Software Requirements Specification

for

Web & Android Application Development II

Version 1.0 approved

Prepared by Romario Renée and Nigel Francis

NIRO Solutions

June 12th 2019

Table of Contents

Table of Contents ii

Revision History iii

1. Introduction 1
   1. Purpose 1
   2. Document Conventions 1
   3. Intended Audience and Reading Suggestions 1
   4. Product Scope 1
   5. References 1
2. Overall Description 2
   1. Product Perspective 2
   2. Product Functions 2
   3. User Classes and Characteristics 2
   4. Operating Environment 3
   5. Design and Implementation Constraints 4
   6. Assumptions and Dependencies 4
3. External Interface Requirements 5
   1. User Interfaces 5
   2. Hardware Interfaces 10
   3. Software Interfaces 10
   4. Communications Interfaces 10
4. System Features 11
   1. View Shuttles 11
   2. Edit Shuttles 11
   3. Delete Shuttles 12
   4. Add Shuttles 12
   5. View Students 13
   6. Edit Students 13
   7. Delete Students 14
   8. Add Students 14
   9. View Drivers 15
   10. Add Driver 15
   11. Enable / Disable Drivers 16
   12. Insights 16
   13. Manage Admin Users 17
   14. Contact Support 17
   15. Admin Log 18
   16. Simulation 18
   17. View News 19
   18. Add News 19
   19. Translate News 20
   20. Translate Shuttle Stands 20
   21. Push Notifications 21
5. Other Nonfunctional Requirements 21
   1. Performance Requirements 21
   2. Safety Requirements 21
   3. Security Requirements 21
   4. Software Quality Attributes 22
   5. Business Rules 22
6. Other Requirements 22

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of this document is to build an online system to manage shuttles, students, driver and provide reports and analysis to improve the shuttle service management.

This web application for the UWI Shuttle Service is created with the focus of allowing admin personnel to manage data about the students and the shuttles coming into the database, being able to analyse this data and also generate reports based on the database information.

## Document Conventions

This document uses the following conventions.

DB - Database

ER - Entity Relationship

UWI - University of the West Indies

DO - Digital Ocean

## Intended Audience and Reading Suggestions

This project is a prototype for the UWI Shuttle Service system and it is restricted within the UWI CIIT premises. This has been implemented under the guidance of teachers at GIST / UWI CIIT.

This document is also intended for the developers of the project allowing them to get a deep dive into what the software is aimed at achieve, how it was architected, the role it plays with its other components and how it was developed.

## Product Scope

The purpose of the Shuttle Service management system is to improve the UWI Shuttle Service System in terms of the digital data collection and also providing a platform for students to easily board the shuttle. The system is a web based online application that is easy-to-use for management who will need to review shuttle data, analyse it and create reports based on it. The system is based on a non-relational database called Firebase. We will be aiming to provide the best experience for admin personnel as they use the system on a daily basis to allow for maximum efficiency.

## References

Building a single page Flask App on Digital Ocean by Peter Kazarinoff

<https://pythonforundergradengineers.com/flask-app-on-digital-ocean.html>

# Overall Description

## Product Perspective

This product is one (1) part of three (3) different systems that are all integrated to form a solution that would replace the physical log book currently being used by the University of the West Indies Shuttle Service drivers and admin staff. It is designed to be the portal that manages all of the data being collected by the system on a daily basis with the ability to run analysis and also generate various reports based on the data and the analysis.

A shuttle service database system such as this will be storing information such as:

* **Shuttle Data**

This includes start and end time for every route, the driver for that route at that specific time. This information is used to keep a log of what is going on on a daily basis, it is also used to create analysis that will help make the execution of the Shuttle Service more efficient.

* **Student Data**

The includes basic information about the student such as their full name, photo, UWI ID number, faculty, their arrears status, which routes they have taken. This information is used to help identity a student in the event that an incident occurs on the shuttle and for emergencies.

* **Driver Data**

This consist of basic information such as name, telephone number, email address. This information is used to sync a driver with a shuttle and to identify them.

## Product Functions

The system focuses on executing the following functions in order to allow the admin user to perform their task.

* + Login
  + Shuttle Management
  + Student Management
  + Driver Management
  + Admin Staff Management
  + Shuttle Analysis
  + Shuttle & Student Review based on Data Collection
  + Shuttle Reporting

## User Classes and Characteristics

Users of the system should be able to retrieve information regarding the shuttles, students and drivers, the schedules, how many students were on board during each trip, how many trips were done, which students were on board and the driver for that specific shuttle. The admin users should also be able to get analysis of that data that can help them to improve the efficiency of the shuttle service.

The admin user should be able to do the following functions.

View Shuttles

Add a New Shuttle

Update a Shuttle

Remove a Shuttle

View Students

Add Students

Update a Student

Remove a Student

Analysis

Get overview of total shuttle trip

View average students on board per trip

View number of students left per trip

## Operating Environment

The UWI Shuttle Service Web Manager was tested in various environments and those results provided us with the best operating environment this web application can run in. It is compatible with various Operating Systems:

* Non-SQL Database
* Client/Server System
* Database: Firebase
* Platform: Flask / Python
* Operating systems:

Windows - 7, 8, 10

Mac OS - High Sierra, Mojave

Linux - Ubuntu

**Browser based, the web application works on the following browsers:**

Chrome - Version 74.0.3729.169 (Official Build) (64-bit)

Firefox - Firefox Quantum 66.0.3 (64-bit)

Safari - Version 12.1.1

Brave - Version 0.64.77

Yandex - 19.4.0.2400 (64-bit)

The application in its first version ( 1.0.0 ) makes use of an external Google service called Firebase and in certain areas of the world such as China, these services are blocked, so as a result, the required minimum environment will need to consist of a VPN along with a stable internet connection of at least 4.0 Mbps.

## Design and Implementation Constraints

The project is susceptible to a few constraints due to the location of the University. The application will mainly be used and is targeted to countries such as Barbados, Jamaica & Trinidad. However, The University of the West Indies has connections with other universities around the world. One of them being Gaobo in Suzhou, China. This specific location brings a few constraints, in China, Google services are blocked which means the web application will not be able to fetch any data. This provides a very serious implementation challenge as it halts the use of the system totally unless a VPN is used.

One of the other constraints are offering multilingual options to the admin staff using the application. The system is built using a focus on only English, NIRO Solutions would have to slightly change the template system used to make it more flexible in order to allow for language changes.

## Assumptions and Dependencies

Let us assume that this UWI Shuttle Service has implemented the other two components - the driver android application and that the student iOS and Android applications have been distributed to most students who use the service on a daily.

Every time the shuttle is used, the driver signs in, selects the route and then allows students to have their QR scanned by the system.

Most students have the app and they are using the QR codes provided in order to board the shuttle.

Assuming that these two transactions are occurring on a weekly basis on most of the shuttles, we are providing that data collected to the admin user in a manner where they can understand what is going on with the service.

