Handover Documentation

Team M

Table of Contents

Introduction 3

Repository 3

  Link 3

Running Locally 3

  Docker Containers 3

  Android Drawing Application 3

  Admin Portal 3

Packaging 4

  Android Drawing Application 4

  Admin Portal 6

  API 6

Hosting 6

Configuration 7

  API 7

    Configuring Database Connection 7

    Configuring Backblaze 7

  Admin Portal 8

  Android Drawing Application 8

Introduction

This handover documentation aims to provide some basic information about the hosting and configuration of the project. For additional information and documentation see the GitHub repository.

Repository

Once the project is finalised on our part, you will need to contact the COMP2003 module leader and request access/ownership of the GitHub repository.

All of the files which are relevant to the project are included in the GitHub repository. The documentation can be found in the "Project Docs" folder and the source code for each project is located in the "Source Code" folder.

Module Leader: **Ji-Jian Chin**

Email: **ji-jian.chin@plymouth.ac.uk**

Link

https://github.com/Plymouth-University/comp2003_2022-team-m

Running Locally

Docker Containers

To run the program locally, you can use Docker. Clone the repository, open up a terminal window and navigate to the "Source Code" folder. Once there, run the following command: "docker-compose build". Wait for the images to build and then run "docker-compose up".

Depending on what you intend to work on, you may not need to run all containers at once. You can choose which ones to run by specifying their names when building and running them like so: "docker-compose build api sql_db" and "docker-compose up api sql_db". This will only build and run the API and SQL database containers. The names of the services are defined in the "dockercompose.yml" file.

This will create three contains — one for the SQL database, one for the API and one for the web drawing application.

Android Drawing Application

To run the Android drawing application, open the "Android" folder in Android Studio and simply run the application by clicking on the green triangle button.

Admin Portal

To run the admin portal, navigate to the "Admin" folder within the source code. First, run the "npm install" command to download the dependencies. Once finished, run "npm run start".

Packaging

Android Drawing Application

To package the Android drawing application, open the “Android” folder in Android Studio. However over “Build” and click on “Generate Signed Bundle / APK” in the taskbar. *(see Figure 1)*

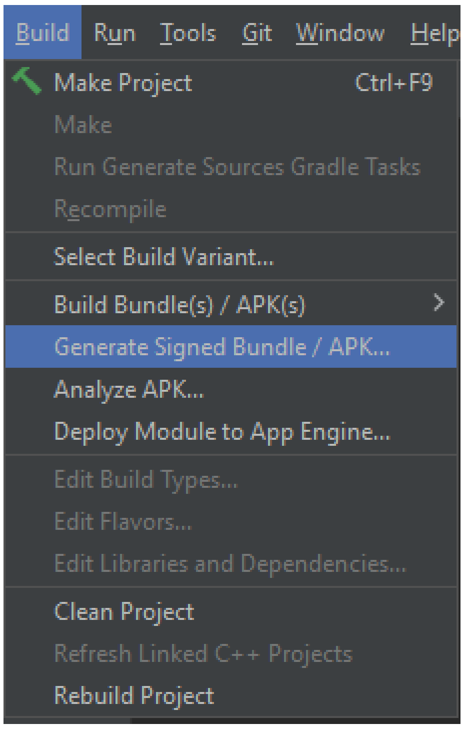
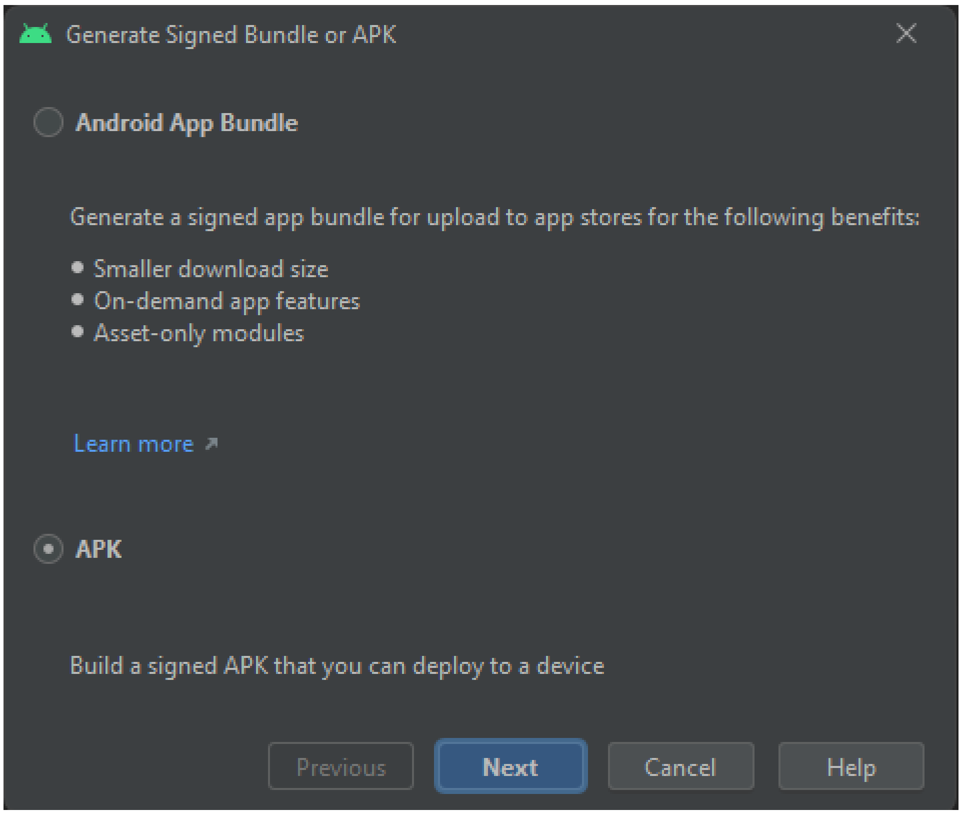


