ECSE 2920: Design Methodology

Deliverable #2

Group #8

01/14/2026

Part 1: Software Bring-Up

Requirements:

P: Practice Committing Changes, make sure they are showing up on your repository. Each team member must push a test file to show you can do this.

In this section of Deliverable #2, our team focused on introducing basic software functionality via GitHub. GitHub is a web-based git repository that allows for the storage, sharing and collaborating of code, which is perfect for team-based projects.

On January 12, 2026, the team met at the UGA Engineering Center to set up the development environment and discuss workflow expectations. During this meeting, we reviewed how to push to and pull from GitHub, ensuring that all team members understand version control procedures. Each member also introduced themselves and discussed which parts of the project they were interested in working on.

As part of the software bring-up process, team members individually pushed their Hello World scripts to the group GitHub repository. At the time of this deliverable, five out of five members successfully completed this step.

Picture(s)

#Individual commit messages of each team member's first_last.py git push

Key Design Decision(s)

- What did your team clarify about the design?
  The team clarified the GitHub workflow, including how to correctly push and pull files, and confirmed the use of individual Hello World scripts to verify software setup.
- What were the competing choices in the design?
  The team could not decide whether to sequentially push to the main Group 8 repo or to each individually fork the repo to upload our own code, then merge all the branches.
- Ultimately what did your team choose and why?

We chose for each team member to push their own Hello World script to sequentially to make sure we all had our bearings for GitHub, leaving forking individual personal repos for when more complex divided work necessitated it.

TEST… Test… test

- What aspects of the design need to be tested?
  - Software?
    - How to GitHub clone, push, and pull
  - Hardware?
    - NA for today's work
- Who is responsible for testing?
  Everyone was responsible for pushing to GitHub
  - Tests ran?
    - Cloning, pushing and pulling code from GitHub on terminal
    - Touching and writing python on terminal
  - Conclusions from testing?
    - The GitHub workflow was successfully verified for all team members, and the development environment was confirmed to be working.

summary/part conclusion (make sure you address all parts of the requirements)

The team successfully set up the GitHub repository, verified basic software functionality using Hello World scripts, and ensured that all team members were able to push code correctly. These steps satisfy the requirements of all team members successfully committing changes to the shared GitHub repo.

Part 2: Understanding the PHP & Hardware Setup (Rpi4)

Requirements:

P: Draw a circuit diagram of GPIO2 on the PHP. Explain how it works.

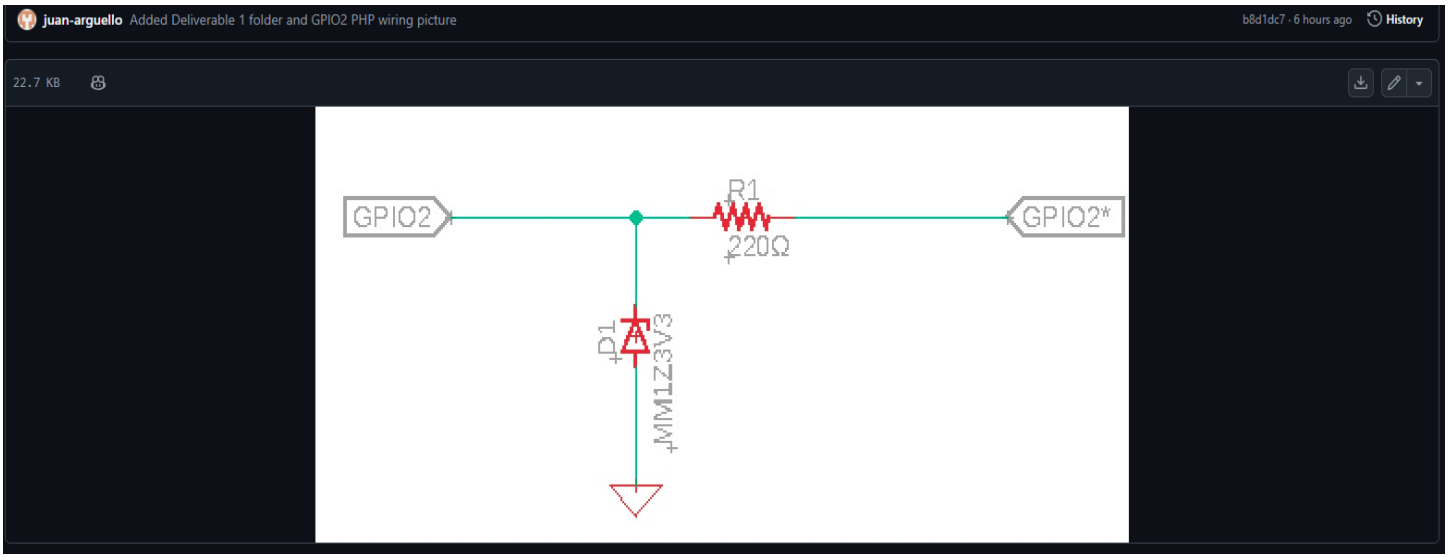
P: Go though the Raspberry Pi setup, and acquaint all members with working on the Pi/Github/etc. Take a photo with "Hello World, from Group XX" with your group Number.

