TYLER: Installation Manual

Note: All installation instructions herein, are specified for the following environment:

1. **OS: Windows 10**
2. **Processor: 64-bit**

### PREPARED FOR

TechE, TechnoUtsav 3.0 2020

Deloitte

### PREPARED BY

SPIT Mumbai\_Deloitteful Devs

1. Jimil Shah
2. Rahul Soni
3. Dipam Shah

TYLER: Table of Contents

1. **Installing the necessary datastores………………………………………………………..3-4**
   1. [Installing MongoDB 3.6.17](#3aa8lbcciob3)…………………………………………………………………………..3
   2. [Installing Elasticsearch 7.6.1](#8hbqbcihodkl)………………………………………………………………………..4
2. **Setting up the backend…………………………………………………………………………..5-9**
   1. [Installing Anaconda for Windows](#hgwncutj0ivv)……………………………………………………………….5
   2. [Installing TensorFlow 2.0.0 and Keras 2.3.1 with GPU support](#8pr0xjnrhp94)……………...7
   3. [Installing the required dependencies](#d0jjimrs8wq)…………………………………………………………..9
3. **Setting up the front-end environment……………………………………………….10-12**
   1. [Installing Android Studio IDE](#ghcsmwplh37g)…………………………………………………………………….10
4. **Testing the product…………………………………………………………………………….13-16**
   1. [Running the Android application](#31oyrwe6nyhz)………………………………………………………………13
   2. [Testing the CBT chatbot via console](#d0fqhj5h28gp)…………………………………………..…….……..16

### GitHub Links for the resulting trained models and weights

All the deep learning models and their weights are uploaded to GitHub for easier access. They can be found on the below link:

<https://github.com/JimilProgGrammer/tyler-models>

The contents of the repository are as described in the below table. Clone or download this repository and place all the contents in the **Models/** subdirectory under the project’s root, if you want to directly test the application without training the models.

|  |  |
| --- | --- |
| Bot\_Model.h5  Bot\_Vocab\_Sentiment | Neural Network model for CBT chatbot. |
| [CNN-LSTM]Model.h5  [CNN-LSTM]Weights.h5 | Neural Network model to predict emotion from submitted audio recording using log-mel spectrogram. |
| model\_weights.h5  params.json  preprocessor.json | Neural network model for detecting symptoms from text records. |

**Link to the demo**: <https://drive.google.com/open?id=1ndY-6MNc-rfvsyK2Eml6EIY3a_UuyqX7>

### 

### Installing the required databases: MongoDB 3.6.17

**Download the installer (.msi) from the** [**MongoDB Download Center**](https://www.mongodb.com/download-center/community?jmp=docs)**:**

1. The Download Center should display MongoDB Community Server download information. If not, select Server, then click the MongoDB Community Server tab.
2. In the Version dropdown, select the version that corresponds to MongoDB 3.6.
3. In the OS dropdown, Windows 64-bit X64 should be selected.
4. In the Package drop down, MSI should be selected.
5. Click Download.

**Run the MongoDB installer.**

1. Go to the directory where you downloaded the MongoDB installer (.msi file). By default, this is your Downloads directory.
2. Double-click the .msi file.

**Follow the MongoDB Community Edition installation wizard.**

1. The wizard steps you through the installation of MongoDB and MongoDB Compass.
2. You can choose either the Complete (recommended for most users) or Custom setup type. The Complete setup option installs MongoDB and the MongoDB tools to the default location. The Custom setup option allows you to specify which executables are installed and where.
3. When ready, click install.
4. Once installation is complete, do the following:

Open up a command prompt and type in the following commands:

|  |
| --- |
| **>**> cd C:\ **>**> md "\data\db" **>**> setx /m PATH "C:\Program Files\MongoDB\Server\3.6\bin" |

5. Test successful installation by restarting the machine and running the command “mongod” in your prompt. If everything was done right, the last line on the console output should something similar:

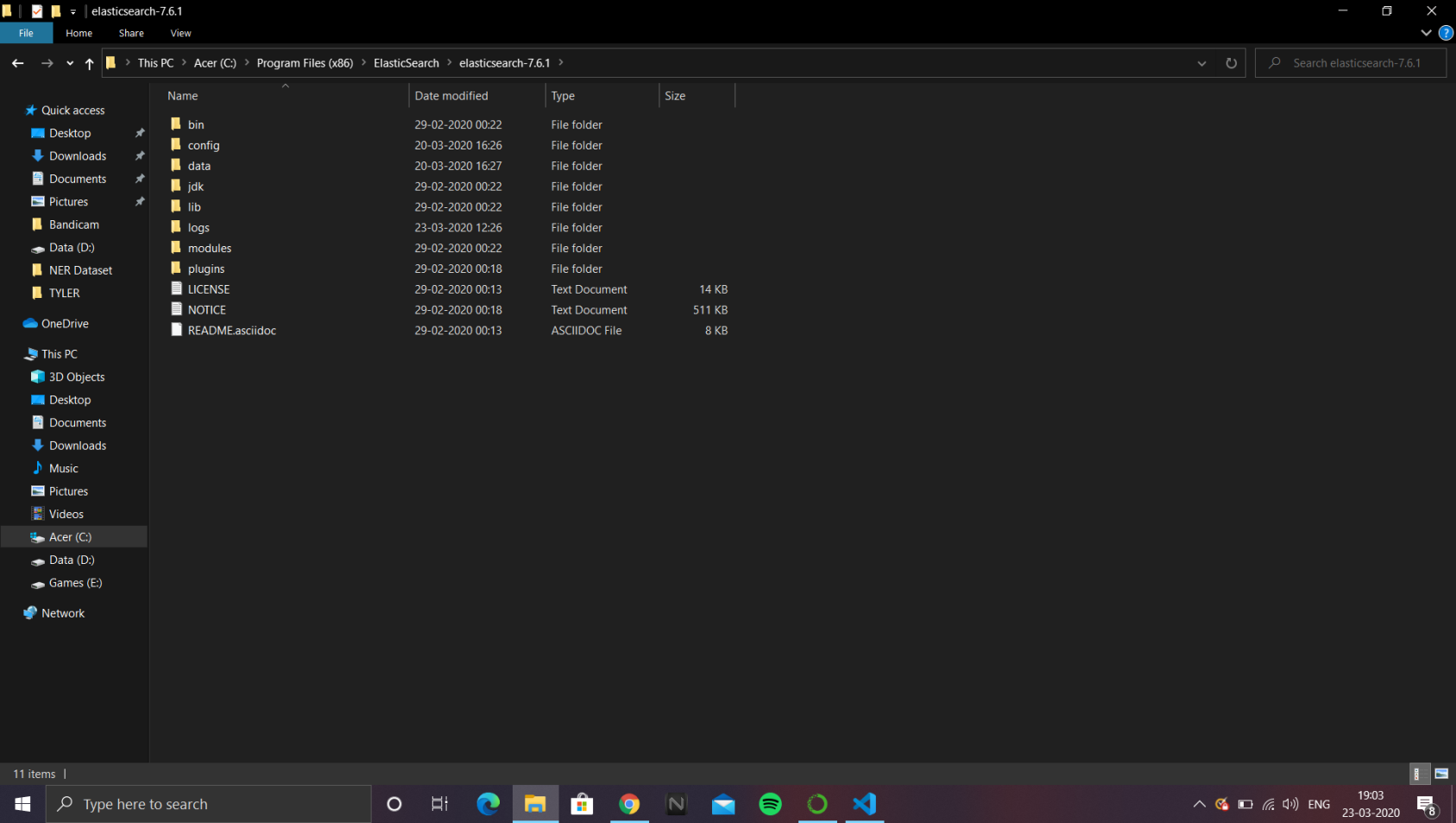
|  |
| --- |
| 2020-03-23T18:58:00.632+0530 I NETWORK [initandlisten] listening via socket bound to 127.0.0.1 2020-03-23T18:58:00.632+0530 I NETWORK [initandlisten] waiting for connections on port 27017 |