Figure 1 – Build context menu in Android Studio

Once clicked a popup will appear *(see Figure 2)*.


Generate Signed Bundle or APK

( ) **Android App Bundle**

Generate a signed app bundle for upload to app stores for the following benefits:

- Smaller download size
- On-demand app features
- Asset-only modules

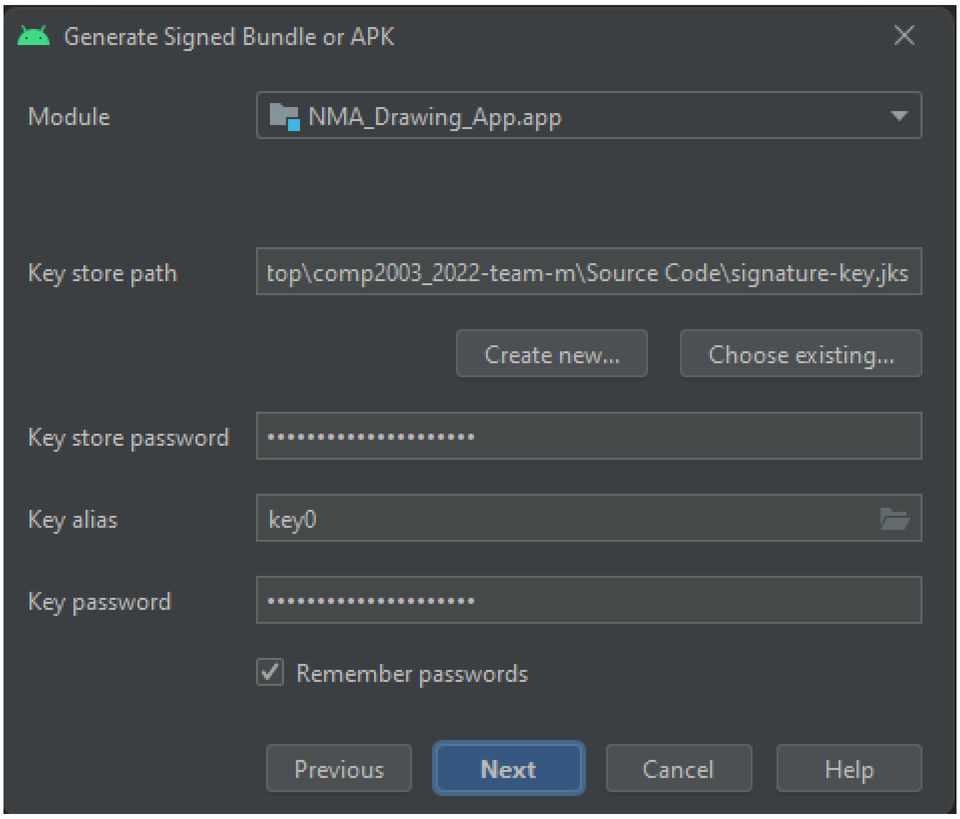
Learn more ↗

(●) **APK**

Build a signed APK that you can deploy to a device

Figure 2 – Generate Signed APK popup window

Select “APK” and click “Next”. In the following screen, click on “Create new...” to create a new signature for your application *(see Figure 3)*.

