override
public void onPause() {
    mFirebaseAdapter.stopListening();
    super.onPause();
}

@Override
public void onResume() {
    super.onResume();
    mFirebaseAdapter.startListening();
}
```

Beside all of that, this is crucial to make the program properly run. Don't know why it is just required.

Step 9 includes using online storage. It is not needed (hopefully)in Spot-A-Flower so I just didn't do it.

So next week I'll implement all of these to Spot-A-Flower.

Journal 7

Apr 29, Wednesday

Plan

I asked Jerry, he agrees with me, that Google login is simply enough for this project. So there is no need to create a special login page with email and password. Just need to add the google login option to the drawer menu and directly link to google.

The goal is to authenticate through firebase, which allows the user log in, log out and to store flowers.

BTW, forgot to capture codes while writing it, so i'll just add an image of the git commit history.

Google Login

Basically redo the steps in the codelab:

Declare variables:

```
// Firebase instance variables
private lateinit var mFirebaseAuth: FirebaseAuth
private lateinit var mFirebaseUser: FirebaseUser
```

Configure google sign in

```
// Configure Google Sign In
val gso : GoogleSignInOptions! = GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
    .requestIdToken("70865465845-usm8bjek97b8nit35dvlrjh3n1g994le.apps.googl...")
    .requestEmail()
    .build()
val mGoogleApiClient : GoogleApiClient! = GoogleApiClient.Builder(this)
    .enableAutoManage(this /* FragmentActivity */, null /* OnConnectionFailedListener */)
    .addApi(Auth.GOOGLE_SIGN_IN_API, gso)
    .build()
```

update UI is a private method which I will show later. It is easier to be called and clearer that way.

```
// Initialize Firebase Auth
mFirebaseAuth = FirebaseAuth.getInstance()
updateUI()
```

Login or Log out when the drawer menu option is clicked. Login goes to the sign in request

```
R.id.account -> {
    drawer_layout.closeDrawer(GravityCompat.START)
    if (mFirebaseAuth.currentUser == null) {
        // log in account
        progressBar.isVisible = true
        val signInIntent : Intent! =
            Auth.GoogleSignInApi.getSignInIntent(mGoogleApiClient)
        startActivityForResult(signInIntent, signInRequest)
    } else {
        // sign out account
        mFirebaseAuth.signOut()
        Auth.GoogleSignInApi.signOut(mGoogleApiClient)
        Toast.makeText( context: this,  text: "Your account is signed out", Toast.LENGTH_SHORT)
            .show()
        updateUI()
    }
    true ^setNavigationItemSelectedListener
```

Sign in request then calls the authwithgoogle method to set the account as a result

```
// Result returned from launching the Intent from GoogleSignInApi.getSignInIntent(...);
else if (requestCode == signInRequest) {
 val result : GoogleSignInResult? = Auth.GoogleSignInApi.getSignInResultFromIntent(data)
 if (result!!.isSuccess) {
     // Google Sign-In was successful, authenticate with Firebase
     val account: GoogleSignInAccount = result.signInAccount!!
     firebaseAuthWithGoogle(account.idToken!!)
 } else {
     Toast.makeText( context: this,  text: "Google Login Failed", Toast.LENGTH_SHORT).show()
 }
```

Like this

```
// sign in from firebase
private fun firebaseAuthWithGoogle(acct: String) {
    val credential : AuthCredential = GoogleAuthProvider.getCredential(acct, null)
    mFirebaseAuth.signInWithCredential(credential)
        .addOnCompleteListener(this) { task ->
            if (task.isSuccessful) {
                Toast.makeText( context: this,  text: "Authentication Successful!", Toast.LENGTH_SHORT).show()
                progressBar.isVisible = false
                updateUI()
            } else {
                Toast.makeText( context: this,  text: "Authentication Failed.", Toast.LENGTH_SHORT).show()
            }
        }
}
```

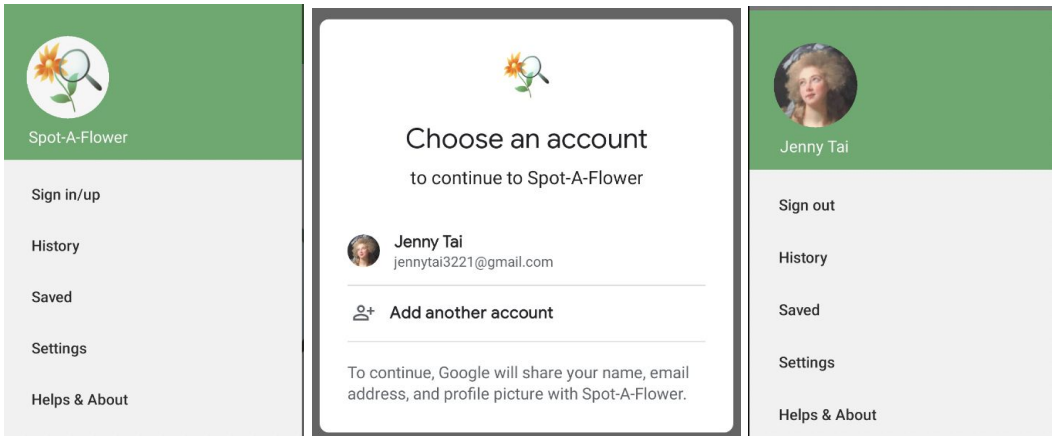
Both when the login and logout and initialization in onCreate, the program call calls the updateUI.

It just basically changes the drawer header to match with user information, and the menu option title to match with current login status.

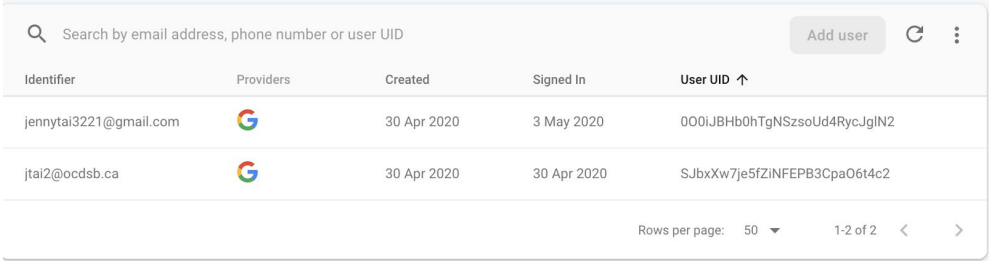
```
// change UI depending on the user login or not
private fun updateUI() {
    if (mFirebaseAuth.currentUser != null) {
        // sign in
        mFirebaseUser = mFirebaseAuth.currentUser!!
        nav_view.getHeaderView( index: 0).username.text = mFirebaseUser.displayName
        if (mFirebaseUser.photoUrl != null) {
            val profilePicture :Bitmap! =
                BitmapFactory.decodeStream(
                    URL(mFirebaseUser.photoUrl.toString()).content as InputStream?
                )
            nav_view.getHeaderView( index: 0).user_profile.setImageBitmap(profilePicture)
        }
        nav_view.menu.findItem(R.id.account).title = "Sign out"
    } else {
        // sign out
        nav_view.getHeaderView( index: 0).username.text = "Spot-A-Flower"
        nav_view.getHeaderView( index: 0).user_profile.setImageResource(R.mipmap.ic_launcher)
        nav_view.menu.findItem(R.id.account).title = "Sign in/up"
    }
}
```

So, a successful login process would include:

Clicking sign in from the menu, having a google dialog to sign up google account, having a toast showing "authentication successful", and having user information recorded.



And the user information will show up in my firebase database:



By the way, besides the thing codelab told me to do, i have to add this to avoid error message:

```
// do this to get internet connection
val policy: StrictMode.ThreadPolicy = StrictMode.ThreadPolicy.Builder().permitAll().build()
StrictMode.setThreadPolicy(policy)
```

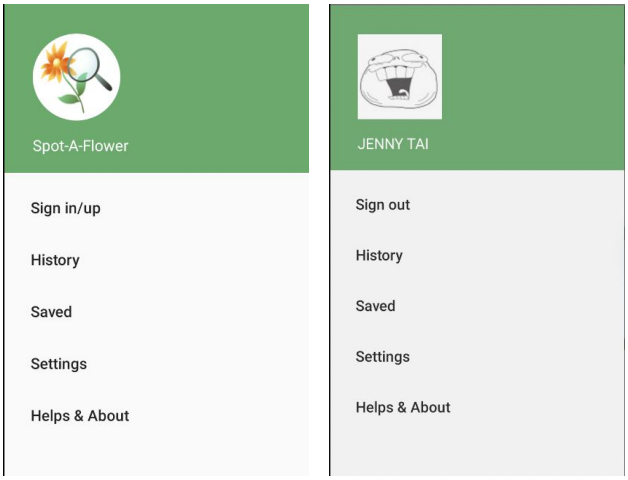
I guess something about my project internet policy is different from the codelab. But it works anyway, peww.

Apr 30, Thursday

Header layout

Changed the layout of the leader from linear to constraint and adjusted many padding due to my OCD.

This was my original header from Journal 2 (can't believe I did so much in the first few weeks, this is so far back), compared to the on the right we can see that there are slight differences…



But that is not the main thing. The user profile picture as an ImageView is a square!!! And that does not match with my artistic values!

I search the internet and developed a really cute method to make a circle:

I added an empty circle png file, and tint it green and put it on the profile picture to cover up the ugly squareness.

Now it just looks like a normal circle.

Cute isn't it

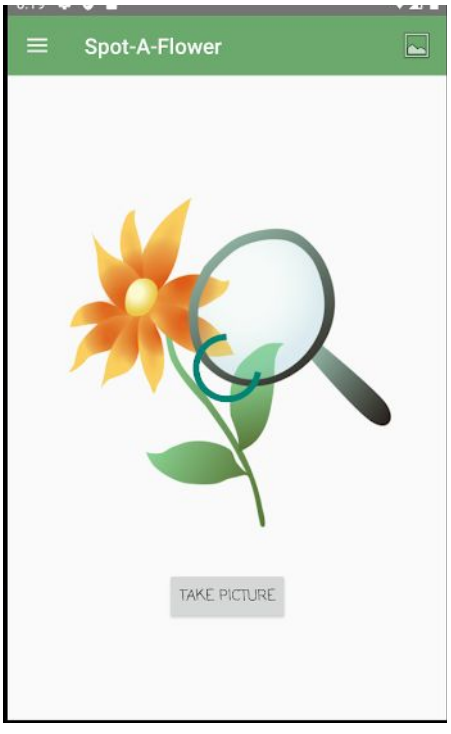
Progress bar

The progress is something that I found had to be done when I was testing. Waiting for the google login to load seems to take forever when the user has no idea what is going on. The loading bar tells them the app is progressing, and gives them something to look at when they are bored.

I thought about making a special "spinning flower" animation, but nah. Not like I have nothing else to do.

So instead I created a normal system default circular progress bar:

With the green rolling circle thing in the middle. The main activity calls progressbar.visible = true when the sign in button is pressed, and progressbar.visible = false when the google sign in request is ended.



May 1 - 2, Friday - Saturday

Overview of Database

In the process of database, I have rewritten it so many times in so many different ways. So I will not describe my thought process, I will just state the final correct way to do this.

I think the reason is that I have little knowledge of how to make the most convenient data structure in the database.

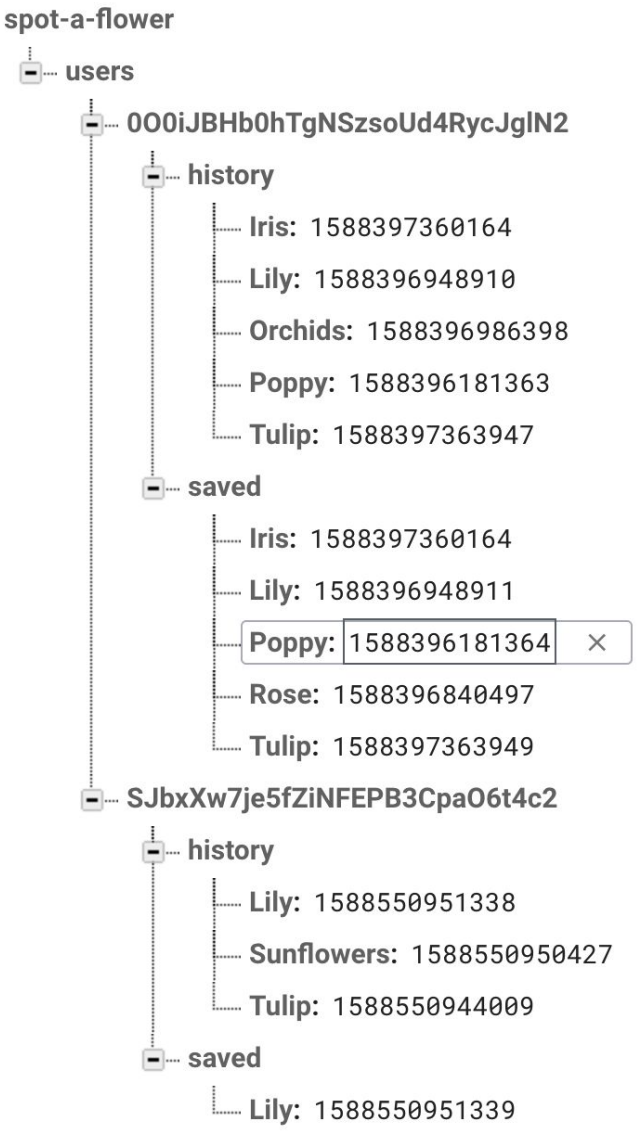
Database

This is the data structure of the user database. Each user has a unique id generated by google, and each user has a history database and a saved flower database. The existence of the flower name means they saved it, and the timestamp shows when they saved it.

I originally only save flower names in the saved database, and reference the timestamp of the flower in history. But then when the history is deleted, the saved flower would not have a valid timestamp.

I guess the point is, does the timestamp of the flower go with the time when it is saved to history, or the time it is saved in saved (stared)? Do they stay the same in both databases?

I chose to record them separately, and keep them separate. But I guess I can renew the saved timestamp to sync with the history timestamp. I chose not to. I instead renew the history timestamp with the saved one. Perhaps I can make it a preference.



Flower Class

This is the class for flowers. I made the timestamp as a string in the class because it is easy to just pass in the timestamp as long from the database, and format it here in the constructor.

```
class Flower {
    var name: String
    var detail: String

    constructor(name: String, time: Long) {
        this.name = name
        val sdf = SimpleDateFormat( pattern: "hh:mm:s
        this.detail = sdf.format(Date(time))
    }

    constructor(name: String, probability: Int) {
        this.name = name
        this.detail = "$probability% Probability"
    }
```

The flower has another constructor, because the flower shows the possibility in the search scenario, so the wording of the detail changes accordingly.

Flower Search Activity

Initialization

The database and user must be declared and initialized to use in later functions.

```
// Firebase instance variables
private var mFirebaseAuth: FirebaseAuth = FirebaseAuth.getInstance()
private var database: DatabaseReference = Firebase.database.reference
```

Set up Layout

```
// set up layout
setContentView(R.layout.activity_search_success)
recyclerView = findViewById<RecyclerView>(R.id.flower_list).apply {
    setHasFixedSize(true)
    layoutManager = viewManager
    adapter = viewAdapter
}

// set up toolbar
setSupportActionBar(findViewById(R.id.toolbar))
supportActionBar?.setDisplayHomeAsUpEnabled(true)
supportActionBar?.title = intent.getStringExtra( name: "Parent")
```

The recycler view and the toolbar needs to be set to the interface.

When Searching Flower

```
"Spot-A-Flower" -> {
    for (i :Int in 1..5) {
        // making up variables, for now TODO neural network
        val name: String = names[(Math.random() * names.size).toInt()]
        myDataset.add(Flower(name, (Math.random() * 100).toInt()))

        // save the flower to history when search if the user choose so
        val sharedPreferences :SharedPreferences! = PreferenceManager.getDefaultSharedPreferences(
        if (sharedPreferences.getString( key: "addHistoryWhen", defValue: "search") == "search") {
            mFirebaseAuth.currentUser?.uid?.let { it: String
                database.child( pathString: "users").child(it).child( pathString: "history")
                    .child(name).setValue(System.currentTimeMillis())
```

When the user searches for a flower, the flower dataset should be filled with flowers from the neural networks depending on the photo bitmap. For now it is just randomly determined.

In addition, if the setting preference says it needs to save the flower to the user history after searching, the flower will be stored to the firebase under this specific user (this user UID) under history.

It tells the database to put the current timestamp under the flower name.

When History Changes

```
"History" -> {
    // read history, turn to flower, and add to dataset
    mFirebaseAuth.currentUser?.uid?.let { it: String
        database.child( pathString: "users").child(it).child( pathString: "history").orderByValue()
            .addValueEventListener(object : ValueEventListener {
                override fun onDataChange(dataSnapshot: DataSnapshot) {
                    myDataset.clear()
                    for (flowerSnapshot : DataSnapshot!  in dataSnapshot.children) {
                        myDataset.add(
                             index: 0,
                            Flower(
                                flowerSnapshot.key!!,
                                flowerSnapshot.value as Long
                            )
                        )
                    }
                    viewAdapter.notifyDataSetChanged()
                    if (myDataset.size == 0) pageEmpty()
                }

                override fun onCancelled(databaseError: DatabaseError) {
                    Toast.makeText(
                         context: this@FlowerSearch,
                         text: "Error loading user database",
                        Toast.LENGTH_SHORT
                    ).show()
                    Log.w( tag: "TAG",  msg: "loadData:onCancelled", databaseError.toException())
                }
```

The history page needs to read the database by using a value event listener. It goes to the data snapshot under the current user history, read the flower and add the flower class to the current dataset.

Because the dataset is ascendingly ordered by timestamp, I add the flower backward to reverse the array to make it descending. If I have time, I could add a choice in the menu, to choose the sorting method (by name or by time).

The onDataChange method is called when it is initialized and whenever the database information changes. So when something changes like when the user saved the flower with a new timestamp, it calls the viewAdapter, which just changes the recycler view accordingly.

The best thing about it is that it doesn't refresh the dataset, it just updates the changed view, fluently.

If the dataset becomes empty after certain changes, like deleting all history and such, the program calls the pageEmpty private method to change the layout to the searchFail page with a warning message.

When Saved Flower Changes

It actually has the exact same codes with the history page, except that it reads the saved database not the history database. I can actually make it the same value change listener, but in case it changes for future development, i'll keep it as it is for now. (because it happened quite a few time)

When Dataset Empty