**Installing the required databases: Elasticsearch 7.6.1**

Elasticsearch can be installed on Windows using the Windows .zip archive. This comes with a elasticsearch-service.bat command which will set up Elasticsearch to run as a service.

Download the .zip archive for Elasticsearch v7.6.1 from: <https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.6.1-windows-x86_64.zip>

Unzip it with your favourite unzip tool. This will create a folder called elasticsearch-7.6.1, which we will refer to as %ES\_HOME%. After unzipping, this is what the contents of the directory look like:



And that’s it. The extracted contents from the .zip file are all that is required to run an Elasticsearch cluster locally. To test successful installation, run the below commands in a prompt:

|  |
| --- |
| **>**> cd C:\"Program Files (x86)"\ElasticSearch\elasticsearch-7.6.1\bin **>**> .\elasticsearch.bat |

If installed successfully, the last line on the console should look something as below:

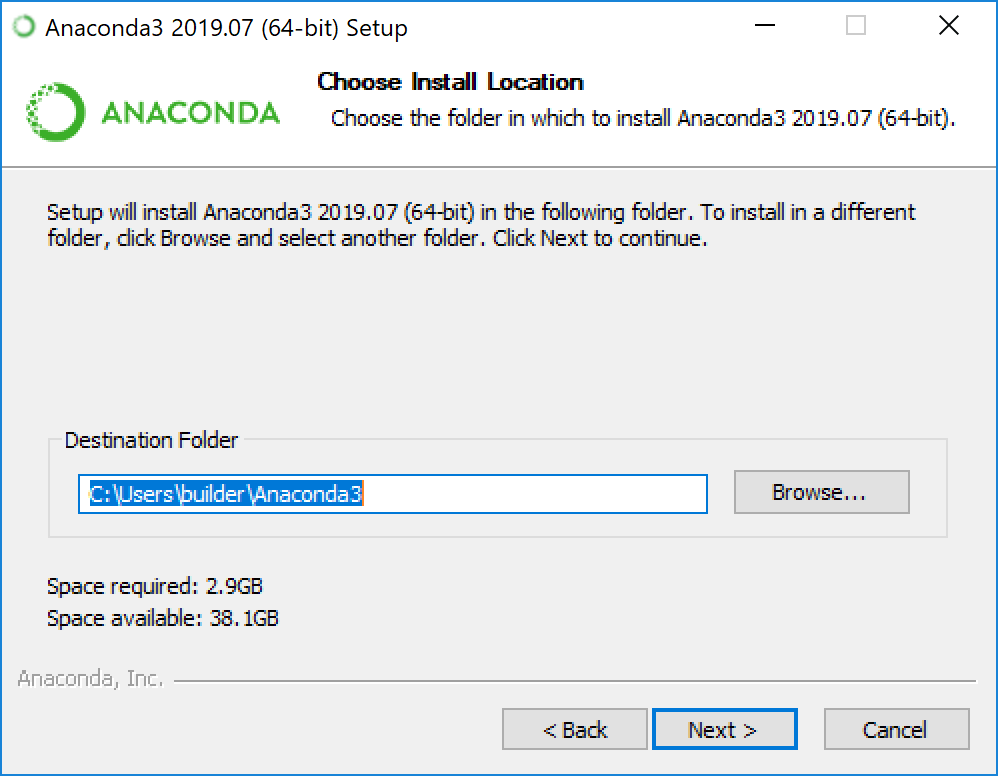
|  |
| --- |
| [2020-03-23T19:07:50,554][INFO ][o.e.h.AbstractHttpServerTransport] [LAPTOP-OO3J9621] publish\_address {127.0.0.1:9200}, bound\_addresses {127.0.0.1:9200}, {[::1]:9200} [2020-03-23T19:07:50,850][INFO ][o.e.c.r.a.AllocationService] [LAPTOP-OO3J9621] Cluster health status changed from [RED] to [GREEN] (reason: [shards started [[user\_symptoms][0]]]). |