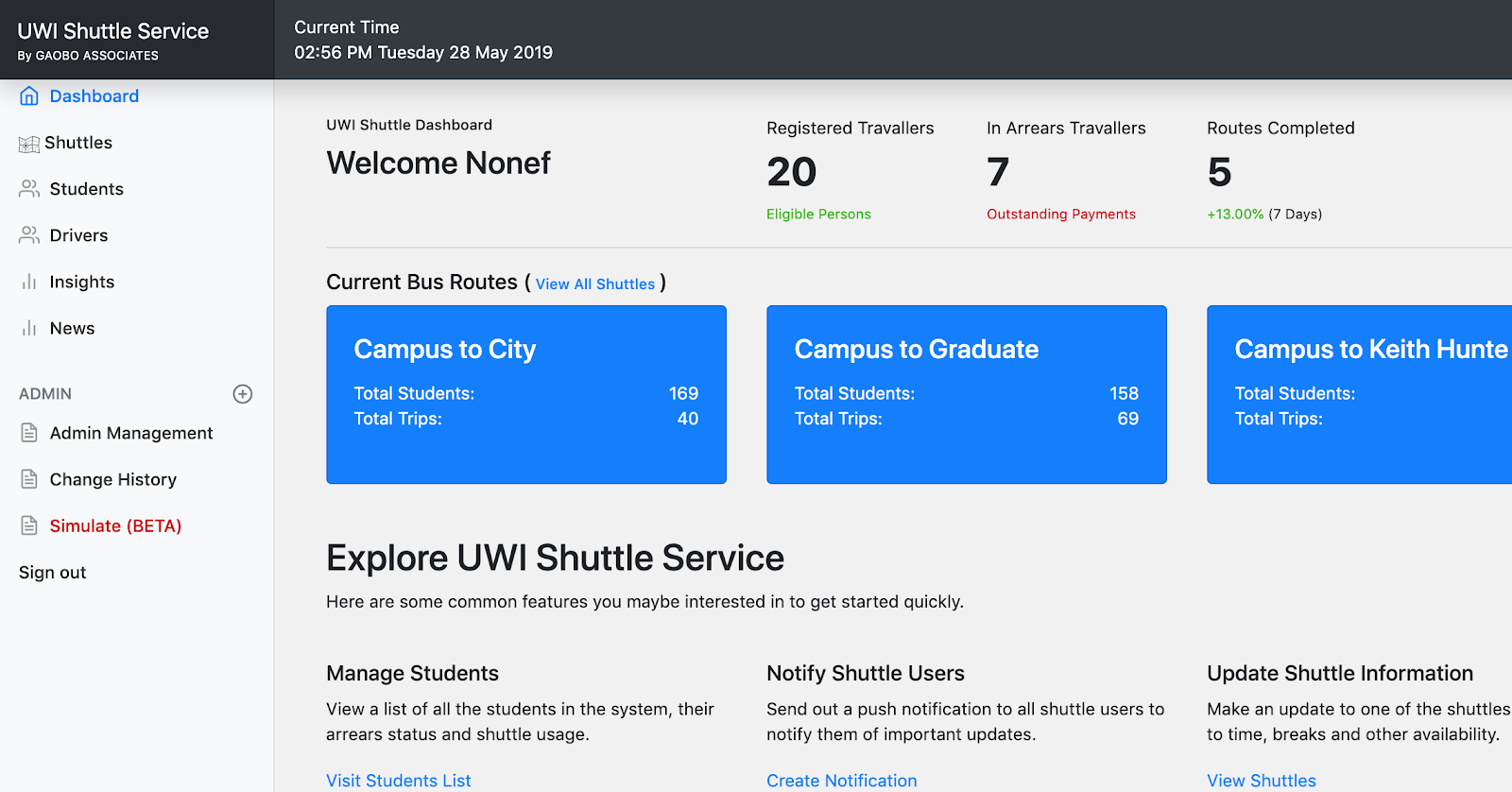
# External Interface Requirements

## User Interfaces

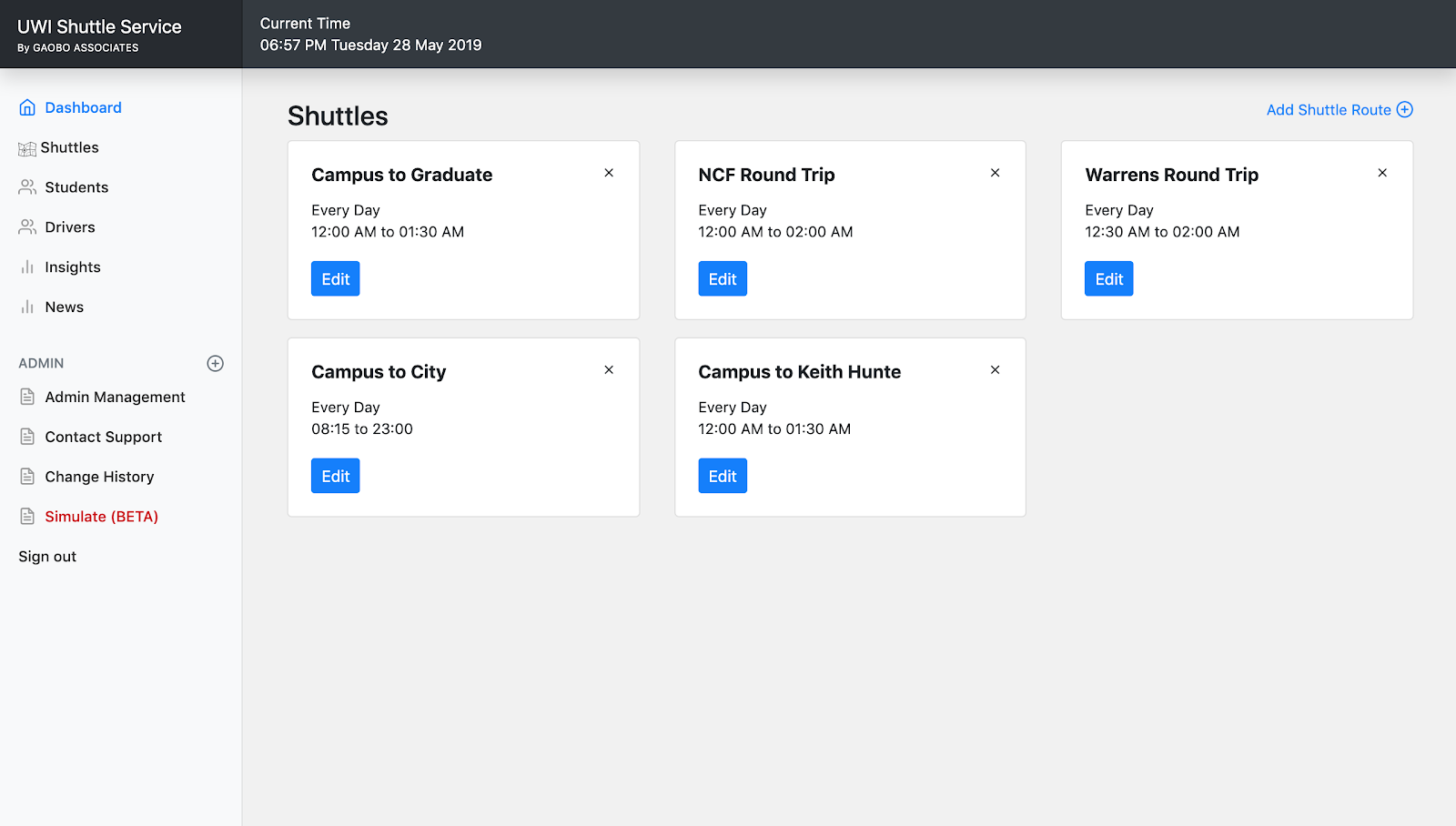
Front End - Python Flask