```
// if dataset empty, all goes to fail page
private fun pageEmpty() {
    setContentView(R.layout.activity_search_failed)
    when (intent.getStringExtra( name: "Parent")) {
        "History" -> {
            findViewById<ImageView>(R.id.failImage)
                .setImageResource(android.R.drawable.ic_menu_myplaces)
            findViewById<TextView>(R.id.failText).text = "Come out and search some flowers!"
        }
        "Saved" -> {
            findViewById<ImageView>(R.id.failImage)
                .setImageResource(android.R.drawable.ic_menu_myplaces)
            findViewById<TextView>(R.id.failText).text = "Save flowers by tapping that star!"
        }
        else -> {
            findViewById<TextView>(R.id.failText).text = "Sorry, search failed"
        }
    }
    // set up toolbar
    setSupportActionBar(findViewById(R.id.toolbar))
    supportActionBar?.setDisplayHomeAsUpEnabled(true)
    supportActionBar?.title = intent.getStringExtra( name: "Parent")
}
```

The activity reset the view to the search failed page.

The toolbar needs to be set again because the content view changed.

Currently when the user is not signed in, it doesn't show search failed, it just shows empty recyclerview because it can't access the user database. I can either add an anonymous mode where it also saves to the database, or it just goes to another search failed page saying "sign up to use this function". We'll see.

Delete User History

```
lertDialog.Builder( context: this)
    .setTitle("Delete History")
    .setMessage(
        "Are you sure to delete user history? " +
                "Your history can not be restored. "
    )
    .setPositiveButton(
        android.R.string.yes
    ) { _, _ ->
        // delete user history in database
        mFirebaseAuth.currentUser?.uid?.let { it: String
            database.child( pathString: "users").child(it).child( pathString: "history").removeValue()
        }
        pageEmpty()
    }
    .setNegativeButton(android.R.string.no,  listener: null)
    .show()
```

It just calls the database to remove those values. Pity. And then the page is empty again, so calls the search fail page function to show the user is empty.

RecyclerView

Initialization

```
// Firebase instance variables
private var mFirebaseAuth: FirebaseAuth = FirebaseAuth.getInstance()
private var database: DatabaseReference = Firebase.database.reference
```

```
val flower :Flower  = Flowers[position]
holder.name.text = flower.name

// TODO get these info from flower database
holder.description.text =
    "                          " + "Lily (membe
holder.detail.text = flower.detail
holder.icon
```

In the onBindViewHolder, the flower information should be gotten from a flower database, with information like description and pictures.

I will do that soon.

Because I don't yet know what flowers are there in the neural networks, so I still have to do that.

Get Saved Status

```
// get save status
mFirebaseAuth.currentUser?.uid?.let { it: String
    database.child( pathString: "users").child(it).child( pathString: "saved").child(flower.name)
        .addValueEventListener(object : ValueEventListener {
            override fun onDataChange(dataSnapshot: DataSnapshot) {
                // initialize and update the  the UI
                if (dataSnapshot.exists()) {
                    holder.saveButton.tag = 1
                    holder.saveButton.setImageResource(android.R.drawable.star_on)
                } else {
                    holder.saveButton.tag = 0
                    holder.saveButton.setImageResource(android.R.drawable.star_off)
                }
            }
            override fun onCancelled(error: DatabaseError) {
                Toast.makeText(context,  text: "Error loading user database", Toast.LENGTH_SHORT
                    .show()
                Log.w( tag: "TAG",  msg: "loadData:onCancelled", error.toException())
            }
        })
}
```

The colour of the star in the flower card view must match with the database information.

So, it access the flower under saved, detect whether it is there, and update the UI

It also changes the tag for the click listener to know what to do.

When open Wiki Link

```
// if the user choose to save to history when open link
if (sharedPreferences.getString( key: "addHistoryWhen",  defValue: "search") == "link") {
    saveTo( branch: "history", flower)
}
```

Pretty straight forward. I made a private function to do this kind of stuff.

```
private fun saveTo(branch: String, flower: Flower) {
    println("save ${flower.name} to $branch")
    mFirebaseAuth.currentUser?.uid?.let { it: String
        database.child( pathString: "users").child(it).child(branch)
            .child(flower.name).setValue(System.currentTimeMillis())
    }
}
```

Like this.

Star Click Listener

```
// change star and save to database
holder.saveButton.setOnClickListener { it: View!
    //holder.saveButton.startAnimation(AlphaAnimation(1.0f, 0.2f))
    if (holder.saveButton.tag == 1) {
        holder.saveButton.tag = 0
        holder.saveButton.setImageResource(android.R.drawable.star_off)
        // delete saved flower from user database
        println("cancel storing " + flower.name)
        mFirebaseAuth.currentUser?.uid?.let { it: String
            database.child( pathString: "users").child(it).child( pathString: "saved")
                .child(flower.name).removeValue()
        }
    } else {
        holder.saveButton.tag = 1
        holder.saveButton.setImageResource(android.R.drawable.star_on)
        // store the flower to saved database
        saveTo( branch: "history",flower)
        saveTo( branch: "saved", flower)
    }
}
```

Value captured in a closure

When the star is clicked, save and delete accordingly. The history timestamp is actually synced with the saved timestamp.

I could actually don't do the tag and image change because it is already linked with the database. However, just to be safe.

And that is it for the user database. It works wonderfully, for now. More testing is needed to see.

May 3, Sunday

Bug

The star function is messed up. The order of the flower card is renewed by the staring in the history, however, the star depends on the current position of the card, so it ends up starring the wrong flower.

So I must not sync the history timestamp with the star timestamp…

So I canceled that function, and I am thinking that I can still update the saved timestamp by the history timestamp according to user preference.

Celebration

50 commits celebration!!!!!

Finished all these things!!! And database on time! Great!!

The only major thing left I need to do is the flower database, using room

Others is just integrating the neural network

Wowwee

Journal 8

May 6, Wednesday (Planning & Researching)

- Integrating Neural Network
- Flower Database using Room
- Anonymous Mode (Prohibit functions or store as device in database)
- Tutorial (Need Brainstorming first)

Integrating Neural Network

The neural network must wait until Jerry finishes it. I have no idea of his process (shy to ask).

If he finishes it, it would take less than a week to integrate it to the app. If he doesn't finish it, I can just do it using tensorflow lite and it would take a week and a half. Not a big deal.

Flower Database using Room

I plan on doing the flower database soon.

The Android official tutorial is always useful. Save data in a local database using Room.

Although I am very confused after reading it once, I also found a lot of blogs that can help me do this, such as Using Room Database | Android Jetpack - MindOrks and Android Room Tutorial: Simplifying How You Work with App Data. I should be fine. This would also take a week or so.

Anonymous Mode

The anonymous mode is something that I am considering. I don't know if i should allow the user to store flowers anonymously. If I do, I must make dialogs to remind the user to make an account. But when?

If I remind them every time they save flowers, It would be so redundant! If I remind them every time they open the save activity, they would not know their flowers are not saved at the beginning.

I guess I can remind them when the app is started and saved pages are opened. Need more thinking.

https://firebase.google.com/docs/auth/android/anonymous-auth

Tutorial

There are a lot of libraries online that can do an app intro. Using shared preference, i can make a boolean and detect whether the app is opened for the first time, and show the tutorial activity.
Determine if Android app is being used for the first time

http://worker8.github.io/TourGuide/#/?id=tour guide

In addition, I did the tutorial switch in the settings page already. I can use that to show labels to tell the user where to click for certain functions.

May 7, Thursday

Change of Plan

The following text is written for the sqlite intro:

Caution: Although these APIs are powerful, they are fairly low-level and require a great deal of time and effort to use:

- There is no compile-time verification of raw SQL queries. As your data graph changes, you need to update the affected SQL queries manually. This process can be time consuming and error prone.
- You need to use lots of boilerplate code to convert between SQL queries and data objects.

For these reasons, we **highly recommended** using the Room Persistence Library as an abstraction layer for accessing information in your app's SQLite databases.

When I saw this text, I thought, great, Room is more advanced so let us use Room.

BUT IT WILL BE GOOD PRACTICE to learn how to do SQLite!!!! It is used in a lot more situations, and I can learn how to use it! Even though it may be more time consuming, it would be fun!!!!

Which means I need to research again…

Useful websites list:

https://www.javatpoint.com/kotlin-android-sqlite-tutorial

https://developer.android.com/training/data-storage/sqlite

https://stackoverflow.com/questions/9357668/how-to-store-image-in-sqlite-database

https://stackoverflow.com/questions/53219223/get-specific-row-of-sqlite-android

https://tutorial.eyehunts.com/android/sqlite-database-in-android-kotlin-example/

Database

From the tutorial i am looking at, they have id as integer, and they set that as "INTEGER PRIMARY KEY". I tried using the name of the flower as the primary key, because I didn't notice that it can only be integer. So now I am using the auto generated id, with a column for the flower name.

May 8, Friday

Yesterday I was too angry to do journalling.

I wrote a sqlite database handler. No big deal.

You know when people are learning while coding, they usually change a lot of things. That never was a problem before.

This time, the sqlite database keeps track of its version, and when it is created, it never reconstructs itself to the newer version of database setting. It needs to be upgraded manually.

It is a simple mistake. But how long did it take to figure it out?

I SAT AT MY DESK FOR 6 HOURS YESTERDAY

I GO TO SLEEP FRUSTRATED QUESTIONING MY *SANITY*

AND I FOUND OUT AFTER I REDO IT TODAY

After a lot of work and testing and such, the database is working.

I can put in a flower information manually when the program is running.

So now I need to pre populate the database, so it can just use the information freely.

SQLite Database

```
class FlowerInfoDB(private val context: Context) :
    SQLiteOpenHelper(context, DB_NAME, factory: null, DB_VERSION) {

    companion object {
        private const val DB_NAME = "FlowerInfoDB"
        private const val DB_VERSION = 1
        private const val TABLE_NAME = "FlowerInfo"
        private const val ID = "id"
        private const val NAME = "name"
        private const val INTRO = "intro"
        private const val ICON = "icon"
    }
```

This is the class for my flower database. The class and inheritance is setted for the sqlite database. The companion object includes all of the string that I need as the setting of the database. Straight forward. The Id is generated by the database, the name is for the flower name, the intro is the description of the flower that needs to be shown in the flower card, the icon is a picture of the flower. Although the database is not set up yet, I will do that, by creating a sqlite database and importing that to my android app.