**Installing the backend: Anaconda, Flask, Tensorflow 2.0 and Keras 2.3.1**

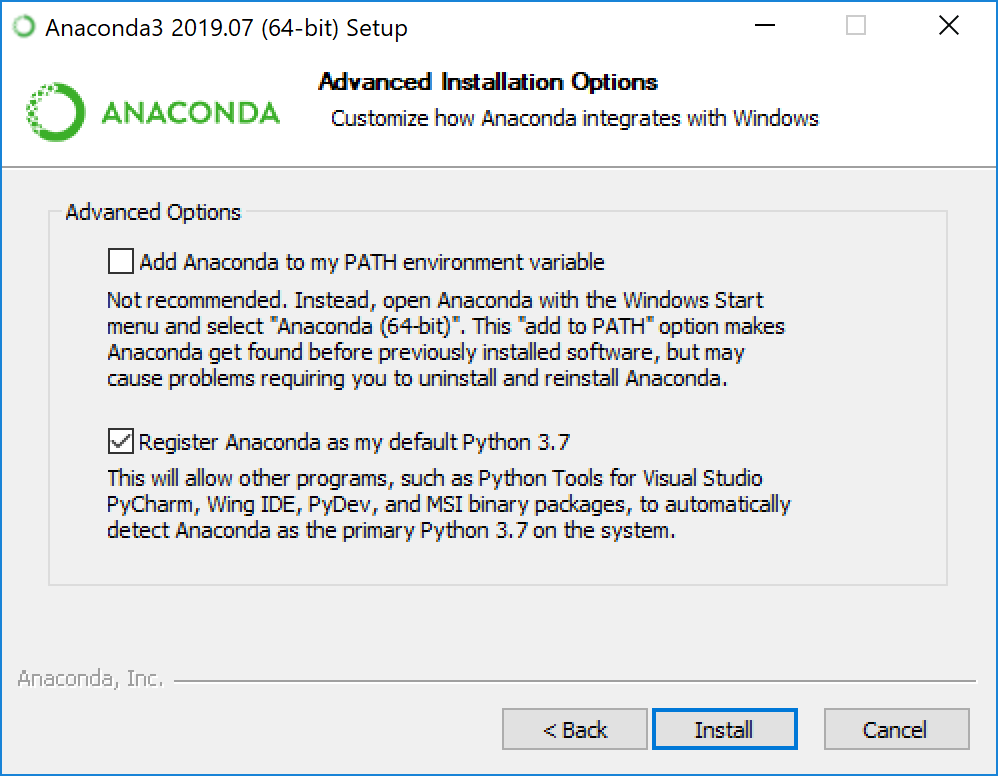
Note: The below installation instructions have been followed and must be replicated on a system with Windows 10, 64-bit processor. Additionally, it also requires a graphic processor. Following instructions used a NVIDIA GTX 1650 4GB graphics. Also, it is assumed that the host system has Python 3.6.0 installed.

**Step 1: Installing Anaconda Environment**

1. [Download the Anaconda installer](https://www.anaconda.com/download/#windows).
2. Double click the installer to launch.To prevent permission errors, do not launch the installer from the [Favorites folder](https://docs.anaconda.com/anaconda/user-guide/troubleshooting/#distro-troubleshooting-favorites-folder). If you encounter issues during installation, temporarily disable your anti-virus software during install, then re-enable it after the installation concludes. If you installed for all users, uninstall Anaconda and re-install it for your user only and try again.
3. Click Next.
4. Read the licensing terms and click “I Agree”.
5. Select an install for “Just Me” unless you’re installing for all users (which requires Windows Administrator privileges) and click Next.
6. Select a destination folder to install Anaconda and click the Next button. Install Anaconda to a directory path that does not contain spaces or unicode characters.Do not install as Administrator unless admin privileges are required.



1. Choose whether to add Anaconda to your PATH environment variable. We recommend not adding Anaconda to the PATH environment variable, since this can interfere with other software. Instead, use Anaconda software by opening Anaconda Navigator or the Anaconda Prompt from the Start Menu.



1. Choose whether to register Anaconda as your default Python. Unless you plan on installing and running multiple versions of Anaconda or multiple versions of Python, accept the default and leave this box checked.
2. Click the Install button. If you want to watch the packages Anaconda is installing, click Show Details.
3. Click the Next button.
4. After a successful installation you will see the “Thanks for installing Anaconda” dialog box. Click the Finish button.

**Step 2: Installing Tensorflow 2.0 with GPU support**

1**.** Once Anaconda is installed, open up the **anaconda prompt** from the start menu. To install tensorflow 2.0.0 in the Anaconda environment, run the below command in the **Anaconda Prompt**:

|  |
| --- |
| **>**> conda install -c anaconda tensorflow-gpu keras |

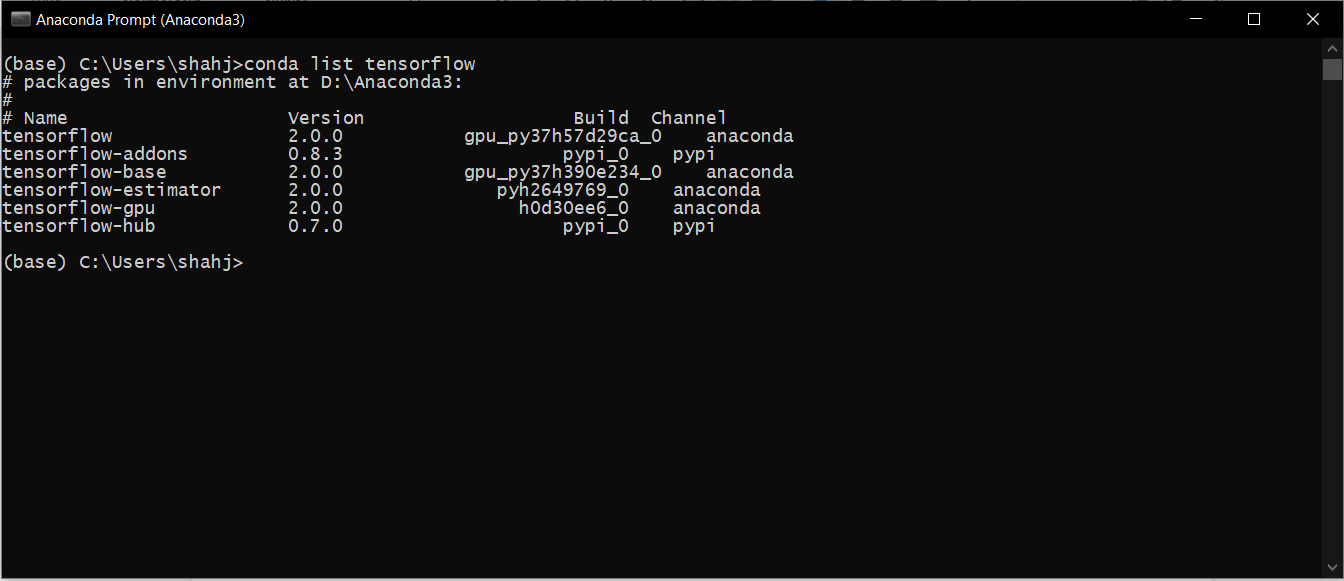
2. This should install the GPU-enabled Tensorflow library along with the required CuDNN and CUDA toolkit versions, inside the Anaconda environment.

