LIGHTBOT

An Adaptive Traffic Light Control Solution

TECHNICAL INSTALLATION MANUAL

Table of Contents

1 INTRODUCTION

The LightBot system is designed to improve the flow of traffic at an intersection. This is done by using an Reinforcement Learning algorithm to decide how to change traffic lights in various scenarios. The system is divided into multiple components, each designed to perform specific tasks in the system. For more details regarding the various components, please refer to the Software Requirements Specification.

This document provides the user with all the information necessary to install the LightBot system on their own machine.

2 OVERVIEW OF DOCUMENT

The remainder of this document is structured as follows:

SYSTEM REQUIREMENTS

Before proceeding. please ensure that your system adheres to the following requirements:

- Operating System: Ubuntu 18.04 or later is preferred.
- Processor: Intel Core 2 Duo or better
- RAM: 2GB or better
- At least 2GB of additional Hard Disk Space
- Connection to the Internet
- Must be logged in as an Administrator of your machine

OVERVIEW OF COMPONENTS

The components of the LightBot system which require attention are:

- Web Application
- Web Server
- Traffic Simulation

Different software packages are required for the full operation of the system, as described in the following sections. Please follow each section carefully, paying attention to version numbers and so forth.

For assistance, please contact us at teamgradient301@gmail.com.

DOWNLOADING LIGHTBOT

Cloning the Repository

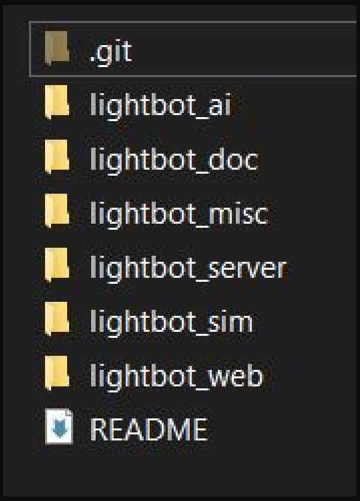
All the necessary files for LightBot are available on a **GItHub repository**. This repository should be cloned to your machine. The repository may be accessed at the following link:

https://github.com/COS301-SE-2020/LightBot

Please note that you need to have a GitHub account, have git installed on your system, and may only access the repository upon invitation. You may use Git command line, or the Github Desktop application to clone the repository.

File Structure

After successfully cloning the repository, you will see the following folder structure:



.git
lightbot_ai
lightbot_doc
lightbot_misc
lightbot_server
lightbot_sim
lightbot_web
README

Figure 1: File structure

These folders contain all of the files that make up the LightBot system. A short description of each folder follows:

- lightbot_ai: Contains files for the automated traffic controller
- lightbot_doc: Contains all documentation related to LightBot
- lightbot_misc: Contains miscellaneous files
- lightbot_server: Contains all files for the Web Server
- lightbot_sim: Contains all files for the Traffic Simulation
- lightbot_web: Contains all files for the Web Application

You may not see the ".git" folder deping on your system settings, don't worry about it.

We will now go through the software that needs to be installed in order to start the system. Note that some of the components of the LightBot system have both a front end and a back end, while some have only a back end. This will be highlighted in the following sections.

WEB APPLICATION

Front End

The Web Application will be viewable on a web browser. This serves as the only point of interaction between a user and the system. Please make sure you have one of the following web browsers installed:

- Google Chrome
- Mozilla Firefox
- Safari

Chrome is the recommended browser to use. Please do not use Internet Explorer.

Back End

The Web Application runs with a JavaScript library called React. To install React, please do the following:

1. Download and install the Node Package Manager (NPM) . Please use this link: https://www.npmjs.com/get-npm for further instructions.
2. Once NPM is installed, navigate to the lightbot_server folder, see figure 1. Open a terminal window in this folder and run “npm install”, to install all dependencies required by the server.
3. After successfully running npm i, run “npm run prerun” in the same folder used in step 2,, ie the lightbot_server folder.

WEB AND SIMULATION SERVER

We recommend installing the web server and simulation on a single Ubuntu Server 18.04 Linux machine (root privileges are required).

Installation

The Simulation Server

1. Install Yarn. Please use this link: https://linuxize.com/post/how-to-install-yarn-on-ubuntu-18-04/ for further instructions.
2. Install Python. Please use this link: https://phoenixnap.com/kb/how-to-install-python-3-ubuntu for further instructions.
3. Install Python-setuptools. Please use this link: https://zoomadmin.com/HowToInstall/UbuntuPackage/python-setuptools for further instructions.
4. Install SUMO. Please use this link: https://sumo.dlr.de/docs/Installing.html for further instructions.
5. Install SUMO-Web3d:
    a. Download SUMO-Web3d files from https://github.com/sidewalklabs/sumo-web3d.
    b. In the git package downloaded in step a, navigate to the sumo-web3d folder. Replace the scenarios.json file and scenarios folder with the files in the lightbot_sim files from Lightbots git repository
    c. Once the files in b have been successfully replaced, follow the instructions in the readme in the SUMO-Web3d repository.
6. Install Concurrently. Please use this link: https://www.npmjs.com/package/concurrently for further instructions.

The Web Server

1. Node incl. npm needs to be installed on the machine. Download and install the Node Package Manager (NPM) . Please use this link: https://www.npmjs.com/get-npm for further instructions.