```
override fun onCreate(db: SQLiteDatabase) {
    val CREATE_TABLE :String  = "CREATE TABLE $TABLE_NAME " +
            "($ID Integer PRIMARY KEY, $NAME TEXT, $INTRO TEXT, $ICON BLOB)"
    db.execSQL(CREATE_TABLE)
}
```

When the app is created, this table is created using sqlite.

```
override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int)
    db.execSQL( sql: "DROP TABLE IF EXISTS $TABLE_NAME")
    onCreate(db)
```

When the database upgrade, the database just deletes itself and recreates it, for now.

This is the step where I did wrong previously. I had no idea I needed to upgrade it.

These sql languages are completely foreign to me. I used a lot of resources like https://developer.android.com/training/data-storage/sqlite#WriteDbRow, and the list I posted before.

Although very frustrating, learning is fun sometimes.

```
//Inserting (Creating) data
fun addFlower(name: String, description: String, bitmap: Bitmap) {
    val stream = ByteArrayOutputStream()
    bitmap.compress(Bitmap.CompressFormat.PNG, quality: 90, stream)
    val image : ByteArray = stream.toByteArray()

    val db : SQLiteDatabase! = this.writableDatabase
    val values : ContentValues = ContentValues().apply { this: ContentValues
        put(NAME, name)
        put(INTRO, description)
        put(ICON, image)
    }
    db.insert(TABLE_NAME, nullColumnHack: null, values)
    db.close()
}
```

When testing, I have to test whether the database could add flowers to the database at runtime (bc I don't know how to prepopulate the database yet). So I input the image as a bitmap through the camera, and insert that into the table.

```
fun getDescription(name: String): String {
    val db : SQLiteDatabase! = this.readableDatabase
    val cursor : Cursor! = db.rawQuery( sql: "SELECT * FROM $TABLE_NAME WHERE $NAME=?", arrayOf(name))
    val intro : String = "                          " + if (cursor.moveToFirst()) {
        cursor.getString(cursor.getColumnIndex(INTRO))
    } else {
        "Lily (members of which are true lilies) is a genus of herbaceous flowering plants growing
    }
    cursor.close()
    db.close()
    return intro
}
```

This is for getting the description for a certain flower. The cursor finds the row where name = the given name. However, if the cursor cannot find the row with the same name (only happens now bc the database is unfinished. Should not happen later, so needs to change that to "Cannot load database"), it shows the default phrase.

I have the same thing for icons as well. The two functions are necessary for the app to get flower information for the cards during run time.

```
fun printAllFlowers() {
    var allFlower = ""
    val db :SQLiteDatabase! = readableDatabase
    val selectALLQuery = "SELECT * FROM $TABLE_NAME"
    val cursor :Cursor! = db.rawQuery(selectALLQuery, selectionArgs: null)
    if (cursor != null) {
        if (cursor.moveToFirst()) {
            do {
                val id :String! = cursor.getString(cursor.getColumnIndex(ID))
                val name :String! = cursor.getString(cursor.getColumnIndex(NAME))
                val description :String! = cursor.getString(cursor.getColumnIndex(INTRO))
                val icon :ByteArray! = cursor.getBlob(cursor.getColumnIndex(ICON))
                allFlower = "$allFlower\n$id $name $icon $description"
            } while (cursor.moveToNext())
        }
    }
    cursor.close()
    db.close()
    println(allFlower)
}
```

This thing is just for debugging. I need to see what is in the database. So the cursor goes through all of the row and prints all of the information.

The database works inside my app after testing. When I pass in a bitmap, it can store it as a blob and the app can use the information on the flower cards.

Now I need to do a sql database somewhere else, and import it to my app.

I have found the following methods:

Ship an Android app with a pre-populated database This one put the database to asset and read it using input stream and output stream

https://github.com/jgilfelt/android-sqlite-asset-helper or i can use this library if the above doesn't work.

Journal 9

May 15, Friday

Sent a message to Jerry *last Wednesday* asking for the training dataset. I wanted to see if the dataset can be used in the database. I need to know the type of flowers to get relevant information, and maybe i can use the dataset structure and alter it to a sql database somehow.

The message is still not read! Although I kinda expected this, but still…

Without relevant information, I can not fill the database with predetermined information. How do I write the description of the flower when I don't even know the name of that flower?

Of the following things, the only thing I can currently do is the anonymous mode.

- Integrating Neural Network
- Flower Database
- Anonymous Mode (Prohibit functions or store as device in database)
- Tutorial (Need Brainstorming first)

I really need to talk to him to see the future of this project.

I need time to make a decision about what to do.

May 16, Saturnday

That does it. I will send him an email saying that, if he still doesn't respond to me next week, I will just do the neural network on my own.

It is not like I can't do it or anything, I don't need to rely on him for this project!

I mean, the message is not even read!

Before next friday, I will work on the anonymous mode:
https://firebase.google.com/docs/auth/android/anonymous-auth

Trying to do this. If it doesn't work, I will just do a warning about not signing in.

And then, if he still doesn't respond, I do the neural network!

May 17, Sunday

Anxiously waiting for a response…

To fill the time, I researched the neural network for android apps. Surprisingly firebase already has this function (can't believe it is so convenient, maybe it is a bad choice to use it since it does everything for me).

First I need to train a neural network (obviously). Of course I would look into the classic iris sorting algorithm. https://codelabs.developers.google.com/codelabs/mlkit-android-custom-model/#6

It is one of the classics with numerous resources online. I think I even watched a crash course on it. Seems straight forward. After I learned how to sort iris, I can change the dataset and make it sort other flowers as well.

There are a lot of flower dataset online. Many of them only have 5-10 common flowers, which is not the point of this app. This one however, has 102 categories, seems good. Visual Geometry Group

After the model is setted up, the firebase thing can just load the model and use it to determine the flower. https://codelabs.developers.google.com/codelabs/mlkit-android-custom-model/#6 This codelab teached it, and there is also this: Manage your hosted models

Journal 10

Caught a Cold (Not Covid-19)

I tried to start working on comp sci on Wednesday as usual (doing english and math on Monday and Tuesday), but sneezed 7 times in 1 hour.

Seeing this trend I decided to rest on Wednesday to prevent being sick, it didn't work.

Still sick on Thursday, so I slept.

On Friday, I ate soup and slept.

On Saturday I felt better but not better enough to think logically and do work.

On Sunday I am basically well again besides occasionally sneezing. But like, I haven't done work all week, I'll just start a new one on Monday, so I didn't work.

It is *definitely* not covid-19 since I haven't gone out in 2 month.

Anyway sorry for being sick, that is now over.

I will really start doing work soon!

Journal 11

May 25, Monday

Anonymous Warning

When the user opens up the saved page or the history page, the program detects whether there is a user, and goes to the warning page.

```
// if user not signed in and open saved or history, warn
if (mFirebaseAuth.currentUser == null && intent.getStringExtra( name: "Parent") != "Spot-A-Flower"){
    pageEmpty()
} else progressBar2.isVisible = true
```

Like before, the page empty is dedicated to change the layout to the fail page. It changes the textview to corresponding warning.

```
// if dataset empty, all goes to fail page
private fun pageEmpty() {
    setContentView(R.layout.activity_search_failed)
    if (mFirebaseAuth.currentUser == null){
        findViewById<TextView>(R.id.failText).text = "Sign up or login to use this function!"
    } else when (intent.getStringExtra( name: "Parent")) {
        "History" ->
            findViewById<TextView>(R.id.failText).text = "Come out and search some flowers!"
        "Saved" ->
            findViewById<TextView>(R.id.failText).text = "Save flowers by tapping that star!"
        "Spot-A-Flower" -> {
            findViewById<ImageView>(R.id.failImage)
                .setImageResource(android.R.drawable.ic_menu_close_clear_cancel)
            findViewById<TextView>(R.id.failText).text = "Sorry, search failed"
        }
    }
```

Altho technically I can make an anonymous mode and save memory for the device, for one it is a lot of work, for two it discourages users from registering accounts. And I want them to make an account! I want everyone to use my precious app!

Searching Progress Bar

When testing the search page, I find that it takes a lot of time to connect to the database. I thought it has a bug or something, but it is just insanely slow. So I added a progress bar to the search page to make it obvious that the program is still running.

It would become visible at the start of the search page. Because the progress bar would not be a When searching, it becomes invisible after the neural network (haven't implemented). When a saved page or history page, it becomes invisible after the data is finished updating.