Backend - Firebase

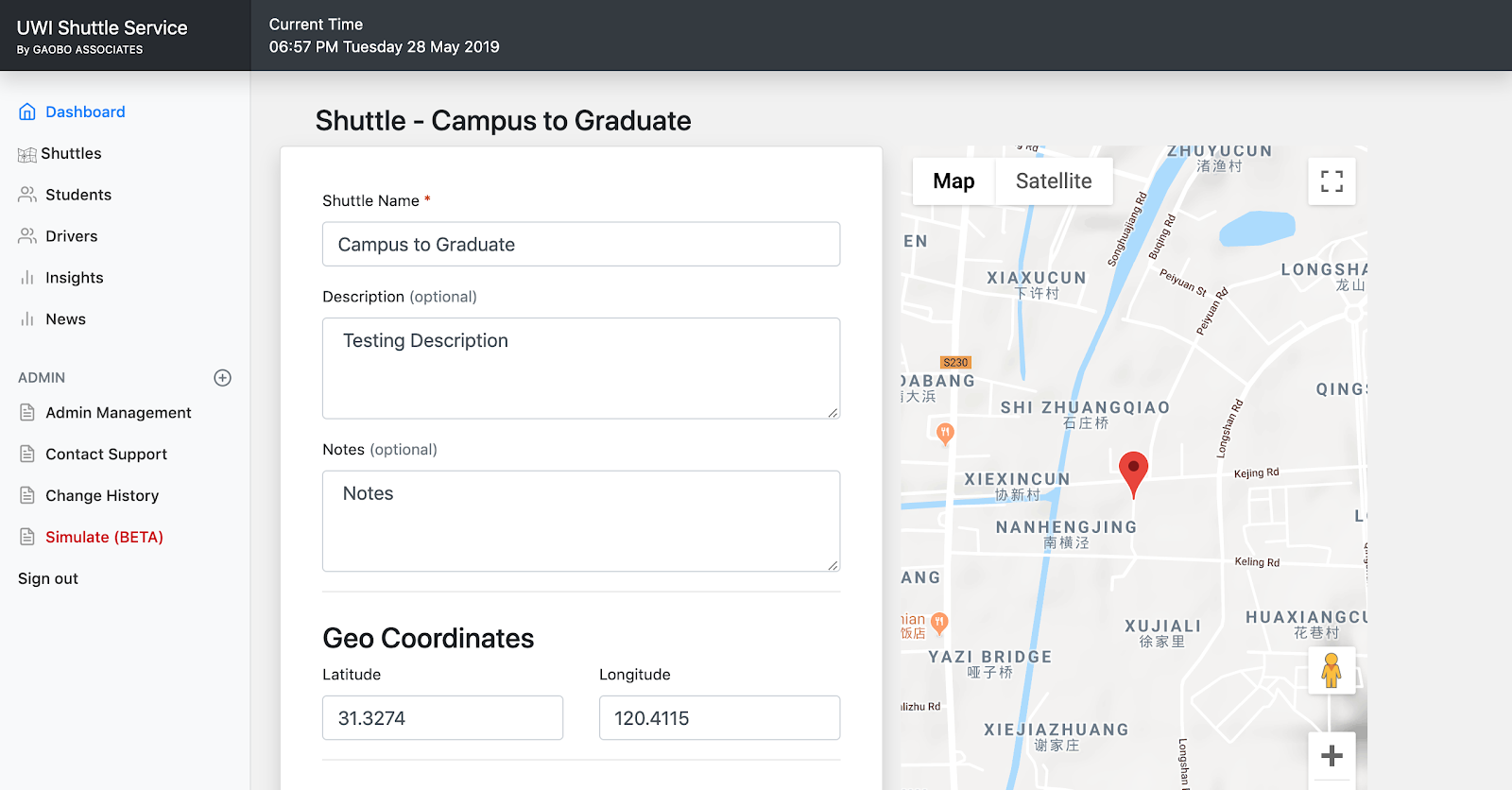
v



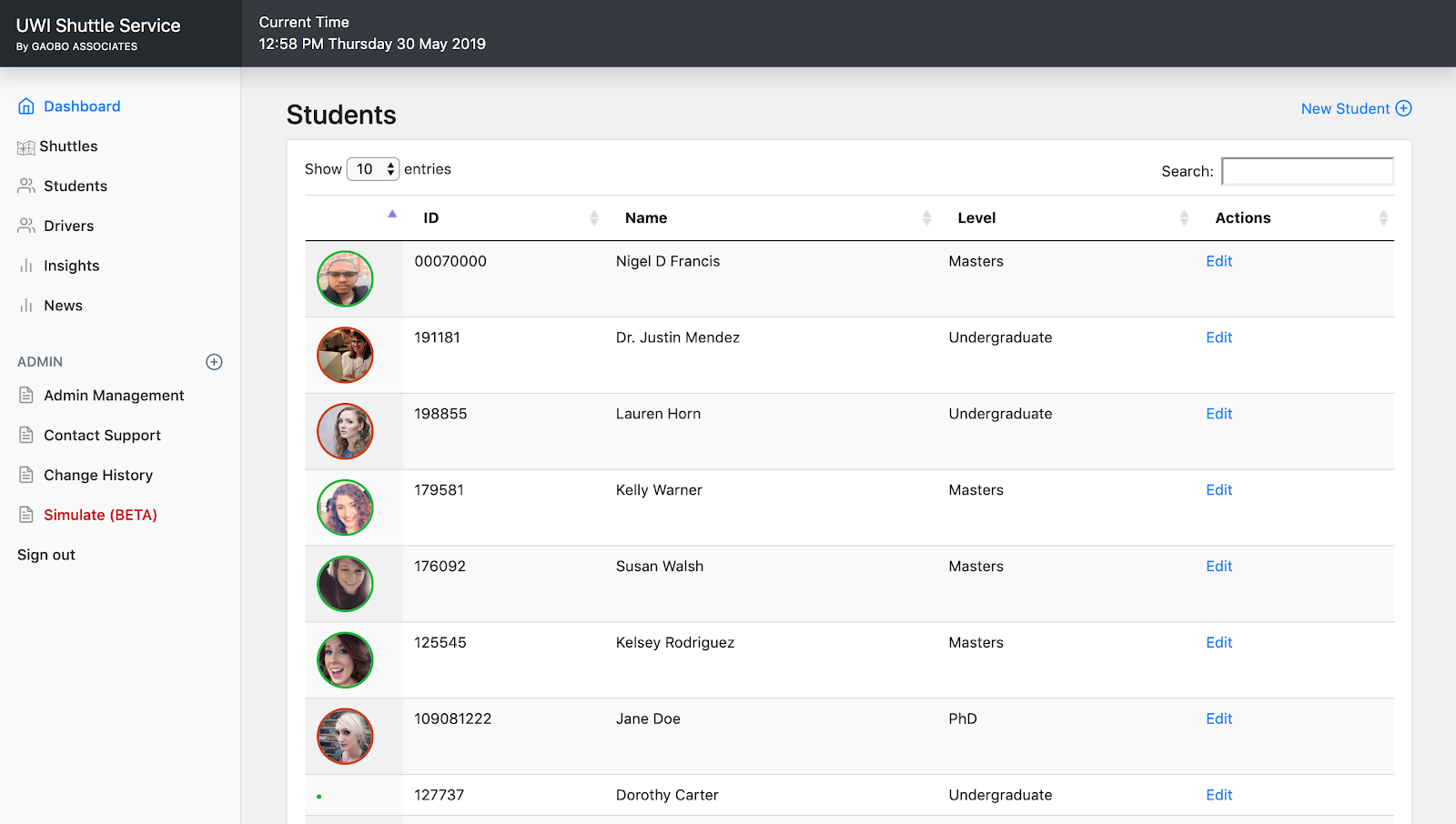
Dashboard Interface