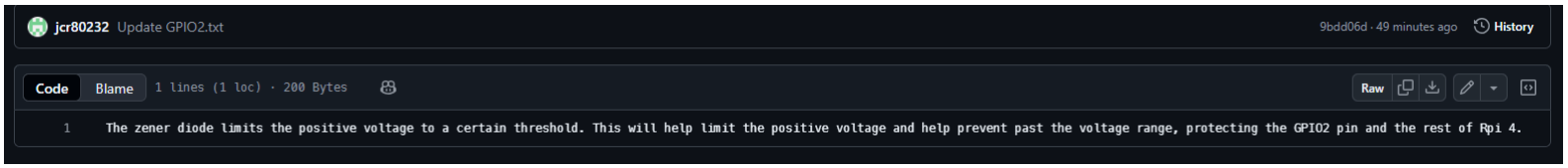
In this section of Deliverable #2, our team focused on understanding the Raspberry Pi 4 Model B and Pi Hat Protector (PHP). For the Rpi4, we focused on software via the Legacy 32-bit OS, creating a virtual environment to run python scripts, and using git to push and pull code from our group 8 GitHub repo.

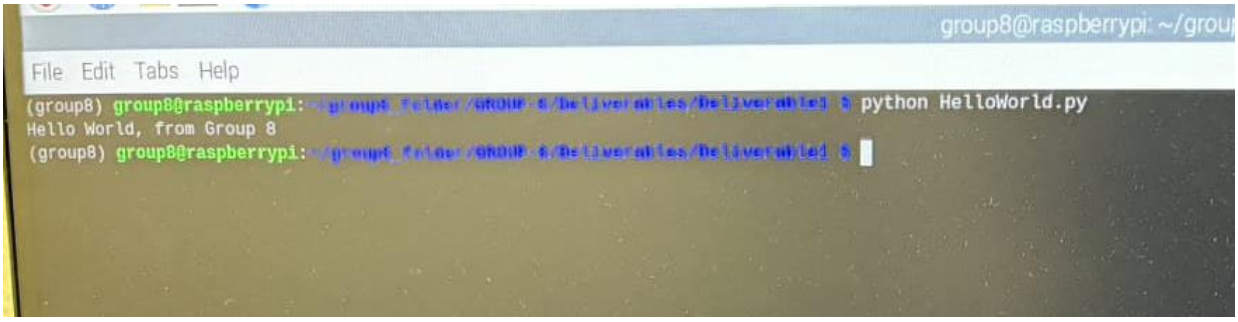
On January 13, 2026, the team met at the UGA Fabrication Lab to understand and set up the Rpi4 as well as claim a locker to store our hardware for the remainder of the semester. During this meeting, we analyzed how the PHP interacted with the GPIO control on the Raspberry Pi platform. GPIO control is a critical foundation for future project components, including sensor input and hardware interaction. Understanding this usage, we found that the PHP connections to GPIO2 (i.e. a resistor and Zener diode in series) acted as a simple voltage regulator for the Rpi4. As listed in our explanation file, the Zener diode maintains a constant voltage, limiting the positive voltage into the Rpi4 to protect the GPIO2 pin as well as the rest of the board. This understanding is included with our diagram of the GPIO2 on the PHP, achieving the objective.

To configure set-up our Rpi4 for the following semester, the team had to configure the Rpi4 from the Legacy 32-bit OS installed the night prior. Although we were given the set-up instructions via “pi_setup.md” under the class repo setup, many of the instructions are now outdated, such as the Rpi4 console option hierarchy and python virtual environment set-up. To overcome this, we sorted through the Rpi4 console to find the options ourselves, as well as reading articles for any problems setting up the virtual environment. After the Rpi4 set-up was complete, we worked through how to launch Python scripts on boot via cron’s timed code execution capabilities, which we were able to access via the cronlog. The photo of the cronlog reveals that we are able to run Python scripts on boot via the Rpi4, finishing the last objective for deliverable 2.