```
// if user not signed in and open saved or history, warn
if (mFirebaseAuth.currentUser == null && intent.getStringExtra( name: "Parent") != "Spot-A-Flower"){
    pageEmpty()
} else progressBar2.isVisible = true
```

It only becomes visible when the search page has a progress bar, that is, it must not go into the warning page.

```
<ProgressBar
    android:id="@+id/progressBar2"
    style="?android:attr/progressBarStyle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:visibility="invisible"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"/>
```

Tutorial

I have a new idea for it. Originally, I thought of making a slide show when the user opened the app for the first time.

But that format would not be able to display all of the information needed of the user. So I could make a github page about FAQs, put it in the app, so when the user has a question, they can open the link to the website and see the tutorial there!

That is certainly something huge to do, but I can do that after this semester. If there are people actually using my app, that is.

May 26, Tuesday

Minesweeper Idea

I know i should work on my project, but I have sudden insight about how to make an ai for the minesweeper game. Not because I was playing it for fun, it was for the glory of comp sci study that I indulged in the exploration of minesweeper.

It is obvious that the corners are mines, because the 1 at the diagonal has only one unknown block beside it. So it knows for sure that it is mine.

The same reasoning could go on forever with other scenarios, like 2 with only 2 unknown blocks, etc. After that mine is identified, the numbers surrounding it reduce by 1. Then it goes on.

S for safe, cross for mine.

But this method has an end:

This scenario can not be solved by the above method. However, intuitively, it can be solved by thinking.

The bottom 2 (actually 1 bc 1 of the mine is already discovered) has 2 unknown blocks around it, so each of them has 50% to be mine.

Then, the second bottom 2 (also actually 1) has 3 blocks around it, including 2 50% blocks. So the remaining unknown block can only be 0% mine, bc it is equal to 2 -1 - 2*0.5 = 0. So it is a safe block.

Likewise, the 3 label two 50% blocks around it. Thereby the 3 above it has a mine surrounding itself. Because the other surrounding block has probability of 3 -1 -2*0.5 = 1. So it is a mine.

Using this method, the probability of every block should be able to be determined, so the ai just clicks the block with highest probability, and recalculates it every step.

The question is, could this method always win the game?

No, because the 50% from the other origin can not share with the other pairs. Using the above method, the safe block turns to be 1,

If we look at the 1, it has 2 50% around it. So the other blocks around it should be safe, but they are not.

Same if we look at the 2, it has 1 mine and 2* 50% around it. That made me click the mine.

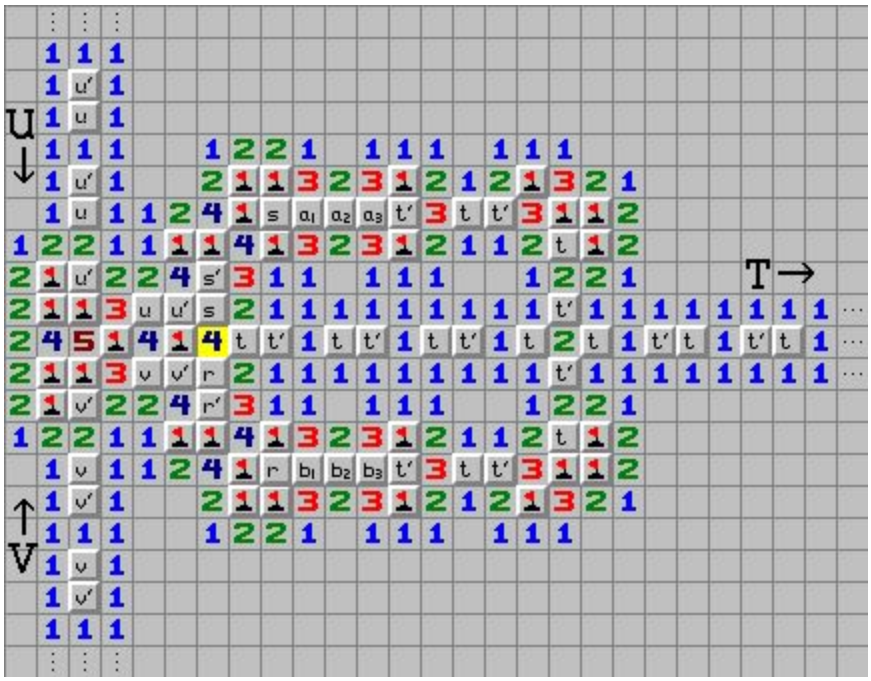
Seems like the possibility needs to be recalculated every step, and considering all of the combination of mine around the edge blocks. So it would find the correct

possibility considering all scenarios. It would just be brute forcing! That is a pretty complex and boring algorithm to make.

But even after that, there must be guesses to solve the game.

So I searched the problem online, and found this

http://web.mat.bham.ac.uk/R.W.Kaye/minesw/ordmsw.htm#about



It becomes a problem like this, where different letters represent a group, the status of the blocks are all connected to each other and we have to guess to find out what they are.

Minesweeper is proved to be NP-complete. No idea what that is but seems really important to the computer science field.

http://web.mat.bham.ac.uk/R.W.Kaye/minesw/ASE2003.pdf

It is too complicated.

I bail. I am out. No more minesweeper for me. I don't even want to play this game anymore.

BTW, I found out that this man thought of the same thing as I did, and actually made the air and reality.

Cool. Good for him. Also a little disappointed that I am not the first one that came up with this idea. But it is not like i am able to be the first one coming up with it anyway.

https://www.youtube.com/watch?v=cGUHehFGqBc

May 27, Wednesday

Iris Classification

```
/Users/aegerita/Coding-Projects/IrisML/ve
TensorFlow version: 2.1.0
Eager execution: True
hello world

Process finished with exit code 0
```

https://github.com/aegerita/IrisML

Because I know nothing of python or tensorflow, so i thought i should follow a codelab or a tutorial to see if I can handle it.

On the left is my first time using python, so excited!

It turns out very well with the iris dataset.

I did spend a lot of effort on it, however looking back at it, it didn't help my project that much, but it is very fun to do.

I have a lot of commenting in the codes (click the github to see the codes). So i will not put it here since it is pretty much irrelevant anyway.

May 28, Thursday

https://www.tensorflow.org/datasets/catalog/oxford_flowers102

https://www.tensorflow.org/datasets/overview

Straight up copied codes from this tutorial and load the dataset above.

https://www.researchgate.net/publication/323725215_An_efficient_classification_of_flower_images_with_convolutional_neural_networks

In that 2000 study and this 2018 study, the researchers achieved a 70+% accuracy… I wish I can do that today!

```
Epoch 3/6
32/32 [==============================] - 7s 216ms/step - loss: 4.6953 - accuracy: 0.0059 - val_loss: 4.7061 - val_accuracy: 0.0041
Epoch 4/6
32/32 [==============================] - 7s 216ms/step - loss: 4.6959 - accuracy: 0.0078 - val_loss: 4.7083 - val_accuracy: 0.0190
Epoch 5/6
32/32 [==============================] - 7s 231ms/step - loss: 4.6946 - accuracy: 0.0088 - val_loss: 4.8053 - val_accuracy: 0.0091
Epoch 6/6
32/32 [==============================] - 7s 224ms/step - loss: 4.6992 - accuracy: 0.0108 - val_loss: 4.7028 - val_accuracy: 0.0070
```

1% accuracy…

Just a setup to see if the it works

```
Epoch 20/20
32/32 [==============================] - 2s 61ms/step - loss: 1.0245 - accuracy: 0.6873 - val_loss: 6.4494 - val_accuracy: 0.0990
Test set accuracy: 10.686%
```

10.686% accuracy…

4 dense layers. I can see the epoch is too small.

```
Epoch 5/5
193/193 [==============================] - 52s 270ms/step - loss: 0.7908 - accuracy: 0.7660 - val_loss: 3.3840 - val_accuracy: 0.2873
Test set accuracy: 30.882%
```

30%!!!!!!

Seeing the accuracy is too small, searched online to see that image classification should use CNN to achieve best accuracy!

But it runs much too slowly. (me at this time have no idea what i am dealing with)