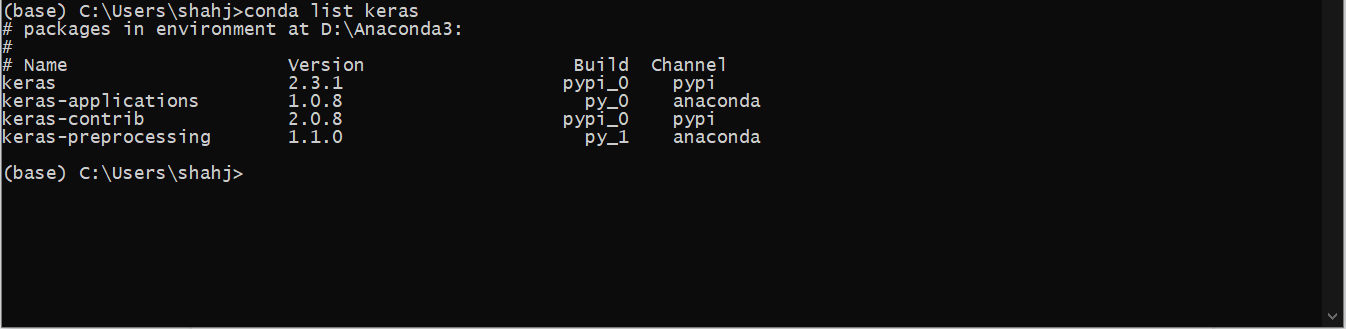
3. With TensorFlow 2.0, the Keras library is also packaged inside the TensorFlow 2 library itself.

4. This setup within the Anaconda environment will help to run the Jupyter notebooks for training that have been submitted in the .zip file. To verify successful installation, run the below command in the Anaconda prompt:

|  |
| --- |
| **>**> conda list tensorflow |

And it should give the following output:





5. Ensure that the tensorflow and keras versions match, else the Jupter notebooks might throw exceptions due to a mismatch in the versions.

6. Once TensorFlow 2.0.0 and Keras 2.3.1 are installed in the Anaconda environment, we also need to install them in the local environment for use by Flask and Python for training the model, and getting predictions from the model. Along with the GPU-enabled versions of Tensorflow 2 and Keras, we will also install the corresponding CUDA toolkit and CuDNN library for accessing the GPU.

7. Firstly, install Visual Studio Code 2017 Community Edition. This is available at the below page:

<https://visualstudio.microsoft.com/vs/older-downloads/>

Run the installer and complete the installation procedure to have Visual Studio 2017 set up on your local machine.

8. The next step is to install **CUDA Toolkit v10.0**. This toolkit is available at the below site:

<https://developer.nvidia.com/cuda-10.0-download-archive?target_os=Windows&target_arch=x86_64&target_version=10&target_type=exenetwork>

Run the installer to complete the installation procedure. Upon successful installation, you must see a subdirectory for **NVIDIA GPU Computing Toolkit > CUDA > v10.0** under the directory you had the installer install the toolkit. For example, the default path is C:\Program Files; then the **$CUDA\_HOME** variable will have the path: **C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.0.**

Also, add the following two paths to your system PATH variable in Windows by searching for “Edit system environment variables” in the search provided with the start menu in Windows 10. In the window that pops up, click on the Environment Variables button at the bottom. In the next window, in the upper half, a Path variable must be defined. Click on Edit and in the new window, click on Add to add the following two paths for the toolkit to operate:

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.0\bin

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.0\libnvvp

If they are already a part of the Path variable, we won’t be adding those again.

9. The final prerequisite for the local installation of Tensorflow is the **cuDNN library**. With the Tensorflow 2.0.0 and CUDA toolkit 10,0, the corresponding cuDNN version to be downloaded is 7.6.5 and is available from the below link:

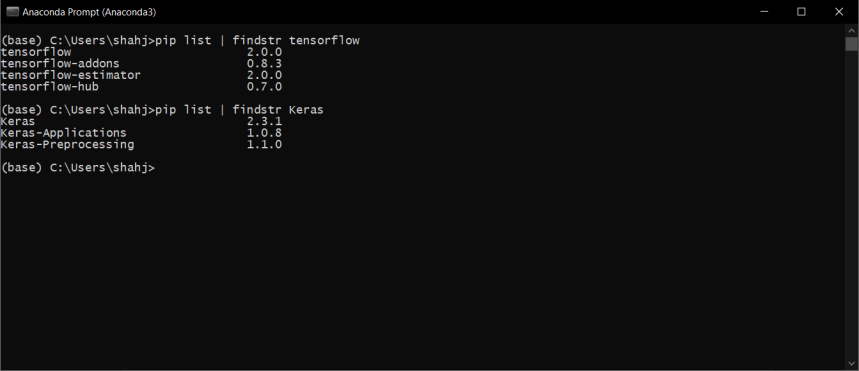
<https://developer.nvidia.com/rdp/cudnn-download#a-collapse765-10>

10. Extract the downloaded zip file. We find three sub-directories - bin, include, and lib. Copy the **cudnn64\_7.dll** under the **bin/** sub-directory to **$CUDA\_HOME\bin\** sub-directory. Copy the **cudnn.h** file under the **include/** sub-directory to **$CUDA\_HOME\include\** sub-directory. Finally, copy the **cudnn.lib** under the **lib/x64** sub-directory to **$CUDA\_HOME\lib\x64** sub-directory.

