# Campus Asset Tracker

## Python + XAMPP Integration Setup Guide

## Overview

This guide covers everything needed to run Python CGI scripts alongside PHP inside XAMPP. Apache handles both PHP and Python, and both sides share the same MySQL database.

## How It Works

Apache inside XAMPP has a built-in module called mod_cgi. Normally it executes PHP files. With one configuration change it can also execute Python files directly, in the same way, on the same port. There is no second server and no extra process to manage.

The integration model is:

- PHP handles login and writes a session to MySQL
- PHP redirects the browser to the Python script URL
- Apache executes the Python script via CGI
- Python reads from MySQL and renders the HTML response
- Both PHP and Python share the same database on localhost

## Requirements

| Requirement | Details |
|---|---|
| XAMPP | Apache + MySQL modules running |
| Python | 3.10 or higher installed system-wide |
| pip package | mysql-connector-python |
| File location | All files inside ./htdocs/ |

## Step 1 - Install Python System-Wide

Python must be installed at the system level so Apache can find it. A virtual environment alone is not enough because Apache runs as a separate process and cannot activate a venv.

1. Go to https://www.python.org/downloads/ and download Python 3.11.x (I'm running 3.11.9 but should be compatible with other 3.11 versions)

2. Run the installer
3. On the first screen, check the box that says Add Python to PATH - this is critical
4. Complete the installation
5. Open a new Command Prompt and confirm it worked:

```
where python
python --version
```

Both commands should return a path and a version number. If where python returns blank output, Python is not on the system PATH and Apache will not be able to find it.

> *Can ignore this. Note: If you already have Python installed through a virtual environment only, install a full system-wide copy alongside it. The venv can still be used for local development; this system install is specifically for Apache.*

## Step 2 - Install the MySQL Driver

Python needs one package to talk to the XAMPP MySQL database. Install it system-wide (not inside a venv):

```
pip install mysql-connector-python
```

Confirm it installed correctly:

```
pip show mysql-connector-python
```

If you need to uninstall this, run:

```
pip uninstall mysql-connector-python
```

## Step 3 - Configure Apache (httpd.conf)

One line needs to be added to Apache's configuration to tell it to execute .py files as CGI scripts.

6. Open *your_xampp_folder*/apache/conf/httpd.conf in VS Code (Or your editor)
7. Find this line (search for AddHandler cgi-script):

```
AddHandler cgi-script .cgi .pl .asp
```

8. Add .py to the end of that line:

```
AddHandler cgi-script .cgi .pl .asp .py
```

9. Save the file
10. In the XAMPP Control Panel, click Stop then Start on Apache

> *Everything else in httpd.conf is already correctly configured in this project. mod_cgi is loaded, ExecCGI is enabled in the htdocs directory block, and AllowOverride All is set. Only this one line needs to change.*

## Finished. Try running everything and let me know if anything comes up. The following content is for my documentation only.

## Step 4 - File Structure (Skip This - For Documentation Only)

The Python dashboard folder must be placed inside htdocs. The expected structure is:

```
D:/ics499/htdocs/
  index.php                 (PHP login page - PHP team)
  logout.php                (PHP logout - PHP team)
  dashboard/                (Python folder)
    dashboard.py            (main Python CGI script)
    .htaccess               (clean URL routing)
    static/
      dashboard.css         (stylesheet)
```

The .htaccess file in the dashboard folder tells Apache to route requests to dashboard.py. Make sure the file is named exactly .htaccess with the leading dot. Windows may try to save it without the dot - verify in File Explorer that it shows as .htaccess.

## Step 5 - Set the Shebang Line in dashboard.py (Skip This - For Documentation Only)

The first line of dashboard.py tells Apache which Python executable to use. It must point to the system Python, not a venv.

Find your Python path by running this in Command Prompt:

```
where python
```

Then set line 1 of dashboard.py to that path, using forward slashes and pythonw.exe to suppress console popups on Windows:

```
#!C:/Python312/pythonw.exe
```

> *Use pythonw.exe instead of python.exe. The w variant runs Python silently without opening a console window on every request. Both executables are in the same folder.*

## Step 6 - Windows Encoding Fix (Skip This - For Documentation Only)

Windows defaults Python's stdout to cp1252 encoding. Apache expects UTF-8. Without this fix, any non-ASCII character in the HTML output will crash the script with a UnicodeEncodeError.

Add these two lines near the top of dashboard.py, right after import sys:

```
import sys
import io

# Required for Windows CGI - forces UTF-8 output
sys.stdout = io.TextIOWrapper(sys.stdout.buffer, encoding='utf-8')
```

## Step 7 - The .htaccess File

The .htaccess file in the dashboard folder enables CGI execution for that folder and sets up clean URLs. Its contents should be:

```
Options +ExecCGI
AddHandler cgi-script .py
ScriptInterpreterSource Script

RewriteEngine On
RewriteRule ^$ dashboard.py [L,QSA]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^([^.]+)$ $1.py [L,QSA]
```

ScriptInterpreterSource Script tells Apache to use the shebang line on line 1 of the script to find Python, rather than looking in the Windows registry.

## Running the Application

Once setup is complete, the entire stack is started from the XAMPP Control Panel only. There is no separate terminal or Flask server to run.

11. Open the XAMPP Control Panel
12. Click Start on Apache
13. Click Start on MySQL
14. Open a browser and navigate to http://localhost/index.php
15. Log in with admin credentials
16. Apache will execute dashboard.py and serve the dashboard

| URL | What it serves |
|---|---|
| http://localhost/index.php | PHP login page |
| http://localhost/dashboard/ | Python dashboard (clean URL) |
| http://localhost/dashboard/dashboard.py | Python dashboard (direct URL) |

## Troubleshooting

### 500 Internal Server Error - couldn't spawn child process

Apache found the .py file but could not execute it. This means the shebang line on line 1 of dashboard.py points to a path that does not exist. Run where python in Command Prompt and update line 1 of dashboard.py to match exactly.

### 500 Internal Server Error - End of script output before headers

Python ran but crashed before printing any output. Check the Apache error log at D:/ics499/apache/logs/error.log for the specific Python error. The most common cause is the stdout encoding fix is missing (Step 6).

### Index of /dashboard shown instead of the page

Apache is showing a directory listing instead of running the script. This means the .htaccess file is either missing or named incorrectly. Confirm the file exists at D:/ics499/htdocs/dashboard/.htaccess with a leading dot. Also confirm AllowOverride All is set in httpd.conf.

### Page loads but has no styling

The CSS file is not being found. Confirm dashboard.css exists at D:/ics499/htdocs/dashboard/static/dashboard.css. Check the browser DevTools Network tab for the exact error on the CSS request.

### UnicodeEncodeError

The Windows encoding fix is missing or not in the right place. Make sure the sys.stdout line from Step 6 appears before any print() calls in dashboard.py, immediately after import sys.

### Console window flashing on every click

The shebang is pointing to python.exe instead of pythonw.exe. Update line 1 of dashboard.py to use pythonw.exe. Both are in the same Python installation folder.

## Note for PHP Teammates

The only change needed in the PHP codebase to integrate with the Python dashboard is one line in index.php. After a successful admin login, the redirect should point to the Python script instead of the original dashboard.php:

```
// Change this:
header("Location: dashboard.php");

// To this:
header("Location: /dashboard/dashboard.py");
```

No other changes are required in the PHP files. logout.php, session handling, and user validation remain exactly as written. The Python dashboard is self-contained and only needs to receive the browser redirect from index.php.