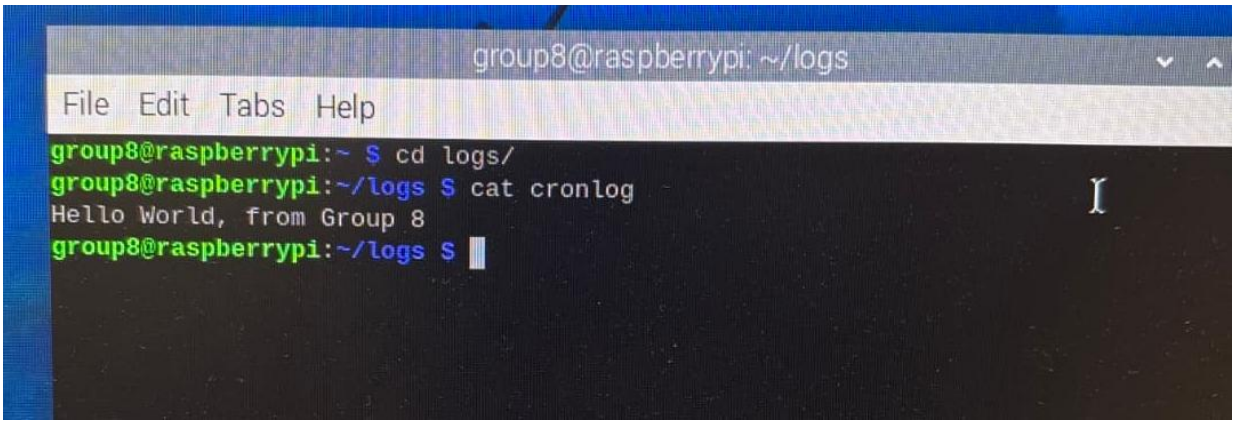
Picture(s)







Testing for Python virtual environment on Rpi4



#cronlog of Rpi running python script on boot

Key Design Decision(s)

- What did your team clarify about the design?
    - How to properly access the option on the Rpi4 config menu
    - How to create a virtual environment on the Rpi4
    - How to find output files from the on boot program on cronlog
- What were the competing choices in the design?
  We were undecided whether or not we should continue to abide by the GitHub instructions and fish for their setting layouts, packages and virtual environments, or to instead adapt the new hardware and use our prior skills to accomplish the deliverables.
- Ultimately what did your team choose and why?
  Initially, we would abide line by line from the provided support material, but when it came up short due to using older models, we used our better judgement to way between our current knowledge, opting to use the instructions as aid rather than law.

TEST… Test… test

- What aspects of the design need to be tested?
    - Software?
        - The Python virtual environment on the Rpi4
        - The launch on boot for code on the Rpi4
        - Hard shit was finding the write numbers for the file path
        - Gets stored in cron log.
    - Hardware?
        - NA for today's work
- Who is responsible for testing?
  All members worked together to troubleshoot issues and read through articles.
    - Tests ran?
        - For the virtual environment, we ran our HelloWord.py to see if it would properly print to terminal.
        - For launch on boot, we rebooted the Rpi4 to see if it would properly print.
    - Conclusions from testing?
        - The print on terminal revealed a successfully created virtual environment.
        - The cronlog revealed a successful launch on boot for the Rpi4.

summary/part conclusion (make sure you address all parts of the requirements)

The team understands the function and purpose of the PHP connections with the GPIOs and successfully set up the Rpi4 for running scripts from our group git repo on boot. These steps satisfy the requirements of diagraming the GPIO2 on the PHP and printing a Hello World from Group 8 on boot from the Rpi4.

Conclusion and Participation (REQUIRED)

1. **Include a selfie/photo from your group meeting/zoom call.**

2. **When did you meet?**
    a. **1/12, 1/13**
3. **Who was present?**
    a. **1/12: All In-Person**
    b. **1/13: 4 In-Person, 1 Virtual**
4. **Who was not present?**
    a. **NA**
5. **What were the main ideas discussed or major decisions (1-2 sentences/bullet points)**
    a. **1/12: The major decisions of our meeting were to begin our project, outline workflow and team roles, and verify the team's ability to commit changes to the Group 8 git repo.**
    b. **1/13: The main ideas discussed during the meeting were assessing how to best set up Rpi4, given the outdated documentation provided.**