11. All the prerequisites are installed now, and we can proceed with installing the Tensorflow 2 and Keras library, locally. Run the commands in an Anaconda Prompt:

|  |
| --- |
| **>**> python -m pip install tensorflow-gpu==2.0.0 **>**> python -m pip install Keras==2.3.1 |

To test successful installation, run the pip list | findstr tensorflow command in the Anaconda prompt:



12. Along with Keras, we also install the **Keras Contrib** package, which is a community contribution package for the Keras library. This package has some advanced layers that we use in one of the models i.e. the CRF layer. To install the Keras Contrib package, run the below commands in the **Anaconda prompt:**

|  |
| --- |
| **>**> python -m pip install git+https://www.github.com/keras-team/keras-contrib.git |

To test successful installation, check the pip list again and search for keras-contrib package:



**Step 3: Setting up Flask and required libraries**

1. **Flask** is a Python micro-framework that is used for building web applications. Flask provides a very quick way to deploy vanilla Python functions as APIs, that can be used by multiple front-ends, including mobile and web applications.
2. To install Flask, run the below commands in the **Anaconda prompt:**

|  |
| --- |
| **>**> python -m pip install Flask |

1. Once Flask is installed, run the below commands to install all the **necessary libraries** for audio and text processing that are being used by the APIs:

|  |
| --- |
| **>**> python -m pip install pandas numpy scikit-learn matplotlib seaborn elasticsearch graphviz librosa pymongo pickle seqeval gensim **>**> conda install librosa pickle |

Once the libraries are installed, the last step is to install **ffmpeg** i.e. a compression software that will help the librosa library to load the audio files for processing that are submitted to it via the front-end. This is because recordings from Android are inherently compressed and require ffmpeg for reading purposes. To download ffmpeg, follow the below steps:

1. To start off, head over to the [official FFmpeg website](https://ffmpeg.zeranoe.com/builds/win64/static/ffmpeg-4.2.2-win64-static.zip) and download the 4.2.2 build.
2. Once downloaded, extract the FFmpeg to the folder or drive of your choice. Rename the extracted folder to ffmpeg for ease of access. In my case, I extracted it to the root of the C drive.
3. To add FFmpeg to Windows 10 path, search for "Edit the system environment variables" and open it and click on "Environment Variables" appearing at the bottom window. Select "Path" variable and click "Edit” and click "New".
4. Here, enter "C:\ffmpeg\bin\" and click on the "Ok" button. If you've placed the FFmpeg folder in another drive or folder then change the directory path accordingly.

Now that the backend and all its dependencies are installed, we can move to installing the necessary software and tools for building and testing the front-end.

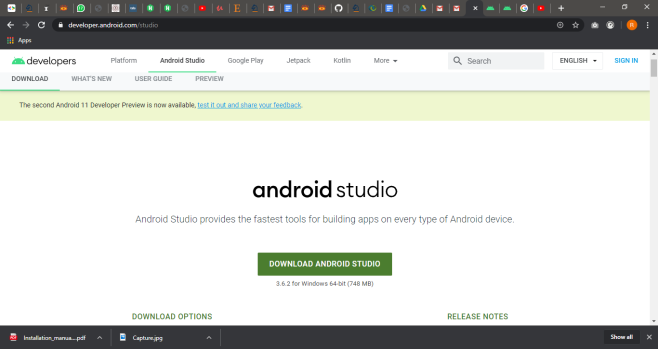
**Installing the frontend: Android Studio**

Following are the version specifications of the softwares used to develop the Android application.

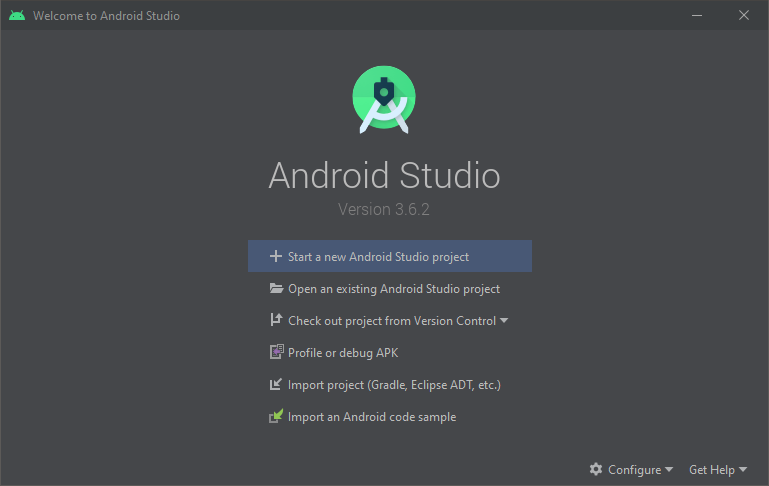
|  |  |
| --- | --- |
| Android Studio IDE | Android Studio 3.6.2 |
| SDK Platform Version | API - 27: Android 8.1(Oreo) revision 3 |
| Android SDK Tools | 26.1.1 |
| Java JDK version | 1 1.8.0\_181 |

**Step 1: To install the Android Studio IDE**

1. The Frontend of the application is built on Android Studio 3.6.2.
2. Download the Android Studio Installer(.exe) for Windows- 64 bit operating System from [here](https://developer.android.com/studio).