```
Epoch 30/30
97/97 [==============================] - 4s 38ms/step - loss: 1.7670 - accuracy: 0.4967 - val_loss: 2.2361 - val_accuracy: 0.4078
Test set accuracy: 43.137%
```

45%!!

Finally! 2 hours later!

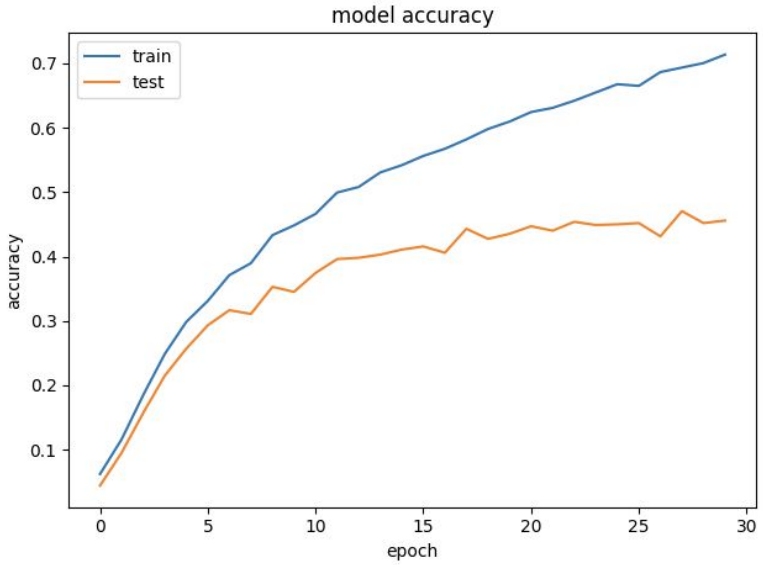
Reduced the image size from 64*64 to 32*32 to make it run faster. The accuracy doesn't seem to change that much, weird.

Added drop out to regulate the overfitting issue.

Added another layer which seems to improve the performance.

Honestly I have no idea what works or not. I am just keeping changing the number, waiting for 10 min and hoping it does something different.

It is called insanity.



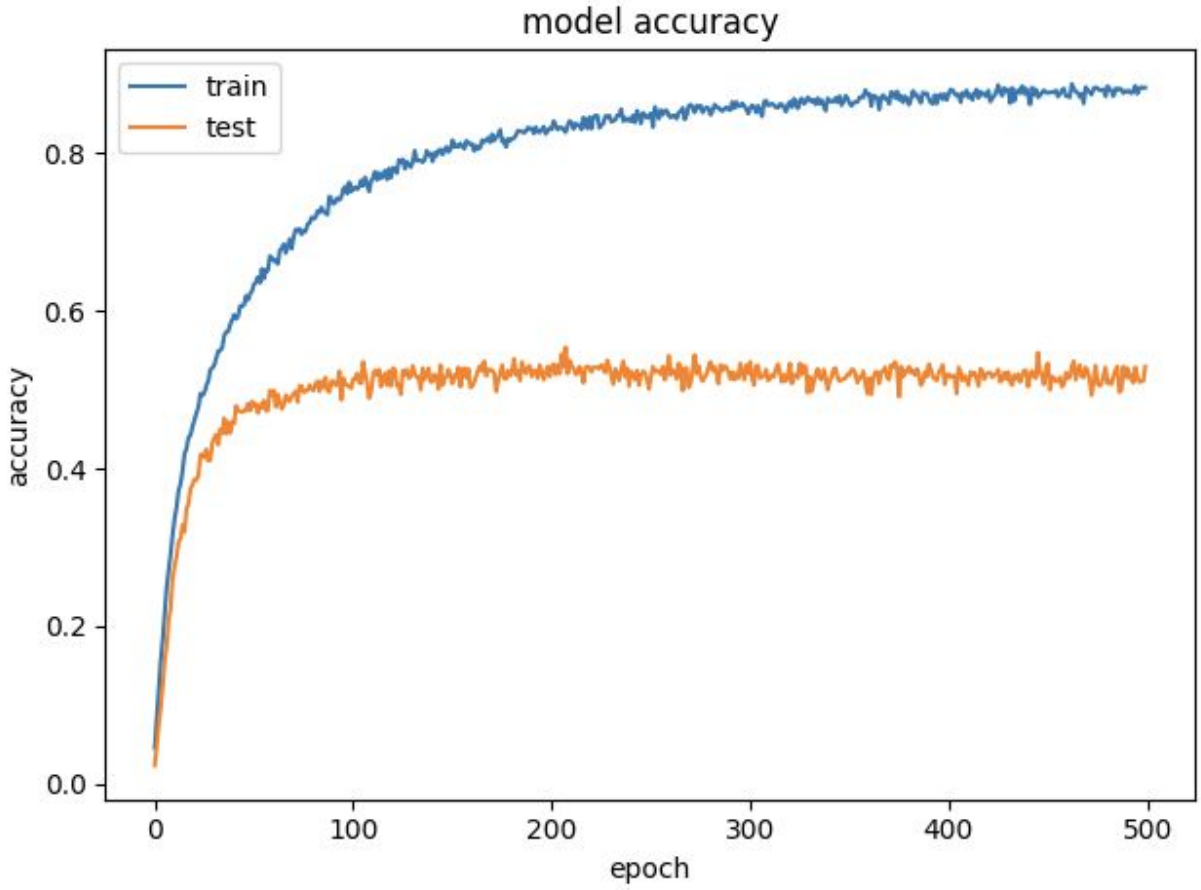
50%!

3 hours later…

Deleted the layer I just added… no not know what effect it has

Added the weight regularizer(L2). Seems to increase the accuracy!

Because I need to go to bed, it is freaking 5 am!!! I set the epoch to 500 so i have something to look forward to in the morning (actually afternoon).



May 29, Friday

56%

Above is the result plot. Should I be happy with a 50? No. But am i proud of my work? Actually no nevermind.

Epoch does not help with accuracy.

So next I try to increase the dataset. I added the testing dataset to the training dataset. It is weird because this dataset has train, test, and validation 3 split, so i guess i should make the most of it to increase my sample size.

It gets to 56%! Vala!

And now I am stuck at 56 for 3 hours.

Tried decreasing dropout, didn't work

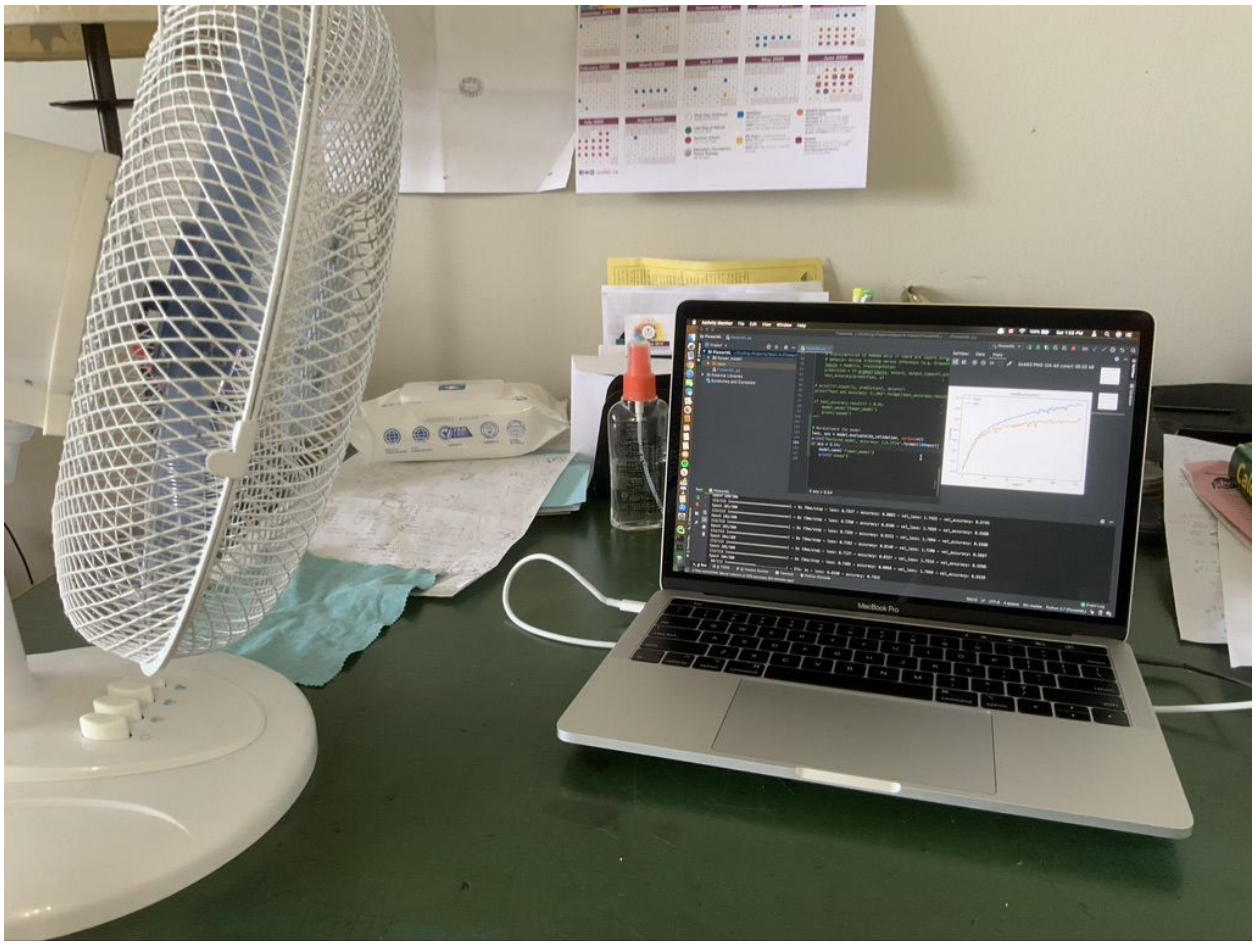
Tried to add more samples, increase to 57, not worth it.

Tried to increase image size, didn't work

Tried more regularizer, didn't work

Tried an extra layer, didn't work

BTW, my laptop keeps getting hotter and hotter, so i used THIS method to make it cooler:



Altho it made it harder to work with (because it is so cold), it is worth it to protect my pretty macbook!

And Finally!!!!!!!!!!!!!!!!!!!!!

60%!!!!!

