Theft Alert System for Supermarkets

Deployment Manual

Secure View Theft Alert System

1. Introduction

1.1 Purpose of this Manual

This manual provides step-by-step guidance for deploying the CCTV Security Platform on a production server. It is designed for system administrators responsible for ensuring the platform is securely and efficiently set up.

1.2 System Overview

The CCTV Security Platform enables real-time video streaming, storage, and monitoring of surveillance footage. Key features include:

Scalable and robust architecture to handle multiple video sources. User management through a secure web interface.

Integrated logging and monitoring for system health and security.

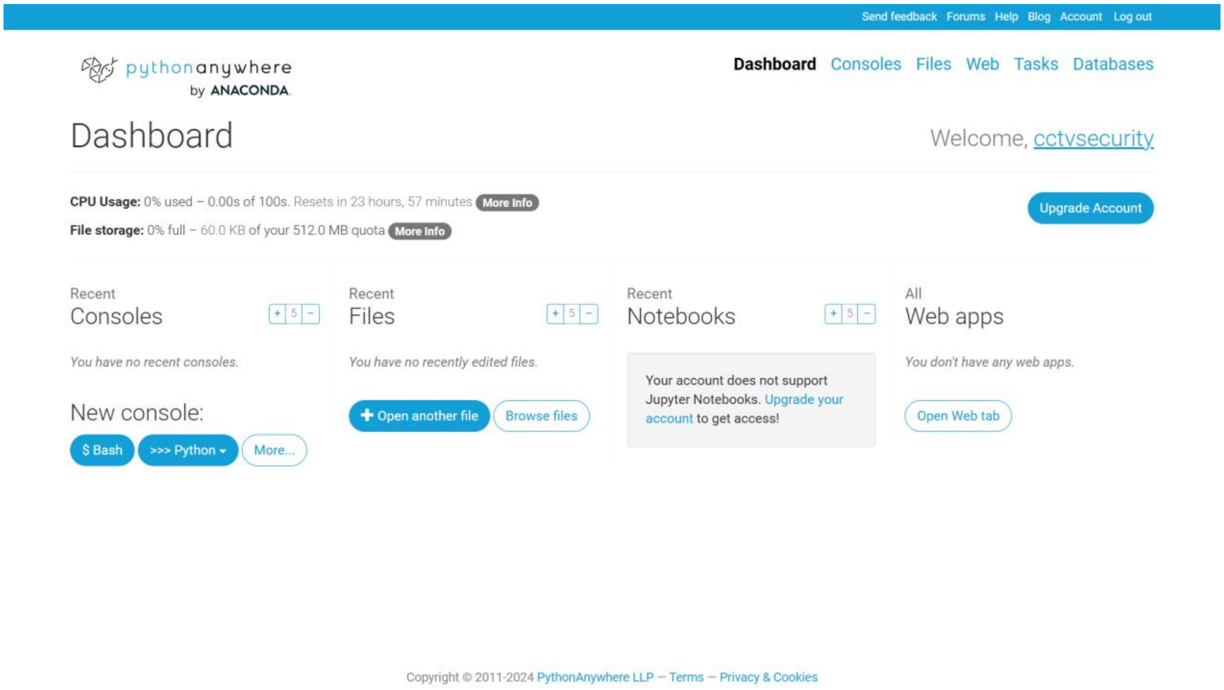
SSL/TLS encryption for secure communication.

2. Prerequisites

2.1 Hardware Requirements

Ensure the following hardware specifications for optimal performance:

- **Server/Host Device**: PythonAnywhere provides the hosting environment, so no local server hardware is needed.
- **Client Devices**: Any device capable of running a modern web browser (e.g., Chrome, Firefox).
- **Storage**: At least 500MB of disk space for the application and logs (included in PythonAnywhere plans).
- **Memory**: PythonAnywhere plans include sufficient RAM for most Django applications. For heavy traffic, ensure your plan supports additional worker processes.



Dashboard

Welcome, cctvsecurity

CPU Usage: 0% used – 0.00s of 100s. Resets in 23 hours, 57 minutes

File storage: 0% full – 60.0 KB of your 512.0 MB quota

Recent Consoles

You have no recent consoles.

New console:

Recent Files

You have no recently edited files.