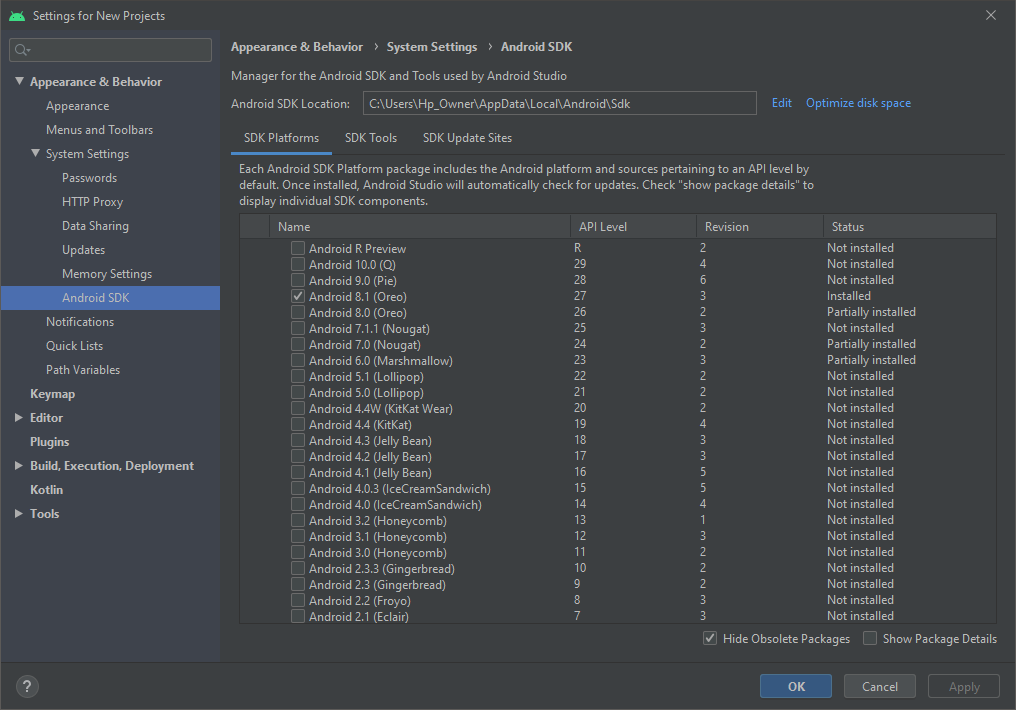


1. Open the installer on completion of download .
2. Run the wizards with the default option.
3. The final window of the installation will be



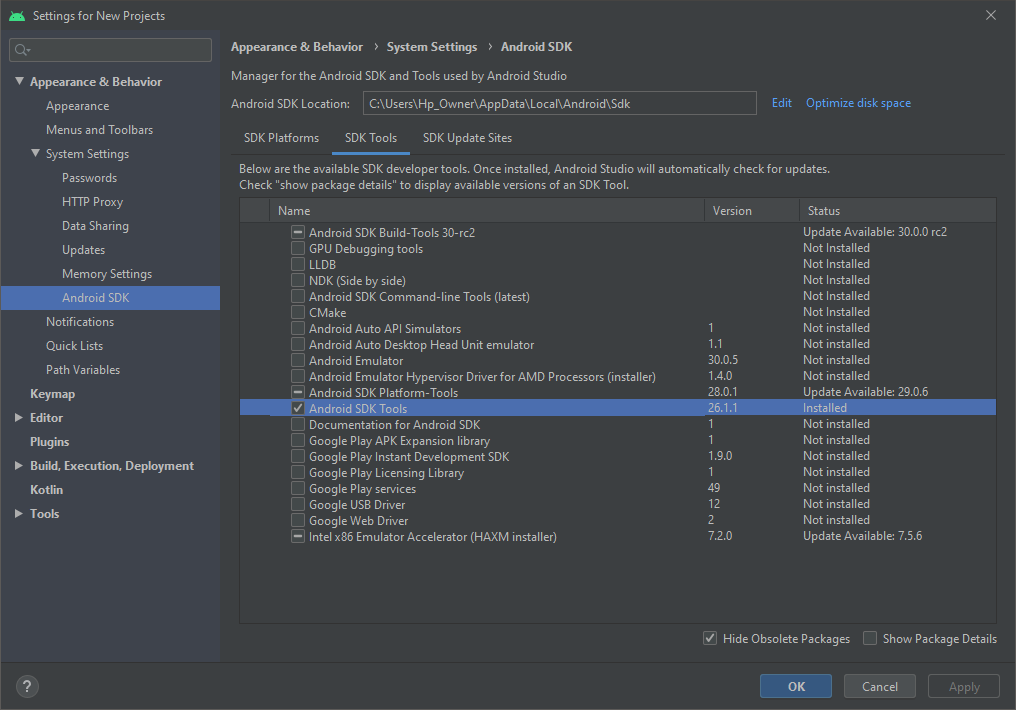
**Step 2: To setup the SDK Platform version**

1. Once you have completed the installation of the IDE, launch the IDE and open the SDK manager from the configure tab in the bottom right of the window.
2. Under the **SDK platform** tab, install the Android 8.1(API level 27). Uninstall all the SDK platforms in the list which are above 27.

****

**Step 3: Setup the SDK tools**

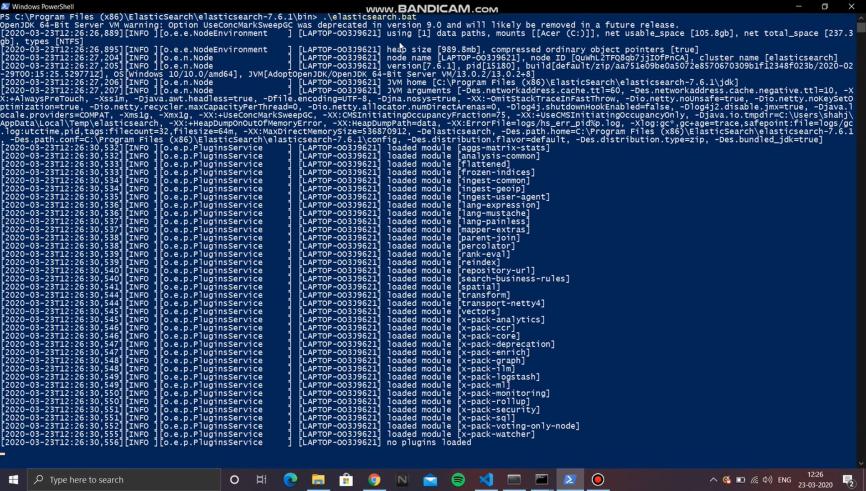
1. Open the **SDK tools** tab from the same window which was opened in the previous step.
2. If the version of installed **Android SDK tools** is less than 26.1.1(latest), the installation is complete, else install the Android SDK Tools 26.1.1 by selecting the check box besides **Android SDK tools.**
3. Click Apply.