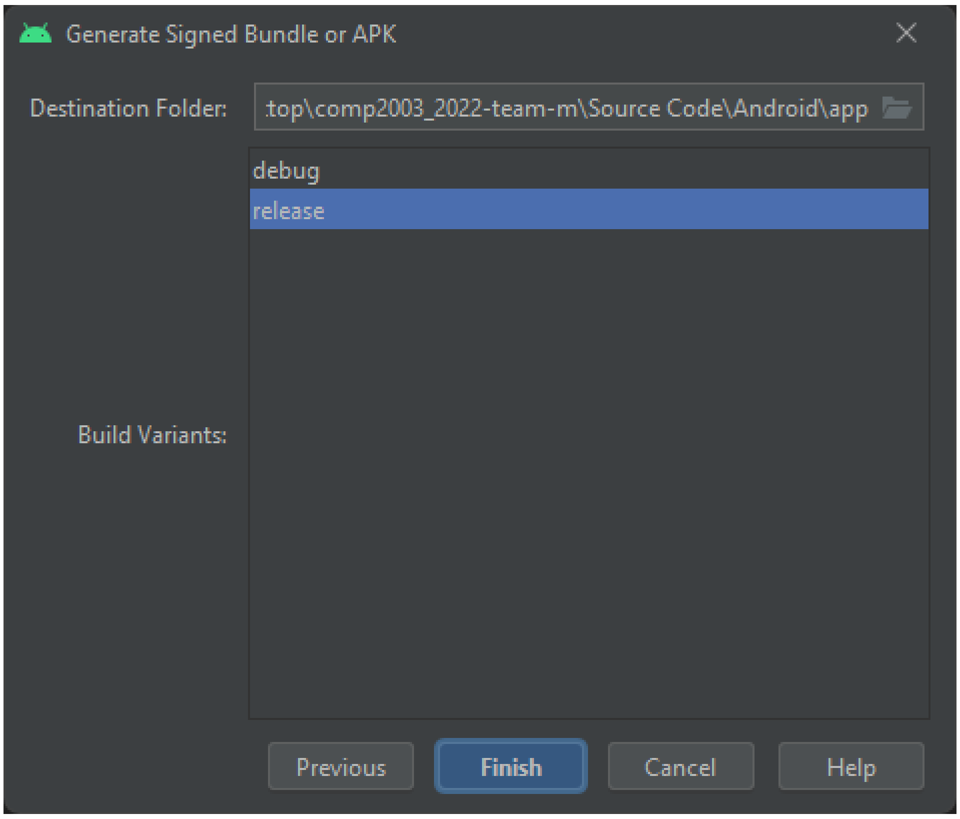

Generate Signed Bundle or APK

Module	NMA_Drawing_App.app
Key store path	top\comp2003_2022-team-m\Source Code\signature-key.jks
Key store password	••••••••••••••••••••
Key alias	key0
Key password	••••••••••••••••••••

[✓] Remember passwords

Figure 3 – Generate Signed APK popup window, signature section

Then fill out the required information and click "OK". You will be navigated back to the previous screen *(see Figure 3)*. Click "Next" where you will be prompted to select the directory where your APK will be stored *(see Figure 4)*.



