Appendix F Installation guide

F.1 Prerequisites

There are a few prerequisites for building the app. It's assumed you have these before proceeding. Note that the exact prerequisites depends on which components you'd like to build.

General prerequisites:

1. Git

Prerequisites for building the web app:

1. Node.js 18, 20, or 21
2. If deploying the server (or simulating a deployment scenario): Docker

Prerequisites for building the flutter app:

1. You will need Windows, Linux or ChromeOS to run the app on an Android mobile device, and macOS to run on an iOS mobile device.
2. Flutter 3.13 or higher. Please select your OS system, and follow the official Flutter Installation Guide.
3. Ensure you have added flutter to your path environment variable (This is also described in the installation guide)
4. Ensure you have all the necessary dependencies for your specific operating system by running 'flutter doctor' in your terminal. If someone is missing, you need to install them. (This is also described in the previously linked Flutter installation guide)

Note: It is not required to build the Android app from source, but it can be done, and the steps for it are mentioned. If you're using iOS, you're stuck building yourself, as Apple doesn't allow side-loading without jail-breaking or using other complicated methods (Costello, 2022), which is unreasonable to require of the average user.

F.2 Cloning the repository

Before anything else can be done, the repository has to be cloned. This is done with

```
# This can also be done with an SSH link
git clone https://github.com/simeeid/TDT4290Group1/
cd TDT4290Group1
```

After this, you can go to one of the next subsections to learn how to compile each component.

F.3 Pre-deployed web app

We've deployed the web app to https://io-t-dashboard-tau.vercel.app/, where the web app. To log in, you can use the credentials found in Appendix H.3.4.

F.4 Building and deploying the React app

Note: If you're planning to use Docker, you can skip straight to F.4.4

Also note that the web app installation works identically on all operating systems, though the process of installing prerequisites varies depending on your operating system.

F.4.1 Installing npm dependencies

```
# cd can be skipped if you opened norbit-webapp directly
# instead of the git root
cd norbit-webapp
npm i
```

npm will then install all the required dependencies.

F.4.2 Setting up AWS

Setting up AWS with the web app is fairly straight-forward. Unless you're planning to set up AWS from scratch, you can copy the credentials from Appendix H.3.1.2 into <project root>/norbit−webapp/.env.

F.4.3 Building the web app

When AWS has been properly configured, you can build the app with

```
npm run build
npm run start
```

This will build and run a production version of the webserver. Alternatively, npm run dev can be used to start a development server. To learn how to use the web app, continue to Appendix G.

F.4.4 Deploying the React app with Docker

Before diving into this section, it's worth emphasising that using Docker is primarily recommended for deployment purposes. Using Docker for development is possible, but it comes with many disadvantages. Particularly, even with a volume mount, at least when using a Windows host, mounted volumes do not pass file update information into Docker, which prevents npm run dev from registering updates as expected ('Docker Compose not synhronising file changes in volume', 2019). We have not verified if this behaviour is different on other operating systems.

Note that this step is not required if you're setting up a local build and running it on your machine. This step is only required if you want to run the server in Docker.

To deploy the React app, it's recommended to use Docker. There's a Dockerfile set up in the repository that can be used that automatically takes care of all dependencies. However, before running docker, the steps mentioned in Section F.4.2 need to be performed. This is required to configure the AWS connection.

Running docker is done with:

```
cd norbit-webapp
docker build -t norbit .
# This is an example that also automatically removes the container
# after use.
```

```
# Different deployment scenarios have different exact run commands,
# but this is left as an exercise to the reader.
docker run -p 3000:3000 --rm -ti norbit
```

Running a dev-server instead (though this makes no practical difference aside running a different command to start the server) can be done by passing ––build–arg BUILD_MODE=dev to docker build. However, it's strongly recommended to leave the default alone.

F.4.5 Running tests

F.4.5.1 Unit and component tests

Unit and component tests are run separately, but neither test categories requires any additional setup, as they are fully self-contained. Simply run

```
# To run component tests:
npm run component-test
# To run unit tests:
npm run unit-test
```

F.4.5.2 Integration tests

Integration tests are more involved, as they run on a live version of the server. This means there's setup required prior to actually running the tests.

First of all, if you have an instance of the server already running, stop it. Then rerun it with:

```
# MacOS/Linux:
NEXT_PUBLIC_MOCK_AMPLIFY=yes npm run dev
# Windows with Powershell:
$Env:NEXT_PUBLIC_MOCK_AMPLIFY = 'yes'
npm run dev
```

After doing this, open a new terminal to run the tests, and run them with:

```
npm run component-test
```

F.5 Building and installing the Flutter app

Due to the reasons mentioned in Section 12.6, the instructions have not been validated for the iOS build. The instructions have been validated on Windows 10 and 11, as well as Linux Mint 21.2 (an Ubuntu derivative).

F.5.1 Android

We recommend using pre-built Android binaries for running the flutter app on your Android device. This is explained in F.7.

Note that this step installs the app to your device. When it has been installed, you can run it by locating flutter_application_1 in your device's app menu, and clicking it. To learn how to use the app, continue to Appendix G.

If you wish to run the application from Android Studio you can follow this guide:

1. Open the cloned project repository in Android Studio
2. Follow this guide to connect your android device: Run apps on a hardware device.
3. Continue to step F.5.3.

F.5.2 iOS

We recommend following this guide to get the app on your iPhone: How to install Flutter App on iOS.

Note that this step installs the app to your device. When it has been installed, you can run it by locating flutter_application_1 in your device's app menu, and clicking it. To learn how to use the app, continue to Appendix G.

If you wish to run the app from Android Studio you can follow this guide (after getting the initial app on your iPhone, as explained above):

1. Open the cloned project repository in Android Studio
2. Follow this guide to connect your iPhone device: Run apps on a hardware device.
3. Continue to step F.5.3.

F.5.3 Running the app

Type the following in your Android Studio terminal:

```
cd norbit-mobileapp
cd flutter_application_1
flutter pub get
flutter run
```

F.6 Running the tests

Open the cloned project repository in Android Studio, and type these commands in your terminal:

```
cd norbit-mobileapp
cd flutter_application_1
flutter test
```

F.7 Installing pre-built Android binaries of the Flutter app

If you prefer not to build the app, or just find it more convenient to use a pre-built binary, GitHub Actions has been configured to automatically build APKs for releases. Releases can be found at https://github.com/simeeid/TDT4290Group1/releases. Simply download the APK file from the latest release. Then, click the APK after it's finished downloading, and you'll be prompted with an install. You may need to trust app installs from your browser before doing so, if you haven't already. You'll be automatically prompted by Android if you haven't allowed it.

You may also receive a warning about the developer being unverified. This is a consequence of skipping the signing process, as we haven't set up a signing system. Look for a button called "Install anyway". It might be in a sub-menu labelled "More details". The exact details, such as whether or not there is a sub-menu, and what the sub-menu is labelled as, vary depending on the vendor of your device.

After that, you can run it as any other app by running "flutter_application_1" from your phone's app list, and proceed to Appendix G.