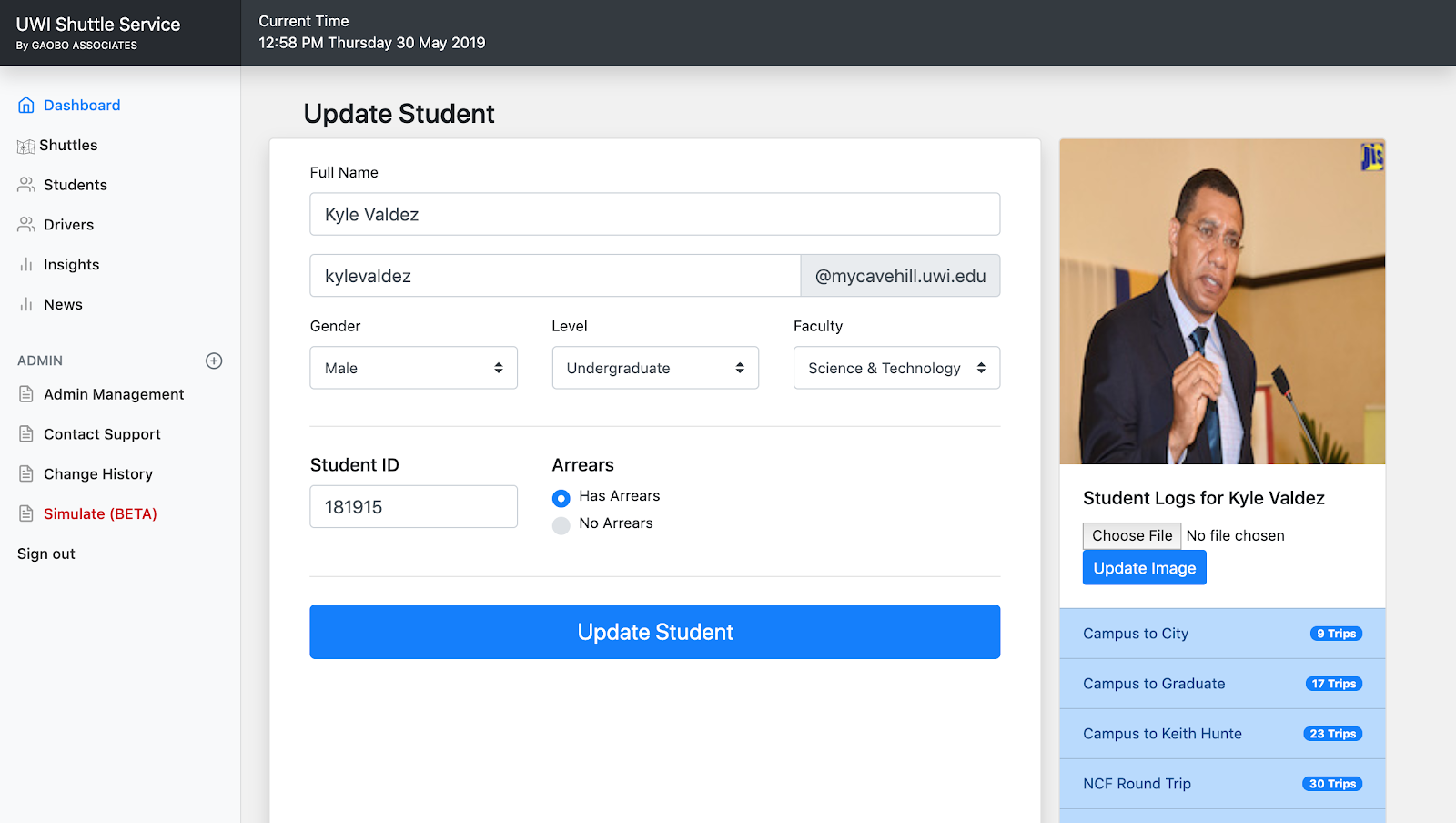
View Shuttles Interface



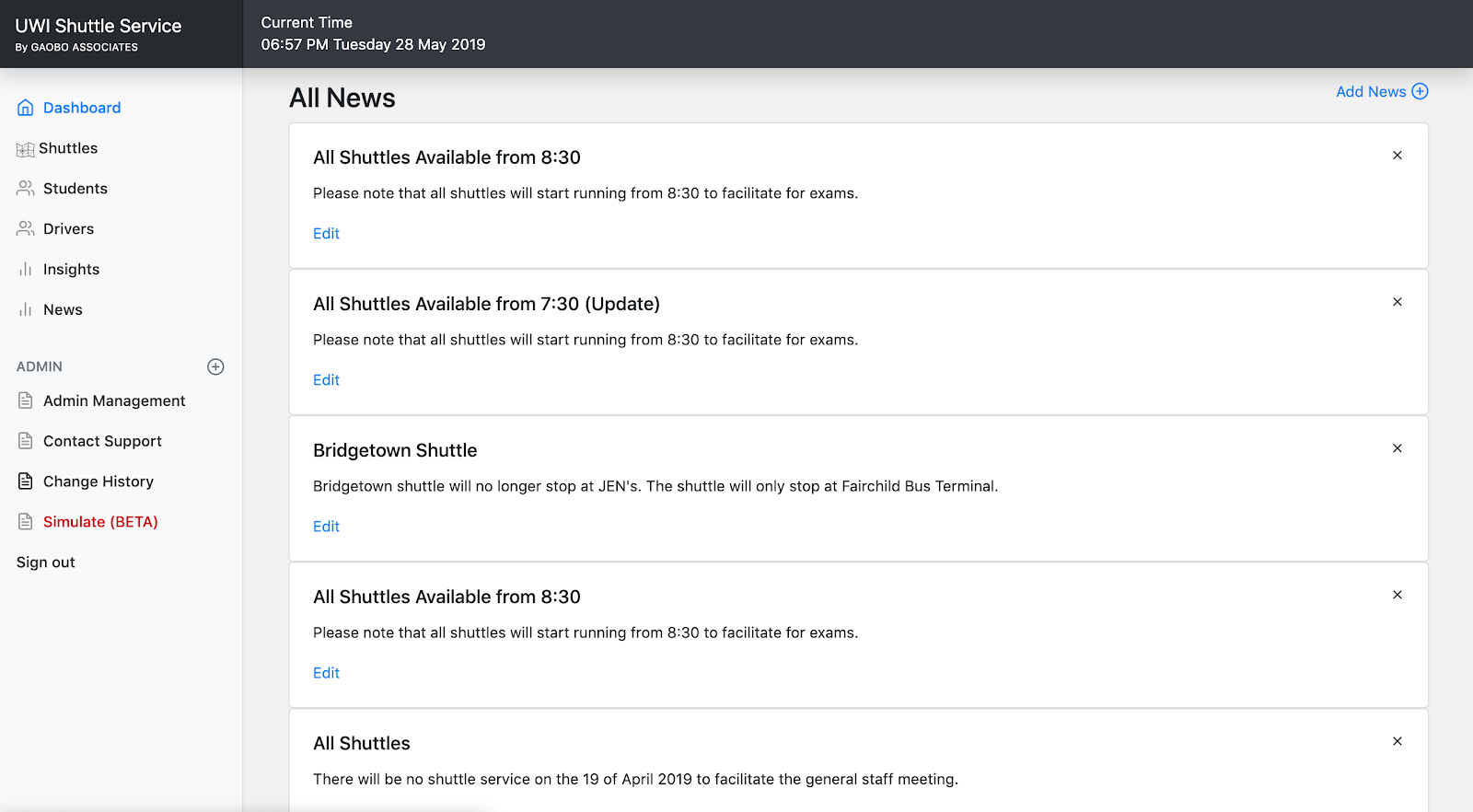
Shuttle Edit Interface with Google Maps Integration