Destination Folder: top\comp2003_2022-team-m\Source Code\Android\app

Build Variants: debug, release

Figure 4 – Generate Signed APK popup window, folder section

Select your preferred folder and click "Finish". Once done, you will find your ".APK" file in the specified directory which you can install on your Android devices and start using.

Note: before packaging the application make sure to update the API URL with your hosted API's URL.

Note: make sure to keep the key which has been created during the process secure, it should not be stored in the Git repository.

Admin Portal

The admin portal can be packaged by running the "npm run build" command while in the "Admin" directory. Once finished the application's setup will appear in the "release" folder.

API

The API is automatically built and packaged when its containerised.

Hosting

Docker images/containers have been configured for the API and the database, therefore, those can easily be hosted on a platform of your choosing, providing it supports Docker.

Note: After hosting the API, make sure to change the current API URL in both client applications — the Android drawing app and the admin portal. (see Configuration section)

Configuration

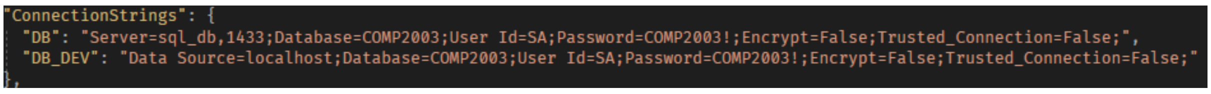
API

To make sure the API is working as intended you will need to configure a couple of things:

1. The database connection string
2. Backblaze secret/bucket keys

Configuring Database Connection

Once your database is hosted you need to add the connection string to the appsettings.json found in the root directory of the API project. *(see Figure 5)*



```
"ConnectionStrings": {
  "DB": "Server=sql_db,1433;Database=COMP2003;User Id=SA;Password=COMP2003!;Encrypt=False;Trusted_Connection=False;",
  "DB_DEV": "Data Source=localhost;Database=COMP2003;User Id=SA;Password=COMP2003!;Encrypt=False;Trusted_Connection=False;"
},
```

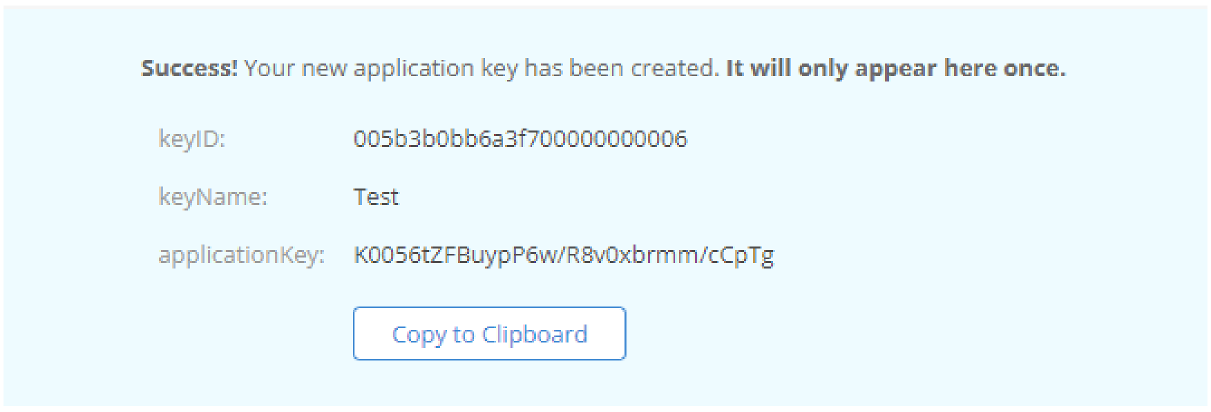
Figure 5, appsettings.json Connection Strings section

Configuring Backblaze

The drawings produced during events are stored using a third-party cloud provider — Backblaze. To configure Backblaze within the API you will need to create a Backblaze account and provide your own API and bucket keys.