Recent Notebooks

Your account does not support Jupyter Notebooks. Upgrade your account to get access!

All Web apps

You don't have any web apps.

Copyright © 2011-2024 PythonAnywhere LLP — Terms — Privacy & Cookies

2.2 Software Requirements

PythonAnywhere already includes most necessary software. Confirm the following versions:

- **Python**: Python 3.8 or later (PythonAnywhere supports multiple versions).
- **Database**: MySQL (if required; PythonAnywhere provides hosted MySQL).
- **Django**: Ensure compatibility with the version specified in requirements.txt.

To install additional dependencies:

```
pip install -r requirements.txt
```

2.3 Network Requirements

Ensure the following network conditions:

- **Internet Access**: Stable connection for accessing the PythonAnywhere dashboard and transferring files.
- **Ports**:
  - °PythonAnywhere handles standard web ports (e.g., HTTP: 80, HTTPS: 443).
  - °No additional port configuration is required as PythonAnywhere handles web traffic

routing.

3. Preparing the Environment

3.1 Setting Up the Server

PythonAnywhere provides a hosted environment, so there's no need to configure a physical or virtual server. Follow these steps to set up your PythonAnywhere account:

Create an Account

- Visit PythonAnywhere and sign up for an account.
- Choose a plan that your project requires.
- Free plans support basic setups, while paid plans allow for more traffic and resource use.

Access the Dashboard

- Log in and navigate to the dashboard, where you can manage your applications, and databases.

3.2 Installing Required Software

PythonAnywhere provides pre-installed software for Python and Django development. However, you'll need to ensure your application's dependencies are installed:

Upload Your Project

- Use PythonAnywhere’s file manager or a secure file transfer protocol (SFTP) client like FileZilla to upload your project files.

Set Up a Virtual Environment

- Create a virtual environment to isolate your application dependencies:

```
mkvirtualenv --python=/usr/bin/python3.x venv
```

Replace 3.x with the Python version your application requires.

Install Dependencies

- Activate the virtual environment:

```
workon venv
```

- Install dependencies:

```
pip install -r requirements.txt
```

3.3 Configuring the Database

Using SQLite

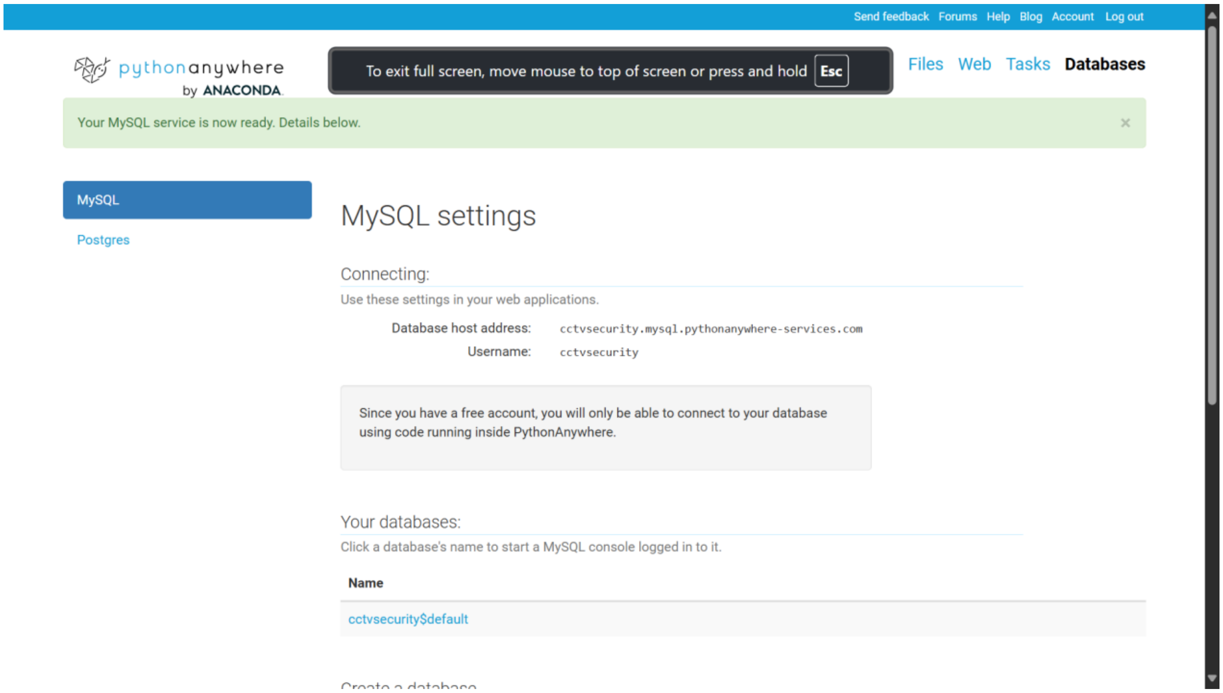
If your project uses SQLite (default for Django), no additional setup is needed. The database will be included in the uploaded project.