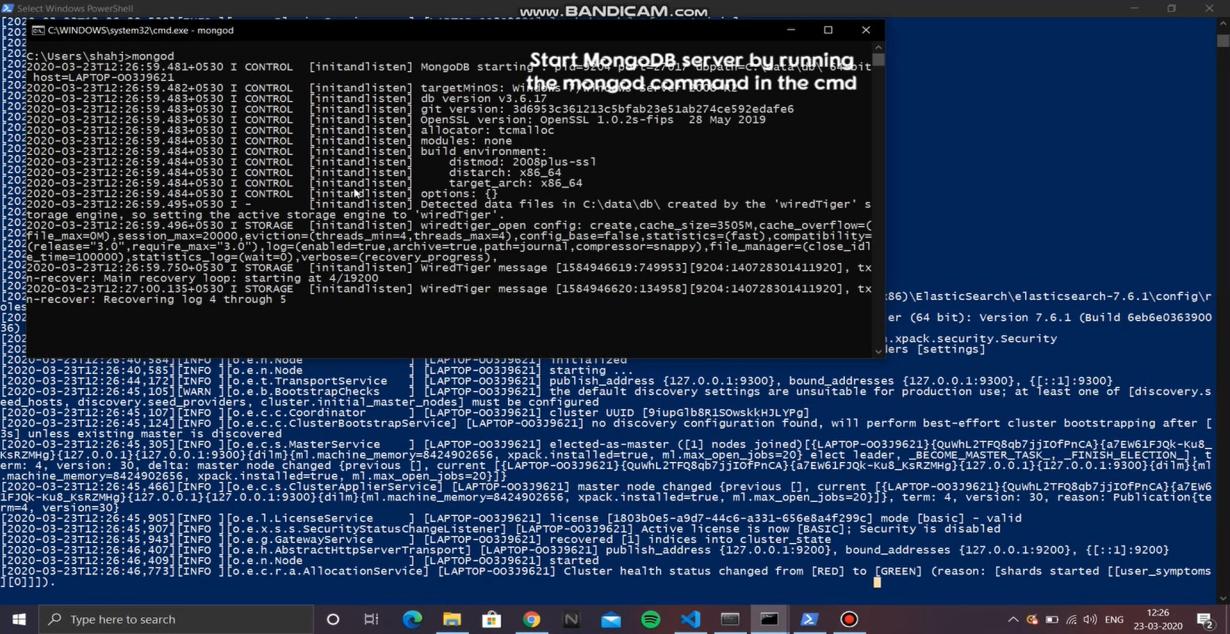
**Steps to run the application**

1, Start the necessary backend database servers i.e. MongoDB server that runs on port 27017, and elasticsearch server that runs on port 9200.

1. Start the elasticsearch server by navigating to the elasticsearch-7.6.1 folder in a command prompt or Windows Powershell. Change the current working directory to the bin/ subdirectory and run the elasticsearch.bat file to start the elasticsearch server.



1. Start the MongoDB server by opening up another command prompt or Powershell window and executing the mongod command. Open another prompt and run a MongoDB client by executing the mongo command. Then run the use tyler command to create the tyler database.



1. Open up a new Windows Command Prompt window and run the below command to create a new Elasticsearch index:

|  |
| --- |
| D:\TYLER> python create\_index.py |

Upon successful execution, the console should show the following response:

|  |
| --- |
| IN main(): Connected to ES cluster. <Elasticsearch([{'host': 'localhost', 'port': 9200}])> **{'acknowledged': True, 'shards\_acknowledged': True, 'index': 'user\_symptoms'}** **Created Index** |

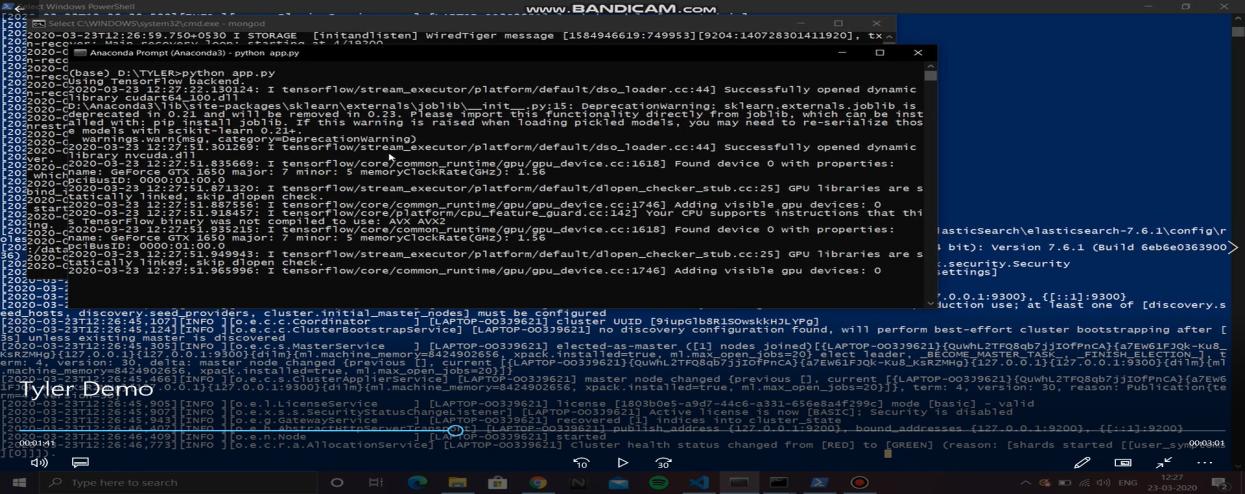
1. In the same command window, follow the below steps to create the tyler database in MongoDB:

|  |
| --- |
| D:\TYLER> mongo **>** use tyler **switched to db tyler** |

2. Run the Flask server and tunnel the port 5000 to a public URL using any tunneling utility. The below example uses ngrok to tunnel the Flask server port.

