**User Manual for Running TicketGrab Flask Application**

**1. Introduction**

**Purpose of the Document**: This user manual is designed to provide guidance on setting up and running our flask web application. This guide documents the step-by-step instructions on how to install, configure and operate our web application.

**Intended Audience**: This document assumes that the audience are tech-savvy users who have a basic understanding of python programming, web application architecture and fundamental command-line operations.

**2. System Requirements**

This section outlines the specific software prerequisites necessary to successfully deploy and run the Flask application. Ensure that your system meets these requirements to facilitate a smooth setup and operation.

**Operating System:** The Flask application is platform-independent and can be run on any of the following operating systems:

* **Windows 10** or later
* **macOS Mojave (10.14)** or later
* **Linux** (Ubuntu 20.04 LTS, CentOS 8, or any other major distribution)

**Python:** Python 3.8 or newer must be installed on your system. Python is the primary programming language used for the Flask application. You can download it from [python.org](https://www.python.org/downloads/).

**Flask:** The application is built using the Flask web framework. Flask can be installed via pip through this command:

pip install Flask

**Database:**

PostgreSQL (for production environments): Recommended for production deployments for its robustness and scalability. Ensure that PostgreSQL is installed and configured on your system. Details on installation can be found on the [official PostgreSQL website](https://www.postgresql.org/download/).

pgAdmin**:** pgAdmin 4 is recommended for download should you prefer to access our PostgreSQL databases through a graphical interface. You can download it from pgAdmin.org if it is not already installed with PostgreSQL.

**Other Dependencies:**

* **pip**: Ensure that pip is installed for managing Python packages. It usually comes with Python installation.
* **virtualenv** (optional but recommended): Used for creating isolated Python environments. Install via pip:

pip install virtualenv

**Additional Python Packages:** The application requires additional Python libraries, which can be installed using pip as specified in the provided requirements.txt file:

pip install -r requirements.txt

**3. Configuration**

Connecting to our cloud database server:

Since the database used in this web application is already hosted on cloud, no set up is required for the database for the web application to function.

To connect into our PostgreSQL database server, you may follow the instructions below:

1. Start pgadmin 4:
2. Under object explorer, right click server and register a new server.

A screenshot of a computer

Description automatically generated

1. Enter the following details:

Name: ticketgradb

A screenshot of a computer

Description automatically generated

Host name/address: ticketgrabdb.cugroa0wbny6.us-east-1.rds.amazonaws.com

Username: DBProjGrp3

Password: strawberries

A screenshot of a computer

Description automatically generated

1. Upon successful connection, ticketgrabdb will be accessible from the Object Explorer.

A screenshot of a computer

Description automatically generated

Using PostgreSQL on your local host:

Should you prefer to set up the database on your own local host, simply create a new database under pgadmin, and modify SQLALCHEMY\_DATABASE\_URI in config.py in the project directory.

A computer screen shot of a black screen

Description automatically generated

You may modify the URI as such:  
SQLALCHEMY\_DATABASE\_URI = 'postgresql:// username:password@ host:port / database

For example, if your username is postgres and you running on local host port 5432, and the name of the database you created is TicketGrabdb, then the following will be the URI:

SQLALCHEMY\_DATABASE\_URI = 'postgresql://postgres:password@localhost:5432/TicketGrabdb'

Afterwards, simply run the application with the following instructions and the script will handle the creation and population of the database tables.

**4. Running the Application**

To start the Flask application server, you will need to execute a command in the terminal or command prompt. Make sure you are in the project’s root directory where the application files are located.

You can start the server using this command: flask run

**Accessing the Application:**

Once the server is running, you can access the application through a web browser by visiting http://127.0.0.1:5000 or http://localhost:5000 in your web browser. This is the default address and port used by Flask.

1. **Walkthrough**

This section provides a more personalized guide to run the set up and run the application based on the developer’s standard procedures.

**10. Contact Information**

* How to reach support or get help with issues not covered in the manual.