```
60/60 [==============================] - 8s 133ms/step - loss: 0.9071 - accuracy: 0.7761 - val_loss: 1.5824 - val_accuracy: 0.6000
4/4 - 0s - loss: 1.5824 - accuracy: 0.6000
Restored model, accuracy: 60.00%
```

By increasing the batch size to 128!

Now i try to increase to 256 to see if it goes even higher!

No it went down to 57%..

I am happy enough with 60% tbh… I haven't had the gall to learn all about image augmentation and segmentation so i am fine with what it is right now. It is the best I can do (said after at least 15 hours devoted to change random numbers)

I can do that after I finish the project!

Useful Links

Here are about 1/6 of the resources i used:

https://www.tensorflow.org/tutorials/keras/overfit_and_underfit#strategies_to_prevent_overfitting

https://www.tensorflow.org/tutorials/customization/custom_training_walkthrough#use_the_trained_model_to_make_predictions

https://www.tensorflow.org/datasets/overview

https://towardsdatascience.com/overfitting-vs-underfitting-a-complete-example-d05dd7e19765

https://datascience.stackexchange.com/questions/18414/are-there-any-rules-for-choosing-the-size-of-a-mini-batch

https://machinelearningmastery.com/improve-deep-learning-performance/

And if i want to do the augmentation later on:

https://www.tensorflow.org/tutorials/images/data_augmentation

https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/

(tried it, failed miserably)

Pytorch

https://www.kaggle.com/c/oxford-102-flower-pytorch/notebooks

Found this website, it holded a contest for this flower dataset! A lot of them scored above 95%!!!! I am so jealous!!!! But I can't find their codes.

But eventually I found one person with her tutorial:
https://www.freecodecamp.org/news/how-to-build-the-best-image-classifier-3c72010b3d55/

I don't know if I will be able to follow the codes and make the same model. They all use gpu, and I know nothing of that method. So I guess I will have to learn that after the project is finished.

I am just too much of an amateur at this. If I had more time I may be able to start over with a normal learning process, and improve gradually. BUT that was Jerry's job.

WAIT, nevermind. The pytorch model cannot be turned into tensorflow lite to be used in my app!

If i do that I will have to follow this tutorial
https://towardsdatascience.com/converting-a-simple-deep-learning-model-from-pytorch-to-tensorflow-b6b353351f5d and use onnx.

Depressing.

May 30, Saturday

Use Tensorflow lite in the App

https://firebase.google.com/docs/ml-kit/android/use-custom-models

Now i need to follow this to use the neural network in my android app.

After a bunch of manifests were set up, I tried to load the tensorflow model in the search activity, but then I realized that the camera image as a bitmap would be difficult to pass through activities. So I did the neural network in the main activity in the search function.

```
// connect to neural network
private fun searchFlower(imageBitmap: Bitmap) {
    val intent = Intent( packageContext: this, FlowerSearch::class.java)
    intent.putExtra( name: "Parent", "Spot-A-Flower")

    // get image from the main activity
    val imageSize = 44
    val bitmap : Bitmap! = Bitmap.createScaledBitmap(imageBitmap, imageSize, imageSize, filter: true)
```

Here I have to resize the bitmap to be the input format of the model. Which is 44*44 (which is small but that is all that my cpu can handle).

```
// load model
val localModel : FirebaseCustomLocalModel = FirebaseCustomLocalModel.Builder()
    .setAssetFilePath("flower_model.tflite").build()
val options : FirebaseModelInterpreterOptions = FirebaseModelInterpreterOptions.Builder(localModel).build()
val interpreter : FirebaseModelInterpreter? = FirebaseModelInterpreter.getInstance(options)
```

I created an assets folder and copied the model to the project, so the main activity can access and make an interpreter by using the absolute file path.

```
// set input output format
val inputOutputOptions : FirebaseModelInputOutputOptions = FirebaseModelInputOutputOptions.Builder()
    .setInputFormat(0, FirebaseModelDataType.FLOAT32, intArrayOf(1, imageSize, imageSize, 3))
    .setOutputFormat(0, FirebaseModelDataType.FLOAT32, intArrayOf(1, 102)).build()
```

Apparently this step is necessary. Here is a bug that bugged me for an hour: the model I trained has an output of 112 instead of 102 flowers because I forgot about that and entered the wrong number to the tensorflow. I had to actually retrain the model because of that…

But besides that it works fine, gladly.

Followed the tutorial, where it said to reformat the input image. Cool

```
// set input to proper format
val batchNum = 0
val input : Array<Array<Array<FloatArray>>> = Array( size: 1) {Array(imageSize){Array(imageSize){FloatArray( size: 3)}}}
for (x : Int in 0 until imageSize-1) {
    for (y : Int in 0 until imageSize-1) {
        val pixel : Int = bitmap.getPixel(x, y)
        // Normalize channel values to [-1.0, 1.0]. This requirement varies by
        // model. For example, some models might require values to be normalized
        // to the range [0.0, 1.0] instead.
        input[batchNum][x][y][0] = (Color.red(pixel) - 127) / 255.0f
        input[batchNum][x][y][1] = (Color.green(pixel) - 127) / 255.0f
        input[batchNum][x][y][2] = (Color.blue(pixel) - 127) / 255.0f
    }
}
```

And after that this is where the interesting thing happens.

```
// pass in the input
val inputs : FirebaseModelInputs = FirebaseModelInputs.Builder().add(input).build()
interpreter!!.run(inputs, inputOutputOptions)
    .addOnSuccessListener { result ->
        val output : Array<FloatArray> = result.getOutput<Array<FloatArray>>(0)
        val probabilities : FloatArray = output[0]
        val reader = BufferedReader(
            InputStreamReader(assets.open( fileName: "flower_labels.txt"))
        )
        val myDataset: MutableList<Flower> = ArrayList()
        for (i : Int in probabilities.indices) {
            val label : String! = reader.readLine()
            reader.readLine()
            myDataset.add(Flower(label, (probabilities[i]*100).toInt()))
            Log.i( tag: "MLKit",  msg: label+": "+(probabilities[i]*100).toInt())
        }
        myDataset.sortDescending()
        intent.putExtra( name: "flower1_name", myDataset[0].name)
        intent.putExtra( name: "flower2_name", myDataset[1].name)
        intent.putExtra( name: "flower3_name", myDataset[2].name)
        intent.putExtra( name: "flower1_detail", myDataset[0].detail)
        intent.putExtra( name: "flower2_detail", myDataset[1].detail)
        intent.putExtra( name: "flower3_detail", myDataset[2].detail)

        startActivity(intent)
    }
```

The interpreter just gives out the result in an array. Because I only have one input, I take the first array and name it to probabilities. It lists out the probability for each flower, in order.

As a result, I have to link the probability to the name of the flower. Normally people might set up a map for that, but since I also need it for the database, I just load the text file of flower names to the asset folder, and let the programm read it line by line.

I made a flower list, and in the loop, I added each of the corresponding flower names and their probability as a flower to the list, so that it includes the name and probability together.

Then I sort the flower by probability, and send the top 3 to the search activity. The reason I don't send the list directly is because android can't do that easily. And I just need 3 of them anyway. So I just put intent.

To do that, I modified the flower object to inherent Comparable.

Flower Object

```
class Flower : Comparable<Flower> {
    var name: String
    var detail: String
    var int: Int

    constructor(name: String, time: Long) {
        this.name = name
        val sdf = SimpleDateFormat( pattern: "hh:mm:ss MM/dd", Locale.CANADA)
        this.detail = sdf.format(Date(time))
        this.int = 0
    }

    constructor(name: String, detail: String) {
        this.name = name
        this.detail = detail
        this.int = 0
    }

    constructor(name: String, probability: Int) {
        this.name = name
        this.detail = "$probability% Probability"
        this.int = probability
    }

    override fun compareTo(other: Flower): Int {
        return int - other.int
    }
```

To compare it, the probability must be saved as an integer (actually I can get it from string manipulation of the detail string, but why would I do that when it is right there). So I added the int parameter, inherent the comparable, and made the compareTo function.

Now the main activity can sort the flower object.

After Neural Network

If the neural network didn't work, the failure listener sent an error message to the user.

```
.addOnFailureListener { e ->
    Toast.makeText(
        context: this,
        text: "There is something wrong with the AI, mind trying again?",
        Toast.LENGTH_SHORT
    ).show()
    Log.w( tag: "TAG",  msg: "loadData:onCancelled", e)
}
```

And if it works, the activity opens intent to the search page.