2. Upload the lightbot_server folder from the Lightbot repository to the server. Navigate to this folder on the server.

3. Run 'npm install' to download node modules not included

4. Set sensitive environment variables as per instructions available in readme.

5. run 'concurrently "npm start" "sumo-web3d". This will start both the web server and simulation server.

Testing and Maintenance

1. All maintenance is made easy due to the Model-View-Controller structured design
    a. New routes can be added simply by adding them to the route manager
    b. New data structures can be supported by creating schema models

    c. New operations can be added via controllers with all generic operations such as mailing, authentication etc are available to any new operation as utilities and middleware's.
2. Unit tests can be accessed within the /spec directory and run by executing the command 'npm test'. Expansions on these tests can be made simply by adding a new described test

Endpoints

1. Currently supported endpoints are those of user (supporting the web application for client information management) and data (for operational data processing and visualisation) and can be accessed off the public DNS of the ubuntu server on port 8000
2. Any unsupported endpoints will return an Http Error 404 Resource not found
3. User endpoint, properties, and operational use

Request	Path	Access	Description
GET	/register	Public	User registration
GET	/login	Public	User sign in
GET	/logout	Private	User sign out route
GET	/me	Private	Get user profile
PUT	/update-details	Private	Update user details
PUT	/update-image	Private	Update user profile image
PUT	/update-password	Private	Update password from application
POST	/recover-password	Public	Password recovery
PUT	/reset-password/:passresetid	Public	Email password reset
DELETE	/delete	Private	Delete account
GET	/list-user	Private	List other users

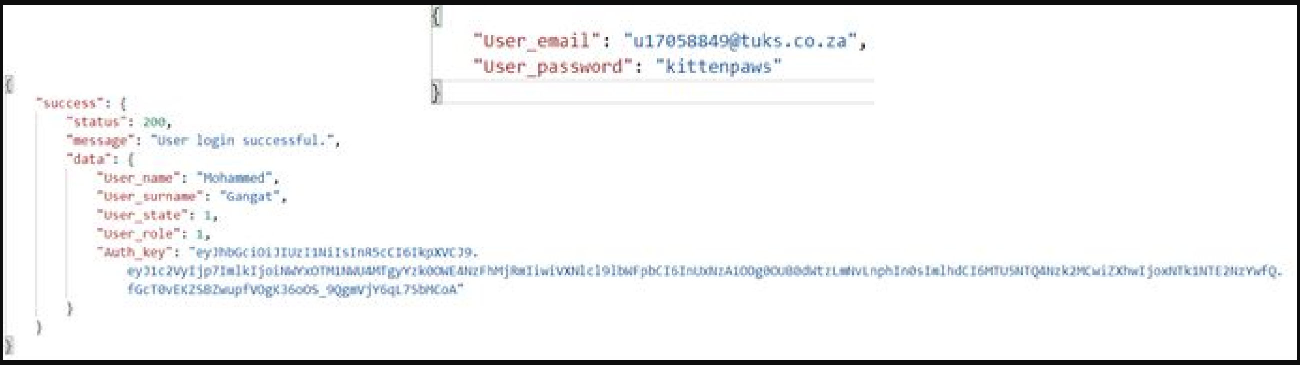
Table 1: User endpoints, properties and operational use

4. Data endpoints, properties, and operational use

Request	Path	Access	Description
GET	/graph	Private	Get latest graph data
GET	/forum	Private	Get latest forum data
POST	/post-forum	Private	Create a new forum post
GET	/state	Private	Get latest state of system
PUT	/post-forum/:forumpostid	Private	Edit a forum post
DELETE	/delete-forum/:forumpostid	Private	Delete a forum post

Table 2: Data endpoints, properties, and operational use

5. Examples of JSON data formats are as follows as well as an example of how data is returned:



```
{
    "User_email": "u17058849@tuks.co.za",
    "User_password": "kittenpaws"
}
```

```
{
  "success": {
      "status": 200,
      "message": "User login successful.",
      "data": {
          "User_name": "Mohammed",
          "User_surname": "Gangat",
          "User_state": 1,
          "User_role": 1,
          "Auth_key": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
              eyJ1c2VyIjp7ImlkIjoiNWYxOTM1NWU4MTgyYzk0OWE4NzFhMjRmIiwiVXNlcl9lbWFpbCI6InUxNzA1ODg0OU80dWtzLmNvLnphIn0sImlhdCI6MTU5NTQ4Nzk2MCwiZXhwIjoxNTk1NTE2NzYwfQ.
              fGcT0vEK2S8ZwupfVOgK36oOS_9QgmVjY6qL75bMCoA"
      }
  }
}
```

TRAFFIC SIMULATION

In order for the Traffic Simulation to run, the following dependencies must be installed:

- Python 3 and additional Python packages
- SUMO

Installing Python

Python should already be installed from the previous sections.

In addition to Python, the following packages are also required. Open up a command prompt/terminal window and type in the following commands. Please type them in one at a time, waiting for the download to finish after each command is executed:

1. pip install numpy
2. pip install matplotlib
3. pip install tensorflow (Note: this one will take a while)
4. pip install traci
5. pip install sumolib