**Part 1: Set Up**

1. Download and install Docker Desktop: <https://www.docker.com/products/docker-desktop>  
     
   Docker is a free container application that sets up a virtual environment with parameters of your choosing. Using Docker largely eliminates issues regarding dependencies, versioning, and system differences that naturally occur over time as technologies age and code becomes obsolete. In our case, the application was originally written in Python 3.8 and has dependencies on the *geopandas* and *matplotlib* libraries. There are ways to make it run smoothly in Python 3.10 today, however it would require a lot of setup. Using Docker is simply more efficient.

**Part 2: Personalizing the code for customized output**

1. By default, the application will produce a map of the City of Toronto proper (the 140 neighbourhoods) with the origin node for all calculations set to UofT, accompanied by a .csv file describing the share of census tracts fastest reached by each transport method.  
     
   At some point however, you may wish to customize the output, for example to produce a map with different origin nodes, or of the GTA rather than only core Toronto. I have tried to streamline this process as much as possible so there are only a few places you would have to adjust the code.   
     
   Most of these customizations are optional and the application will run fine without them, however, please be sure to update line 10 in directions.py with your Google Maps API key. **The requests will fail without this change.** Below is a list of customizable options in the application:

* directions.py, lines 7-9: These lines contain place IDs of the 3 origin nodes we had initially agreed upon (UofT, Union Station, and Pearson Airport). You may wish to change these nodes or add more for different maps.
* **directions.py, line 10**: Contains a variable for the Google Maps API key used to make requests. Please replace the value with your API key before running the application.
* directions.py, line 154-165: Contain the function calls to make the requests and record the data, given an origin node. By default, only the first function call will run, corresponding to UofT as the origin node. The latter 2 calls, corresponding to the Union Station and Pearson Airport origin nodes, are commented out. Feel free to comment out/uncomment any of the 3 calls based on what maps you would like to produce.
* visualizer.py, line 13-14: Contain 2 variables corresponding to the size of the map (“core” Toronto or “greater” Toronto), and the origin node (“uoft”, “unionst”, or “pearson”). Feel free to change either of these variables as per the map configurations you wish to produce. Please be careful with the string values you assign to these variables, as any typos may result in the wrong data being accessed and merged, or no data being accessed at all.

**Part 3: Running the application:**

1. Launch Docker, which you installed in Part 1. It may take a minute for the engine to start.
2. Open the commands.txt text file. Here are 4 commands you will need to run the application in the Docker shell.
3. In your File Explorer, navigate to the project folder which contains all the files and data for this application. Hold Shift and right-click on some empty space within the window, and select “Open PowerShell window here”. This option should be approximately in the middle of available options. This will open up a new terminal window.
4. Enter the first command from the commands.txt file: docker-compose up --build -d, and press Enter. This will create the Docker shell and the environment in which to run the application. It may take a minute or 2 to finish setting up.
5. Once you are able to enter commands again, paste the 4th command from the commands.txt file: docker exec -it toronto-transport-methods bash and press Enter. This command may also take a minute or 2 to run.
6. Once the previous command has completed, type in generate.sh and press Enter. This will run the 3 primary files of the application: data\_formatter.py, directions.py, and visualizer.py. This command will take several minutes to run as it makes all the requests to Google Maps and records all the results.
7. Once execution is complete, the results are ready. The map(s) you have just produced will be in the Maps folder. There will also be a .csv file corresponding to each of your maps describing the share of census tracts that are fastest reached by each transport method. The naming convention for the maps is “[***origin***]\_[***map\_size***]\_coloured.png” and the transport shares .csv file is “transport\_shares\_[***map\_size***]\_[***origin***].csv”, where ***origin*** corresponds to the origin nodes (“uoft”, “unionst”, or “pearson”), and ***map\_size*** corresponds to whether “core” or “greater” Toronto is being mapped.
8. Once you are done with running the application, you may paste the 2nd command from the commands.txt file, docker-compose down -v, and press Enter. This will shut down the Docker environment, and once the command is complete, you may exit Docker and all project windows.