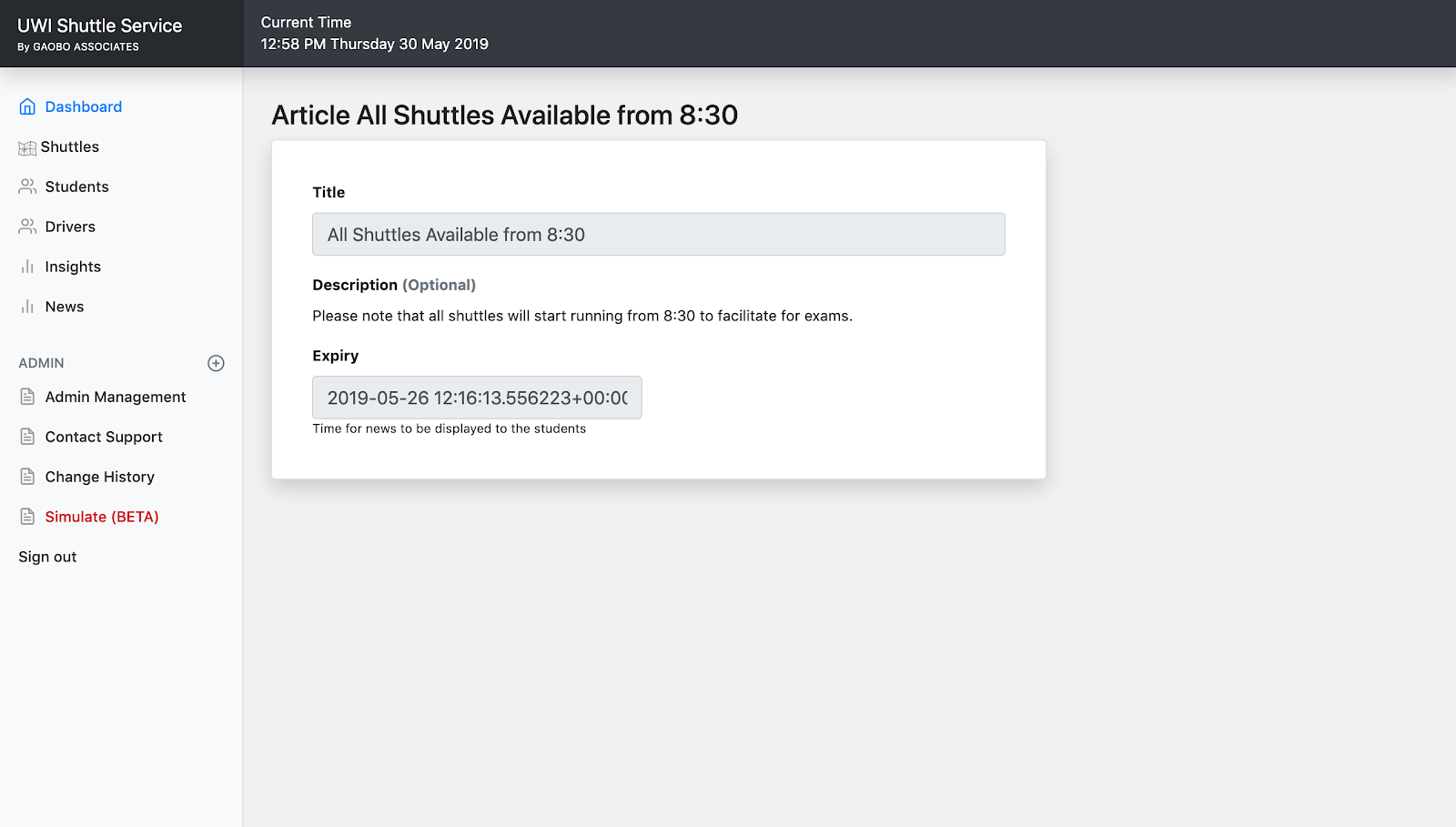
Student Listing with Pagination, Quick Search and Filtering



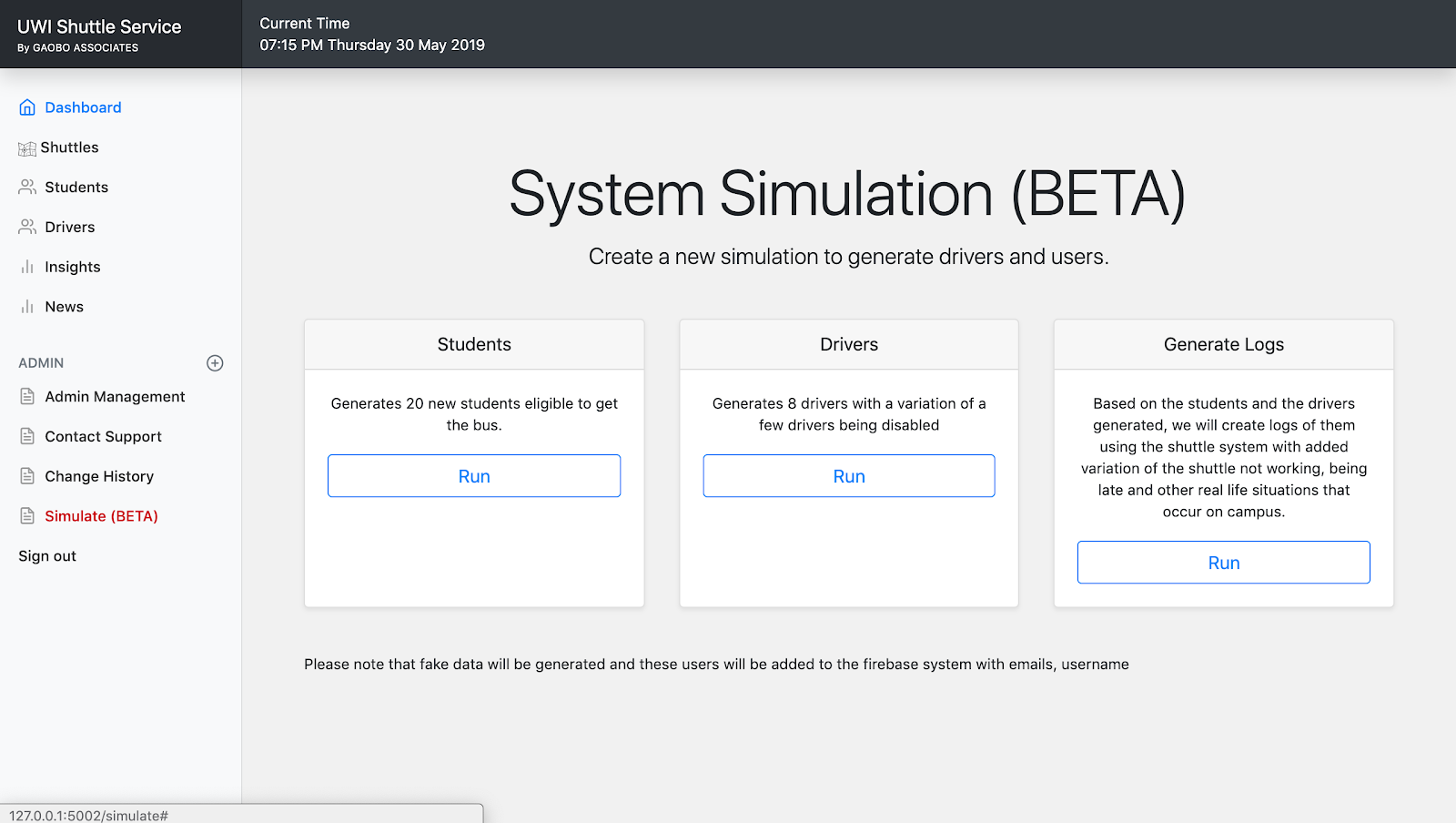
Student Update Profile Interface with Image Upload