Using MySQL

If your project uses MySQL, configure a database on PythonAnywhere:

Go to the **Databases** tab on the PythonAnywhere dashboard.

Create a new MySQL database and note the credentials.



To exit full screen, move mouse to top of screen or press and hold Esc

Your MySQL service is now ready. Details below.

MySQL settings

Connecting:

Use these settings in your web applications.

```
Database host address:   cctvsecurity.mysql.pythonanywhere-services.com
             Username:   cctvsecurity
```

Since you have a free account, you will only be able to connect to your database using code running inside PythonAnywhere.

Your databases:

Click a database's name to start a MySQL console logged in to it.

Name
cctvsecurity$default

Create a database

Update your settings.py with the database connection details:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'cctvsecurity $cctvdb',
        'USER': 'admin11’,
        'PASSWORD': '***************',
        'HOST': 'your_username.mysql.pythonanywhere-services.com',
        'PORT': '3306',
    }
}
```

4. Deploying the Application

4.1 Obtaining the Source Code

Upload the Project Files

- Use PythonAnywhere’s file manager or an SFTP client to upload your project files to a directory, such as ~/mysite.

Cloning from a Repository

- If your project is hosted on GitHub or another Git hosting service, you can clone it directly:

```
git clone https://github.com/KipkorirVictor/Supermaket-Survaillance-System.git
```

4.2 Configure Django Settings

Modify settings.py

- Update the ALLOWED_HOSTS in settings.py:

```
ALLOWED_HOSTS  =  [‘cctvsecurity.pythonanywhere.com’, ‘127.0.0.1’]
```

Set DEBUG to False for a production environment:

```
DEBUG = False
```

Update Database Settings

- Configure the database settings if you're using MySQL (refer to the database section above).

Set Up Environment Variables

- Create a. env file to store sensitive information, such as SECRET_KEY.
- Update settings.py to read from this file (using a library like python-decouple or dotenv).

4.3 Setting Up Static and Media Files

Collect Static Files

- Run the following command to collect all static files into the STATIC_ROOT:

```
python manage.py collectstatic
```

- Confirm that static files are in the correct directory.

Media Files

- Ensure that media files are uploaded and accessible. Specify a directory for media files in your settings.py:

```
MEDIA_URL = '/media/'
MEDIA_ROOT =  '/home/cctvsecurity/mysite/media'
```

4.4 Configuring the Web Server

PythonAnywhere uses a WSGI server instead of consulting Nginx or Apache directly.

Set Up the Web App

Go to the **Web** tab on the PythonAnywhere dashboard.

Click **Add a new web app** and follow the prompts:

- Choose **Manual configuration**.
- Select the Python version that matches your project.

Configure Static and Media Files

- Under the **Static files** section in the Web tab, add entries like:
  - URL /static/ → Directory /home/cctvsecurity/mysite/static
  - URL /media/ → Directory /home/cctvsecurity/mysite/media

4.5 Setting Up WSGI

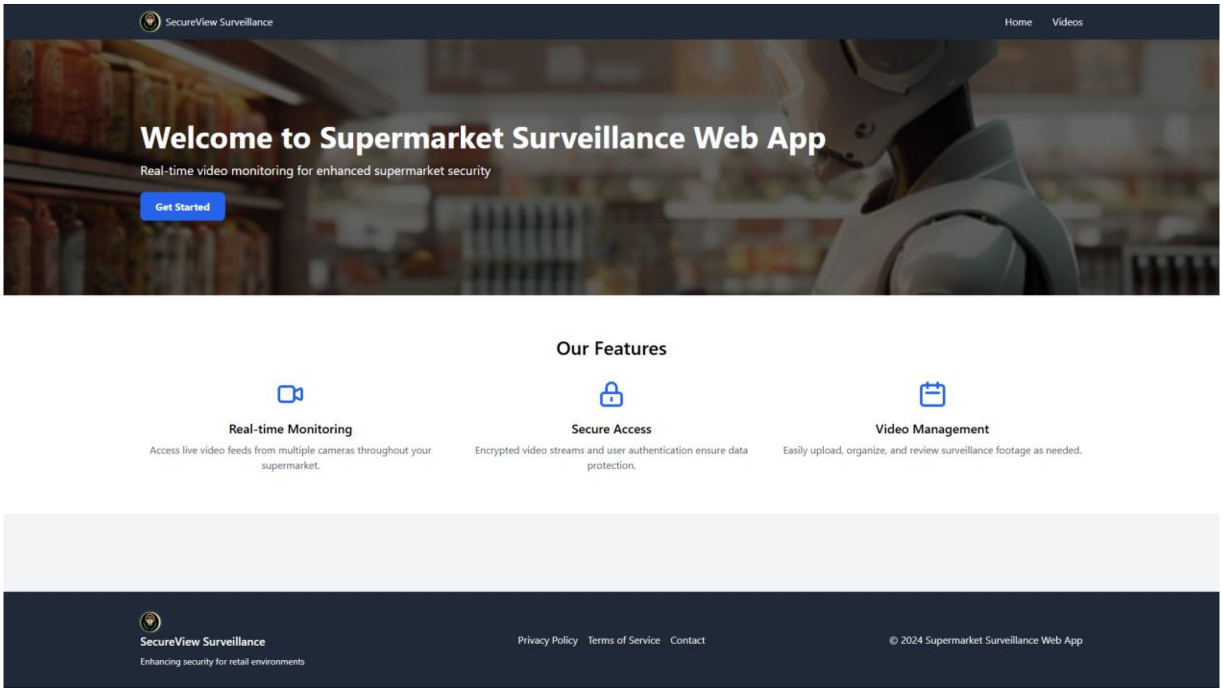
Edit the WSGI File

- Open the WSGI configuration in the Web tab.
- Update it to point to your Django project:

```
import os
import sys

# Adjust paths
path = '/home/cctvsecurity/mysite'
if path not in sys.path:
    sys.path.append(path)

os.environ['DJANGO_SETTINGS_MODULE'] = 'cctv_project.settings'

from django.core.wsgi import get_wsgi_application
application = get_wsgi_application()
```

Reload the Web App

Click **Reload** in the Web tab to apply the changes.



Welcome to Supermarket Surveillance Web App

Real-time video monitoring for enhanced supermarket security

Our Features

Real-time Monitoring

Access live video feeds from multiple cameras throughout your supermarket.

Secure Access

Encrypted video streams and user authentication ensure data protection.

Video Management

Easily upload, organize, and review surveillance footage as needed.

SecureView Surveillance

Enhancing security for retail environments

© 2024 Supermarket Surveillance Web App

5. Database Migration and Initial Setup

5.1 Running Database Migrations

Activate the Virtual Environment

- Ensure you are in your project directory and activate your virtual environment:

Run Migrations

- Apply migrations to set up the database:

```
python manage.py migrate
```

This command creates the required database tables for Django’s built-in apps and any custom apps you’ve added.

5.2 Creating a Superuser

Run the create superuser Command

- Create an admin user to access the Django Admin interface:

```
python manage.py createsuperuser
```

Provide Required Information

- Enter the following details when prompted:
  - Username
  - Email address
  - Password

Verify the Superuser

- Ensure the user is created successfully. You will use this account to log in to the admin panel.

5.3 Initial Data Setup

Load Initial Data (if applicable)

- If your project requires pre-defined data, use Django’s load data command to import textures.

```
python manage.py load data initial_data.json
```

Verify the Data

- Access the Django Admin interface by visiting:

```
https://cctvsecurity.pythonanywhere.com/admin
```

Log in with the superuser credentials and confirm that the necessary data is loaded.