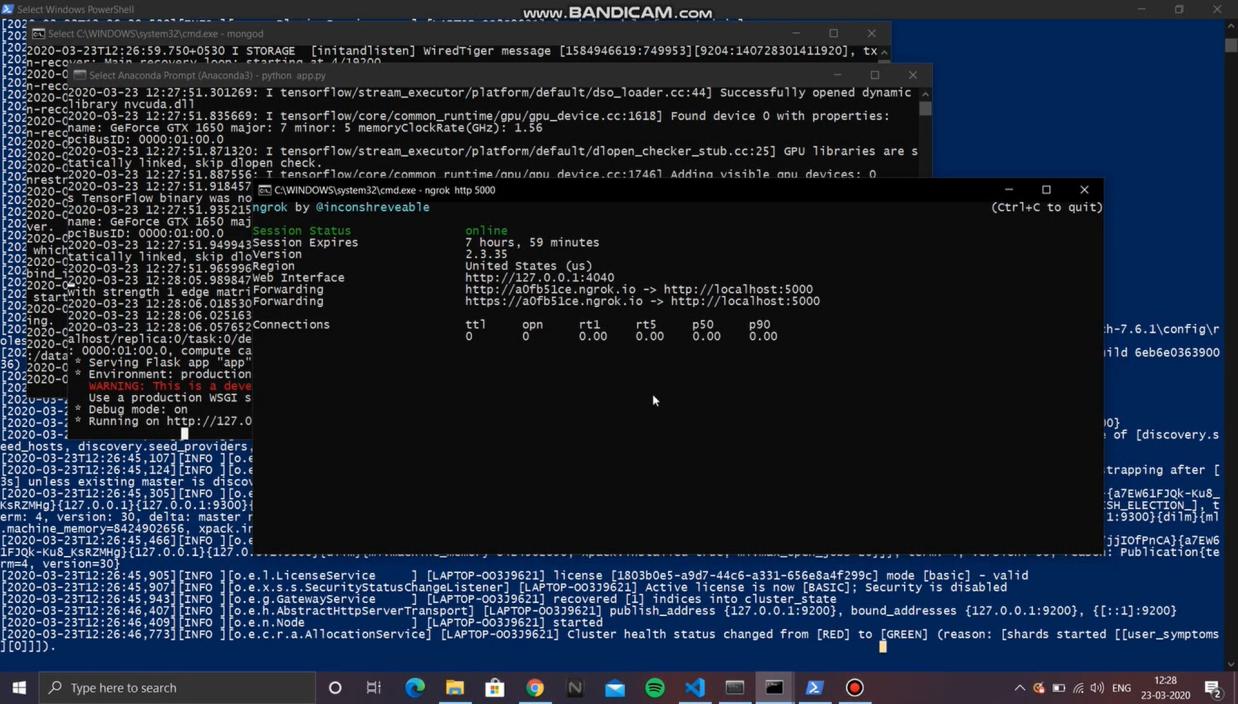
1. Open an Anaconda Prompt window and navigate to the root directory of the submitted project. Under the root directory, run the command:

|  |
| --- |
| D:\TYLER> python app.py \* Serving Flask app "app" (lazy loading) \* Environment: production  WARNING: This is a development server. Do not use it in a production deployment.  Use a production WSGI server instead. \* Debug mode: on \* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit) |



1. Once the Flask server is running on port 5000, download the ngrok utility from the below link: <https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-windows-amd64.zip>. Extract the .zip file. This will create a directory for ngrok that will have a single executable. Open this directory and Shift + Right Click to see “Open Powershell Window here”. Click on the option and in the Powershell window that pops up, run the ngrok utility by running the command:

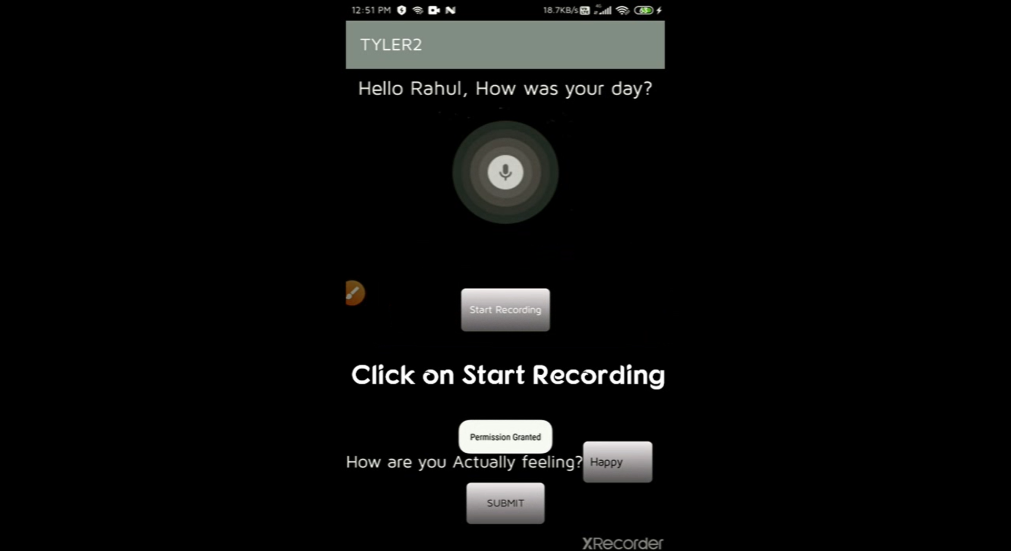
|  |
| --- |
| C:\Users\shahj\Downloads\ngrok> ngrok http 5000 |



3. Open up the frontend codebase in the **TYLER2/** subdirectory of the submitted zip file in Android Studio IDE. Edit line 115 in MainActivity.java file. This is the line where the host URL is configured. Paste the URL that was generated by the ngrok utility as a string parameter to the baseURL() method.

4. Attach an Android device to the host machine and turn on USB Debugging on the Android phone. Run the Android application on the connected phone by clicking the green play button in the top bar. This opens a “Select Deployment Target” window. Choose the Connected Device (whichever is connected to the host machine) and click the OK button.

5. Upon starting the Android app on the phone, it asks for basic storage and audio recording permissions. Grant those to the app. Once that is done, the app will be loaded and looks as below:



Upon clicking on the Start Recording button, a modal will popup. Record the audio that you want to submit. This audio can contain details about how your day was, symptoms you experienced, and the medicines you took. Once this request reaches the Flask backend, the server responds with the detected mood to confirm whether it is correct or not. The detected symptoms and medicines along with the raw text (converted from speech) is stored in the Elasticsearch cluster. The submitted audio is also stored locally in the Audios/ subdirectory. Confirm your mood and click on the Submit button to save the detected mood along with user & audio recording details in the MongoDB database.

**Steps to run the Cognitive Behaviour Therapy (CBT) Chatbot**

1. Under the root directory, you can run the bot.py file in an Anaconda Prompt by running the command: python bot.py

2. The bot starts by asking you your name. You can have a healthy chat with the bot on the console by providing answers to its simple questions.