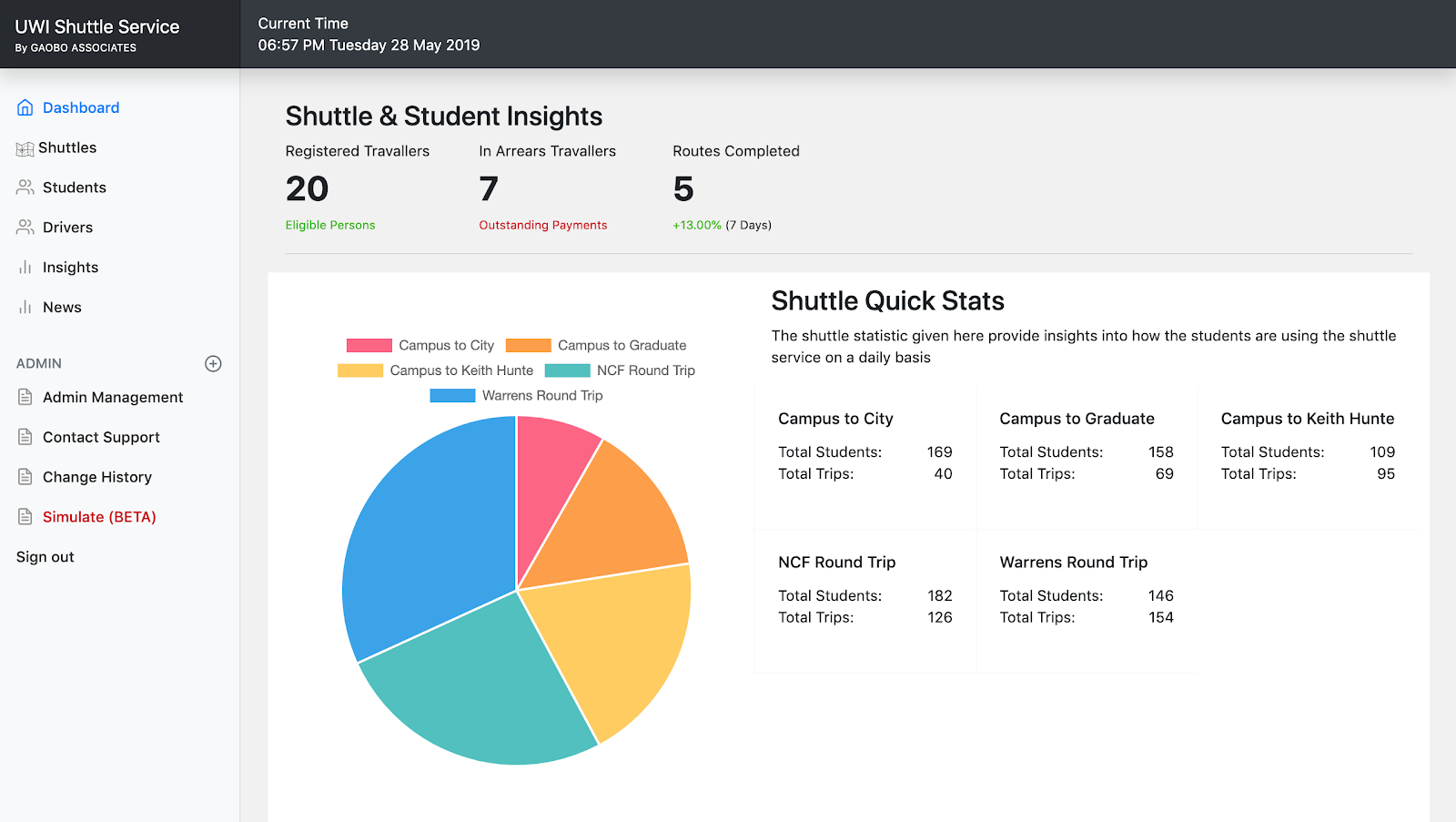
View all News with API Search



View News Information



System Simulation



Shuttle Service Insights Interface

## Hardware Interfaces

A browser which supports HTML & Javascript.

Stable internet connection

## Software Interfaces

The web application makes use of a lot of components in order to achieve various functionalities that are available to the user and make executing task much easier.

The system is built primarily using Flask from Python. The system also makes use of

**Javascript Libraries**

Chart JS

JQuery

**Python Add-ons**

Google Translate

Python Faker

Firebase

Moment

One Signal

Flask Mail

pyfcm

Python Dotenv

Pyrebase

Flask WTForms

Other components used mainly for the styling and templating were:

CSS 3

HTML

Server - NGINX & APACHE

SSL - Let’s Encrypt

## Communications Interfaces

This project supports all types of web browsers. We are using simple forms for the shuttle, student and drivers forms, javascript is used in minimal sections of the application.

# System Features

## View Shuttles

***Requirement #: 1***

**Use Case:** *View Shuttles*

**Rationale:** *This allows authorised admin users to view all of the routes available.*

**Description (***User Requirement***):** *The application shall show the user all routes in the system*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *Shuttles are available*

**Priority:** *High*

**Owner:** *User*

## Edit Shuttles

***Requirement #: 2***

**Use Case:** *Edit Shuttles*

**Rationale:** *This allows authorised admin users to edit shuttle information such as name, description, notes, geo coordinates, shuttle schedule & times, intervals and breaks.*

**Description (***User Requirement***):** *The application shall allow the admin user to update any shuttles information.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *None*

**Priority:** *High*

## Delete Shuttles

***Requirement #: 3***

**Use Case:** *Delete Shuttles*

**Rationale:** *This allows authorised admin users to remove a shuttle fro m current to an archives list which is only available to the developers with access to the Firebase Database*

**Description (***User Requirement***):** *The application shall allow the admin user to remove any shuttle.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *Shuttle Log*

**Priority:** *High*

## Add Shuttles

***Requirement #: 4***

**Use Case:** *Add Shuttles*

**Rationale:** *This allows authorised admin users to edit shuttle information such as name, description, notes, geo coordinates, shuttle schedule & times, intervals and breaks.*

**Description (***User Requirement***):** *The application shall allow the admin user to update any shuttles information.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *None*

**Priority:** *High*

## View Students

***Requirement #: 5***

**Use Case:** *View Students*

**Rationale:** *This allows authorised admin users to view all of the students available, search through students, sort by id, name or level with the option of pagination.*

**Description (***User Requirement***):** *The application shall allow the admin user to view all students allowed to use the system*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised and Students are available*

**Relates to/Dependencies:** *None*

**Priority:** *High*

**Owner:** *User*

## Edit Students

***Requirement #: 6***

**Use Case:** *Edit Students*

**Rationale:** *This allows authorised admin users to edit students information such as name, email, gender, level, faculty, student ID, arrears, student image. It also shows what routes the student has being utilising and how often.*