Instructions

1. Create an account.
2. Navigate to "My Account" page.
3. Navigate to "App Keys" section.
4. Click on the "Add a New Application Key" button and create a new key.
5. You will then be shown the Key ID and the Application Key. *(see Figure 6)*

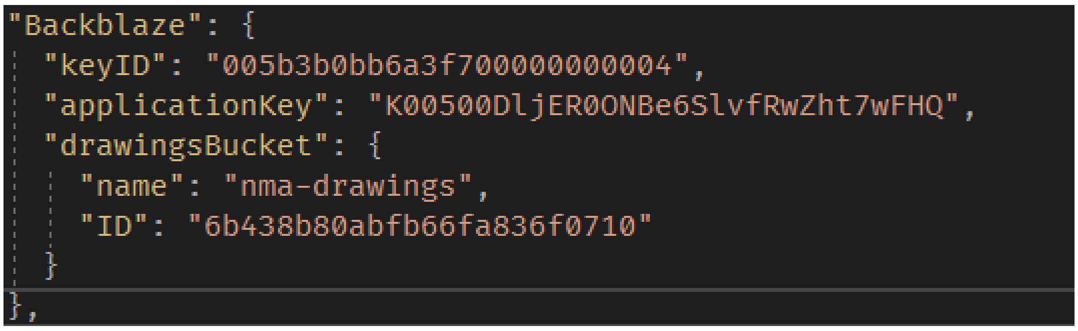


Success! Your new application key has been created. **It will only appear here once.**

keyID:	005b3b0bb6a3f700000000006
keyName:	Test
applicationKey:	K0056tZFBuypP6w/R8v0xbrmm/cCpTg

Figure 6 – Popup after API key creation in Backblaze account settings

6. Copy the "keyID" and "applicationKey" values from the success dialog and paste them into the appsettings.json file in the API. *(see Figure 7)*



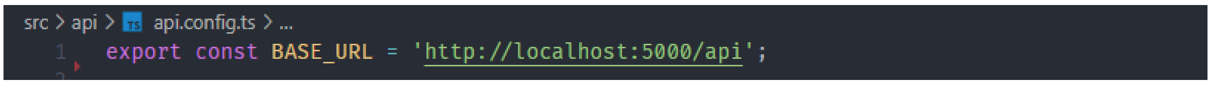
```json
"Backblaze": {
  "keyID": "005b3b0bb6a3f700000000004",
  "applicationKey": "K00500DljER0ONBe6SlvfRwZht7wFHQ",
  "drawingsBucket": {
    "name": "nma-drawings",
    "ID": "6b438b80abfb66fa836f0710"
  }
},
```

Figure 7 – appsettings.json Backblaze section

7. Then while still in account settings, head to “Buckets”.
8. Click “Create a Bucket”.
9. Once created, copy the bucket name and the “bucket ID” values and paste them into the appsettings.json file under the “drawingsBucket” property. *(see Figure 7)*
10. Build the application.
11. All done, the API should be able to communicate with your Bacbklaze service now and read and write drawing files.

Admin Portal

Once the API is hosted you will need to update the API URL within the admin portal application’s API configuration file. It can be found by navigating to *“Source Code/Admin/src/api/api.config.ts”.* You simply need to change the BASE_URL variable to your hosted web service’s URL. *(see Figure 8)*



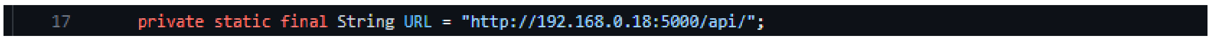
```
src > api > api.config.ts > ...
export const BASE_URL = 'http://localhost:5000/api';
```

Figure 8 – Screenshot of BASE_URL constant variable in api.config.ts

Android Drawing Application

Similarly, to the admin portal application, you need to update the API’s base URL in the RetrofitClient.java file. It can be found by navigating to *“Source Code/Android/app/src/main/java/com/example/nmadrawingapp/model/data_sources/network/RetrofitClient.java”.*

You just need to update the URL variable to your web service’s base URL. *(see Figure 9)*



```java
    private static final String URL = "http://192.168.0.18:5000/api/";
```

Figure 9 – Screenshot of URL constant variable in RetrofitClient.java