Raspberry Pi for Beginners - Final Project

Time to start the final project of this course, and practice everything you have seen until now!

First, make sure you have watched the video showing the final result of the project.

Here are some tips to help you get started with the project.

So, you are going to create two different Python programs.

The first Python program will do the following:

- Monitor the PIR sensor. When some movement is detected, make sure the movement is detected for three consecutive seconds. To do that, you can create an infinite while loop. You will have to keep the last PIR sensor state into a variable and compare it with the current state. As an additional visual indicator, you can also power on an LED when the PIR sensor detects some movement.

- At this point, take a photo and save it to the /home/pi/camera folder. Create a function “take_photo()” to take a photo and save it. The file name starts with “img_” and then has a timestamp to make it unique. To get a timestamp, you can call the function “time.time()”. Then call this function in the appropriate place in your main while loop.

- Then, add the name of the photo (complete path + file name) to a log file named photo_logs.txt (use the append mode with “a” when opening the file). During the setup of the program, check whether this file exists and remove it so it doesn’t contain data from previous program runs.

- Then, send an email to yourself with the photo as an attachment. Set up an email (with Yagmail) in the setup section of the program. Create a “send_email_with_photo” function to send the photo that was just taken by email.

- Also, as an additional feature, make sure you don’t take more than one photo (+ write to log a file and send email) every 60 seconds. For that, you can also keep a “last_time_photo_taken” variable with the time.

The second Python program will do the following:

- Run a Flask application with a default route “/”, just returning “Hello”.

- Add another route “/check-movement”.
- When the function for this URL is called, read the log file (photo_logs.txt created by the other Python program, using the read mode “r”). Count the number of lines in the file and get the last photo path (last line). You can use a “for line in f” loop to go through all the lines in the file.
- Then, compare the line count with the previous line count (= the last time you called this URL). For that, you will need to use a global variable to keep track of the count.
- Return a string telling how many new photos have been taken since the last time the URL was called, + give the path to the photo file.

- Finally, use an <img src=”photo_path”> tag inside the string you return to print the image in the browser. To make things work, you will also have to change the line “app = Flask(__name__)” to become “app = Flask(__name__, static_url_path=CAMERA_FOLDER_PATH, static_folder=CAMERA_FOLDER_PATH)”, where CAMERA_FOLDER_PATH is "/home/pi/camera".

Here’s a quick overview of what I’m going to do in each step of the solution. If you are completely stuck, you can always watch the next solution step, and then work by yourself again to complete the next step.

Steps 1-4 are for the first Python program; steps 5-6 for the second Python program.

- **Step1:**Set up the code for the PIR sensor. Detect when there is movement. If there is movement, check that the movement has been detected for three full consecutive seconds. If yes, check that we didn’t take a photo in the last 60 seconds. If yes, then print a message to say we are going to take a photo and send it by email. Also, power on one LED when the PIR sensor detects movement.

- **Step 2:** Set up the camera before the main while loop. Create the function to take a photo and save it into a file. Call that function from the main while loop.

- **Step 3:**In the setup section of the program, remove the log file if it exists. Create the function to save new photo paths to the log file, and call that function just after taking a photo.

- **Step 4:**Set up an email with Yagmail. Create the function to send an email with an attachment. Call that function just after updating the log file.

- **Step 5:**Create the Flask application with a default route and “/check-movement” route. Read the log file and return the number of new photos since the last check.

- **Step 6:**Display the file name of the latest photo, and the photo itself, when calling the URL “/check-movement”.

- **Step 7 (Bonus)**: Start the two Python programs on boot with systemd, so you don’t need to manually get access to your Pi and start the programs from the terminal.

All right, that’s enough tips for now! Time for you to start working on this project :)

Again, if you need any help, feel free to do the following:

- Re-watch lessons from the previous sections
- Search on Google
- Just take a break and come back later, this is often a good thing to do
- And if completely stuck, watch the next solution step

The solution will help you in three ways: first, if you are really stuck in one part, don’t waste 10 hours being stuck. After some point, it’s best to watch the solution and try to understand it. Second, after you finish the project, you can see how someone else is thinking about the same problem. And third, I will try to give you a solution with clean code and best practices, so you can use those tips to make your programs cleaner and more readable in the future.