```
"Spot-A-Flower" -> {
    myDataset.add(Flower(intent.getStringExtra( name: "flower1_name"), intent.getStringExtra( name: "flower1_detail")))
    myDataset.add(Flower(intent.getStringExtra( name: "flower2_name"), intent.getStringExtra( name: "flower2_detail")))
    myDataset.add(Flower(intent.getStringExtra( name: "flower3_name"), intent.getStringExtra( name: "flower3_detail")))

    // save the flower to history when search if the user choose so
    val sharedPreferences :SharedPreferences! = PreferenceManager.getDefaultSharedPreferences( context: this)
    if (sharedPreferences.getString( key: "addHistoryWhen",  defValue: "search") == "search") {
        mFirebaseAuth.currentUser?.uid?.let { it: String
            for (flower :Flower  in myDataset){
                database.child( pathString: "users").child(it).child( pathString: "history")
                    .child(flower.name).setValue(System.currentTimeMillis())
            }
        }
    }
    progressBar2.isVisible = false
}
```

The program turns the passed in string back as a flower object, and adds them to the dataset for the recycler view. And all of these flowers are saved in the for loop.

Wait, on second thought, only the flower with the most probability should be saved, not 3 of them! I need to change this!

Hand Testing Neural Network

And now I can test it! So excited.

Well it turns out really bad. I passed an image of a sunflower, and it tells me it is 39% Gazania! I mean they do really look alike, but still!

After linking the neural network to the app, i know how bad it is!! It never gives the correct result and it just breaks my heart…

It is really really really really really hard for me

;(

Everything else is working perfectly, which means that if I have a better model (which i will be able to make in the future!) I can just update the tensorflow lite file and it will work perfectly.

No pic because I am writing this journal one day later (it was ***6 AM*** and i really needed to sleep), and there is another major thing happening.

May 31, Sunday

Prepopulate Database

While some of this happened yesterday (did this while waiting, retraining and messing up the neural network), I put it here because it is about databases and I better put them together.

Prepopulating the database has always been a torturous thing to think about. Technically I can link a .db file with my app using https://github.com/jgilfelt/android-sqlite-asset-helper, but first I need to have a .db file. That requires me doing SQL, but I don't want to. I really don't want to. I think I really don't like that kind of stuff.

Then I thought, hmmmmmm, I already have a text file stacking all of the names of the flowers, why don't I just put other stuff too in the text file? It is super super easy! I know it is probably not safe, anyone can change that, but why the hell would any user change that anyway.

I can put the description as txt, but what about the icon?

Picking Images for Icon

Luckily I have images from the training set! I can just hand pick one picture from the training set, and load them to the asset folder!

Great Idea!!!!

But hand picking images is more work than I thought. After doing like the first five of them, I wrote a program to help me.

See the pic below, in the dataset, each folder has number 1-102 representing different flowers. And the image has a random number.

So after picking them, I put the random number to an array (as a string because some of them have 0 as the first number). The program go through them, cut them to 100*100 size, name them 1.jpg to 102.jpg, and save them in the assets folder.

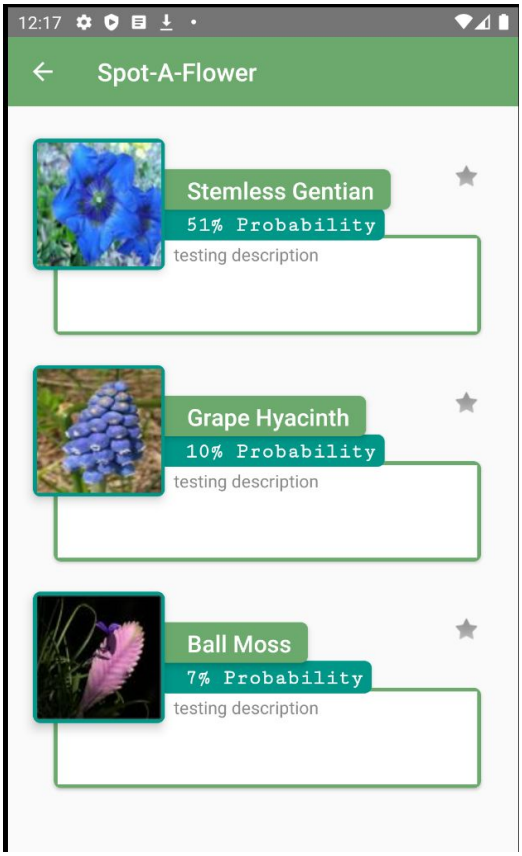
Very convenient indeed. Took me 1 hour to go through the photo though. It requires great art to pick the best flower!

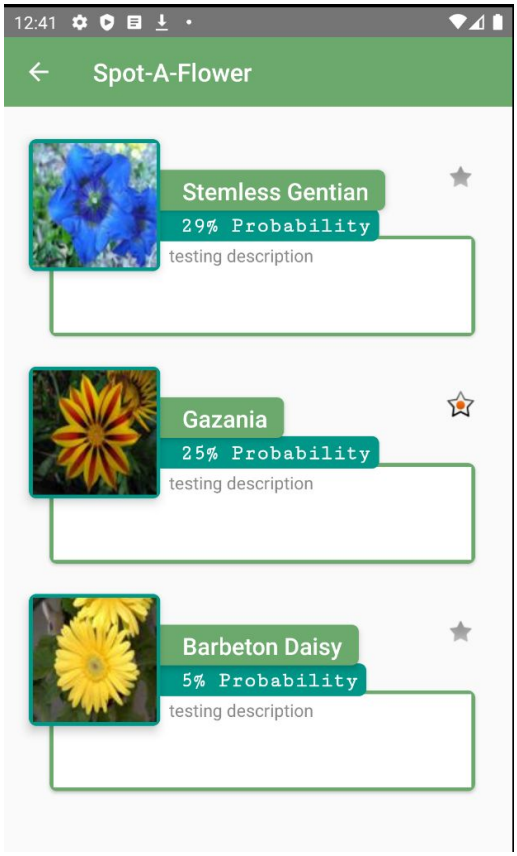
Then, i put the them to the assets folder, and wrote this to initialize the database

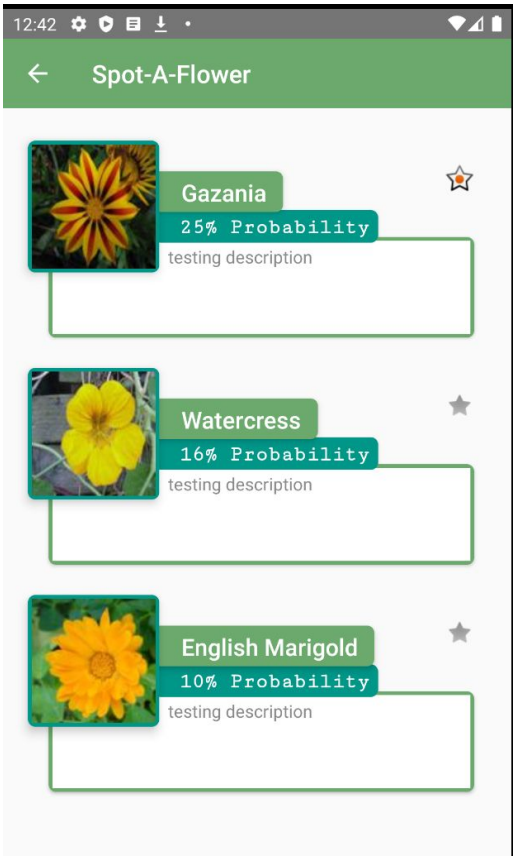
Initialize in onCreate

```
override fun onCreate(db: SQLiteDatabase) {
    val CREATE_TABLE :String  = "CREATE TABLE $TABLE_NAME " +
            "($ID Integer PRIMARY KEY, $NAME TEXT, $INTRO TEXT, $ICON BLOB)"
    db.execSQL(CREATE_TABLE)
    val reader = BufferedReader(
        InputStreamReader(context.assets.open( fileName: "flower_labels.txt"))
    )
    for (i :Int  in 1..102) {
        val name :String!  = reader.readLine()
        val description :String!  = reader.readLine()
        val inputStream :InputStream  = context.assets.open( fileName: "icons/$i.jpg")
        val bitmap :Bitmap!  = BitmapFactory.decodeStream(inputStream)
        addFlower(name, description, bitmap, db)
    }
    printAllFlowers(db)
}
```

There originally is a bug. Because addflower calls back on the onCreate method to open the database, so it gets called recursively. Now I made the database a parameter, so that the addFlower doesn't need to call onCreate anymore.







Then it loads the flowers. Like this

LOOk they are so beautiful!

The description now is just "testing description" but I will soon add a description for each flower!

Just glad to see it working!

TODO list

I really can't believe this is the last journal!!!!!

It is too soon to end! I have a lot more things to do! So I will make a TODO list!

Immediate

- Add flower descriptions
- Add wiki links also and add the link column to the database
    - Because some of the links are not working in wikipedia, have to add by hand
- Reread and comment on all codes
- Reread all of the journals
    - to check for things left out
- Do all functions to check bugs
- Put all journals in an ultimate document!
    - Easier to read the progress
    - Save it personal account cause school email is going away

Should Do

- Make a tutorial
    - Slider show at the beginning
    - Show up tips arrows and explanations
    - Github README and app "about" for more informations
- Remake the Neural Network
    - To 90%!!!!
    - Well maybe 70% is a more reasonable goal
- Add a search function
    - Able to list all 102 flowers using search page
    - Sliding bar searching by alphabet
- Delete selected history

    - Make a little trash can button
- Upload the app to Google Play!!!!!!!
    - I totally should do this!
    - Add advertising and stuff, maybe i will earn some money! Right?

In a Lifetime

- Make a proper SQL database
    - No more text file, use asset helper instead
- Add more login method
    - email/password
- Add sound effect
    - It already have some default android effects
- Draw the camera button
    - The current one looks bad
- Proper anonymous mode

Other Stuff

I can not believe I can find this much to improve! Well, I will just keep working.

Funny to think the whole thing used to be Jerry's idea, hell it even isn't my idea!! Now I spend so much effort and love on it!

I am really thankful that I did this!

I will definitely make sure to brag about it on my resume