**Description (***User Requirement***):** *The application shall allow the admin user to update any student information.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *None*

**Priority:** *High*

**Owner:** *Administrator*

## Delete Students

***Requirement #: 7***

**Use Case:** *Delete Students*

**Rationale:** *This allows authorised admin users to remove a student. This removes them from the current list and moves them to an archives section that only developers with access to the Firebase database can view.*

**Description (***User Requirement***):** *The application shall allow the admin user to remove a student.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *None*

**Priority:** *High*

**Owner:** *Administrator*

## Add Students

***Requirement #: 8***

**Use Case:** *Add Students*

**Rationale:** *This allows authorised admin users to add students information with the given information such as name, email, gender, level, faculty, student ID, arrears, student image.*

**Description (***User Requirement***):** *The application shall allow the admin user to add any student information.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *None*

**Priority:** *High*

**Owner:** *Administrator*

## View Drivers

***Requirement #: 9***

**Use Case:** *View Drivers*

**Rationale:** *This allows authorised admin users to view a list of all drivers and their status.*

**Description (***User Requirement***):** *The application shall allow the admin user to view all drivers.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *Drivers are available*

**Priority:** *Medium*

**Owner:** *Administrator*

## Add Driver

***Requirement #: 10***

**Use Case:** *Add Drivers*

**Rationale:** *This allows authorised admin users to add a new user to the Firebase Users which will then be categorised in the database as a driver. This driver will then have access to the Shuttle Android App.*

**Description (***User Requirement***):** *The application shall allow the admin user to create a new driver user.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *None*

**Priority:** *Medium*

**Owner:** *Administrator*

## Enable / Disable Drivers

***Requirement #: 11***

**Use Case:** *Enable / Disable Drivers*

**Rationale:** *This allows authorised admin users to allow or disallow drivers.*

**Description (***User Requirement***):** *The application shall allow the admin user to allow or disallow drivers.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *None*

**Priority:** *Medium*

**Owner:** *Administrator*

## Insights

***Requirement #: 12***

**Use Case:** *View Insights*

**Rationale:** *This allows authorised admin users to view insights on the data collected by the system.*

**Description (***User Requirement***):** *The application shall allow the admin user to view all students allowed to use the system*

**Details (***System Requirements***):**

**Acceptance Criteria:** *There is at least a months worth of data on students and shuttles to allow the insights to show something meaningful.*

**Relates to/Dependencies:** *Simulate*

**Priority:** *High*

**Owner:** *Administrator / Developers*

## Manage Admin Users

***Requirement #: 13***

**Use Case:** *View list of authorised users*

**Rationale:** *This allows authorised admin users to view what other users are authorised to use the system.*

**Description (***User Requirement***):** *The application shall allow the admin user to view all users allowed to use the system*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *None*

**Priority:** *High*

**Owner:** *Administrator*

## Contact Support

***Requirement #: 14***

**Use Case:** *Get in contact with Support*

**Rationale:** *This allows authorised admin users to send queries to support*

**Description (***User Requirement***):** *The application shall allow the admin user to send a message via Gmail SMTP.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *Gmail API*

**Priority:** *Low*

**Owner:** *Administrator / Developers*

## Admin Log

***Requirement #: 15***

**Use Case:** *View Admin Logs*

**Rationale:** *This allows authorised admin users to view a log of every task that has been executed in the system.*

**Description (***User Requirement***):** *The application shall allow the admin user to view all students allowed to use the system*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *None*

**Priority:** *Low*

**Owner:** *Administrator*

## Simulation

***Requirement #: 16***

**Use Case:** *Run Simulation*

**Rationale:** *This allows authorised admin users to generate realistic data for students, shuttles and drivers in order to populate the Insights section.*

**Description (***User Requirement***):** *The application shall allow the admin user to generate data for students, drivers and shuttles.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *None*

**Priority:** *Medium*

**Owner:** *Administrator / Developers*

## View News

***Requirement #: 17***

**Use Case:** *View News*

**Rationale:** *This allows authorised admin users to view all of the news available in the system and search through news articles.*

**Description (***User Requirement***):** *The application shall show the admin user all news in the system*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised and Students are available*

**Relates to/Dependencies:** *None*

**Priority:** *High*

**Owner:** *User*

## Add News

***Requirement #: 18***

**Use Case:** *View Students*

**Rationale:** *This allows authorised admin users to add new news article to the database with a title, description and an expiry.*

**Description (***User Requirement***):** *The application shall allow the admin user to create a new news article.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised*

**Relates to/Dependencies:** *None*

**Priority:** *High*

**Owner:** *User*

## Translate News

***Requirement #: 19***

**Use Case:** *Translate*

**Rationale:** *The system translates the users news article into multiple languages automatically such as French & Chinese.*

**Description (***User Requirement***):** *The application shall translate all news articles that the admin enters into the system.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised and news article is created in English*

**Relates to/Dependencies:** *News Article Entry*

**Priority:** *High*

**Owner:** *Admin / Developers*

## Translate Shuttle Stands

***Requirement #: 20***

**Use Case:** *Translate*

**Rationale:** *The system translates the shuttle stand entries into multiple languages automatically such as French & Chinese.*

**Description (***User Requirement***):** *The application shall translate all shuttles stands that the admin enters into the system.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised and shuttle stand is created in English*

**Relates to/Dependencies:** *Shuttle Stand Entry*

**Priority:** *High*

**Owner:** *Admin / Developers*

## Push Notifications

***Requirement #: 21***

**Use Case:** *Push Notifications*

**Rationale:** *The system sends out a push notification using the News Article as the data for the notification and One Signal as the intermediary to connect the notification to the iOS or android device which should have the UWI Shuttle App installed.*

**Description (***User Requirement***):** *The application shall allow the admin user to send out a push notification when a news article is created.*

**Details (***System Requirements***):**

**Acceptance Criteria:** *User is authorised and shuttle stand is created in English*

**Relates to/Dependencies:** *New News Article, One Signal Third Party Plugin, UWI Shuttle App*

**Priority:** *High*

**Owner:** *Admin / Developers*

# Other Nonfunctional Requirements

## Performance Requirements

The system should respond to the user within 5 seconds after an initial click given that the admin is using the recommended internet speed. The interface should let the user know at all times what is happening in the event that an action is being performed or is triggered.

## Safety Requirements

This system holds a lot of important data that is based on real time events, every day we will run backups that will allow us to keep a copy of the data so in the event that a fatal crash occurs, data can be restored immediately as the problem is investigated and rectified.

## Security Requirements

Security is very important to this system as information about individuals are stored, this information can also be used to identify a student or a driver and the routes that they often travel. One of the steps to be developed into the application would be to fully encrypt the information being saved and received by the system. As a result, if a break occurs the data would be worthless.

## Software Quality Attributes

**AVAILABILITY:** The admin user should be able to access the data without internet connectivity since the system should store the data locally and listen for updates when new data is available.

**CORRECTNESS:** All shuttle log information should be displayed to the user with specific characteristics such as a start time, driver, student count and students left.

## Business Rules

They are no business as yet as the system is a prototype, we would need to learn a bit more about the daily roles of the individuals managing the system.

# Other Requirements

**Setting up the server with a domain and SSL**

**Step 1: Acquire Server from Digital Ocean**

* Ubuntu 18.04.1 x64
* Size: Memory 1G, SSD 25 GB, Transfer 1 TB
* Datacenter Region: Singapore
* Additional Options: None
* SSH keys: Created keys and uploaded them to Digital Ocean

Create a non-root sudo user

Creating the user was based on the [Digital Ocean Initial Server Setup Tutorial](https://www.digitalocean.com/community/tutorials/initial-server-setup-with-ubuntu-18-04).

adduser uwishuttle

usermod -aG sudo uwishuttle

ufw allow OpenSSH

ufw enable

Copy SSH keys to the non-root sudo user

rsync --archive --chown=uwishuttle:uwishuttle ~/.ssh /home/uwishuttle

Acquire & Setup the domain

For this we used Godaddy and then configure the Name Servers to point to Digital Ocean like so.

Link domain to IP address in Digital Ocean.

So linking in DO is pretty easy, once you go to domains, enter the name of the domain you just created and then attach it to the droplet ( virtual Server we just acquired ). You can also setup cnames, mx records and a records.

Install packages

Before the single page flask app can be built, a number of packages need to be installed on the server. We logged onto the server with our ssh keys and installed the following packages:

sudo apt-get update

sudo apt-get upgrade

sudo apt-get install python3-pip

sudo apt-get install python3-dev

sudo apt-get install python3-setuptools

sudo apt-get install python3-venv

sudo apt-get install build-essential libssl-dev libffi-dev

Create a virtual environment and install flask

Similar to developing on a local system, we also setup a virtual environment.

cd ~

mkdir flaskapp

cd flaskapp

python3.6 -m venv flaskappenv

source flaskappenv/bin/activate

(flaskappenv) pip install wheel

(flaskappenv) pip install flask

(flaskappenv) pip install uwsgi

(flaskappenv) pip install requests

(flaskappenv) nano flaskapp.py

At this point we are following a tutorial and we just want the setup to work so instead of uploading all of the files, we only upload one python file that is really simple

*# flaskapp.py*

**from** flask **import** Flask

app = Flask(\_\_name\_\_)

@app.route("/")

**def** index():

**return** "<h1>The temperature is 91.2 F</h1>"

**if** \_\_name\_\_ == "\_\_main\_\_":

app.run(host='0.0.0.0')

Testing our first simple flask app

So with the domain and the server up as well as the packages installed we tried to run our app on port 5000 but first we have to make it accessible.

(flaskappenv)sudo ufw allow 5000

(flaskappenv)python flaskapp.py

Set up uWSGI and systemctl

There are going to be two layers between the flask app and the outside internet. Get requests from web browsers will first come into NGINX then go to uWSGI before being passed to flask.

Configuring our uWSGI

We installed uWSGI earlier when we pip installed flask. Now uWSGI needs to be configured and tested. The following tutorial was used to set this up - [Digital Ocean tutorial](https://www.digitalocean.com/community/tutorials/initial-server-setup-with-ubuntu-18-04).

(flaskappenv)$ pwd

*# ~/flaskapp*

(flaskappenv)$ nano wsgi.py

In the *wsgi.py* file, include:

*# wsgi.py*

**from** flaskapp **import** app

**if** \_\_name\_\_ == "\_\_main\_\_":

app.run()

Testing uWSGI

Next, we test the configuration and ran the uWSGI from the command line with a couple flags:

(flaskappenv)$ uwsgi --socket 128.199.157.202:5000 --protocol=http -w wsgi:app

When we point a browser to the droplet IP address followed by :5000, this flask app showed up, so everything is working so far.

Construct the uWSGI configuration file

Now for another layer of uWSGI goodness- building a uWSGI *.ini* configuration file.

(flaskappenv)$ deactivate

pwd

*# ~/flaskapp*

nano flaskapp.ini

Inside the *flaskapp.ini* file, we included the following:

[uwsgi]

module = wsgi:app

master = true

processes = 5

socket = flaskapp.sock

chmod-socket = 660

vacuum = true

die-on-term = true

Constructing a systemd file

So of course we want to have the flask app running all the time, so we needed to create a systemd control file to get the flask app running as a system service on the server.

sudo nano /etc/systemd/system/flaskapp.service

[Unit]

Description=uWSGI instance to serve flaskapp

After=network.target

[Service]

User=uwishuttle

Group=www-data

WorkingDirectory=/home/uwishuttle/flaskapp

Environment="PATH=/home/uwishuttle/flaskapp/flaskappenv/bin"

ExecStart=/home/uwishuttle/flaskapp/flaskappenv/bin/uwsgi --ini flaskapp.ini

[Install]

WantedBy=multi-user.target

Test with systemctl

After the *flaskapp.service* file is created, we had to reload the systemctl daemon before starting the flaskapp service.

sudo systemctl daemon-reload

sudo systemctl start flaskapp

sudo systemctl status flaskapp

One the status call displayed the service as active (running). We knew everything went well config wise.

flaskapp.service - uWSGI instance to serve flaskapp

Loaded: loaded (/etc/systemd/system/flaskapp.service; disabled; vendor preset

Active: active (running) since Wed 2019-05-05 18:09:15 UTC; 7s ago

Configure NGINX and apply SSL security

We'll use NGINX as a proxy server to work with uWSGI and the flask app. The general control flow resulting from GET request will be:

GET request → NGINX → uWSGI → flaskapp

Install NGINX

Before we can use NGINX, NGINX needs to be installed on the server.

sudo apt-get install nginx

Configure NGINX

sudo nano /etc/nginx/sites-available/flaskapp

server {

listen 80;

server\_name gaoboassociates.club wwww.gaoboassociates.club;

location / {

include uwsgi\_params;

uwsgi\_pass unix:/home/uwishuttle/flaskapp/flaskapp.sock;

}

}

Next we link the NGINX config file to the /etc/nginx/sites-enabled directory and restart NGINX with the new configuration.

sudo ln -s /etc/nginx/sites-available/flaskapp /etc/nginx/sites-enabled

sudo systemctl restart nginx

sudo systemctl status nginx

Since NGINX and uWSGI are running, we can shut off the :5000 development port.

sudo ufw delete allow 5000

sudo ufw allow 'Nginx Full’

When we browse now to our domain below, we can view our flask app without the :5000 port. BOOM!

http://gaoboassociates.club/

Apply SSL Security

During the semester we learnt of HTTPS using Lets Encrypt so we wanted to add that onto this project.

$ sudo add-apt-repository ppa:certbot/certbot

$ sudo apt install python-certbot-nginx

$ sudo certbot --nginx -d gaoboassociates.club -d www.gaoboassociates.club

As part of the certbot setup, we selected option 2.

2: Redirect - Make all requests redirect to secure HTTPS access. Choose this for

new sites, or if you're confident your site works on HTTPS. You can undo this

change by editing your web server's configuration.

IMPORTANT NOTES:

- Congratulations! Your certificate **and** **chain** have been saved **at**:

/etc/letsencrypt/live/gaoboassociates.club/fullchain.pem

Your **key** file has been saved **at**:

/etc/letsencrypt/live/gaoboassociates.club/privkey.pem

```

```bash

$ sudo ufw **delete** allow 'Nginx Full'

$ sudo ufw allow 'Nginx HTTPS'

Now we test the application again. Once everything came up good without the port and with HTTPS, we began to install the following packages that would be used for our real project.

pip install Faker

pip install firebase-admin

pip install moment

pip install translate

pip install onesignal-sdk

pip install Flask-Mail

pip install pyfcm

pip install python-dotenv

pip install googletrans

pip install Pyrebase

pip install Flask-WTF

Afterwards we moved our entire file directory and let the debugging begin.