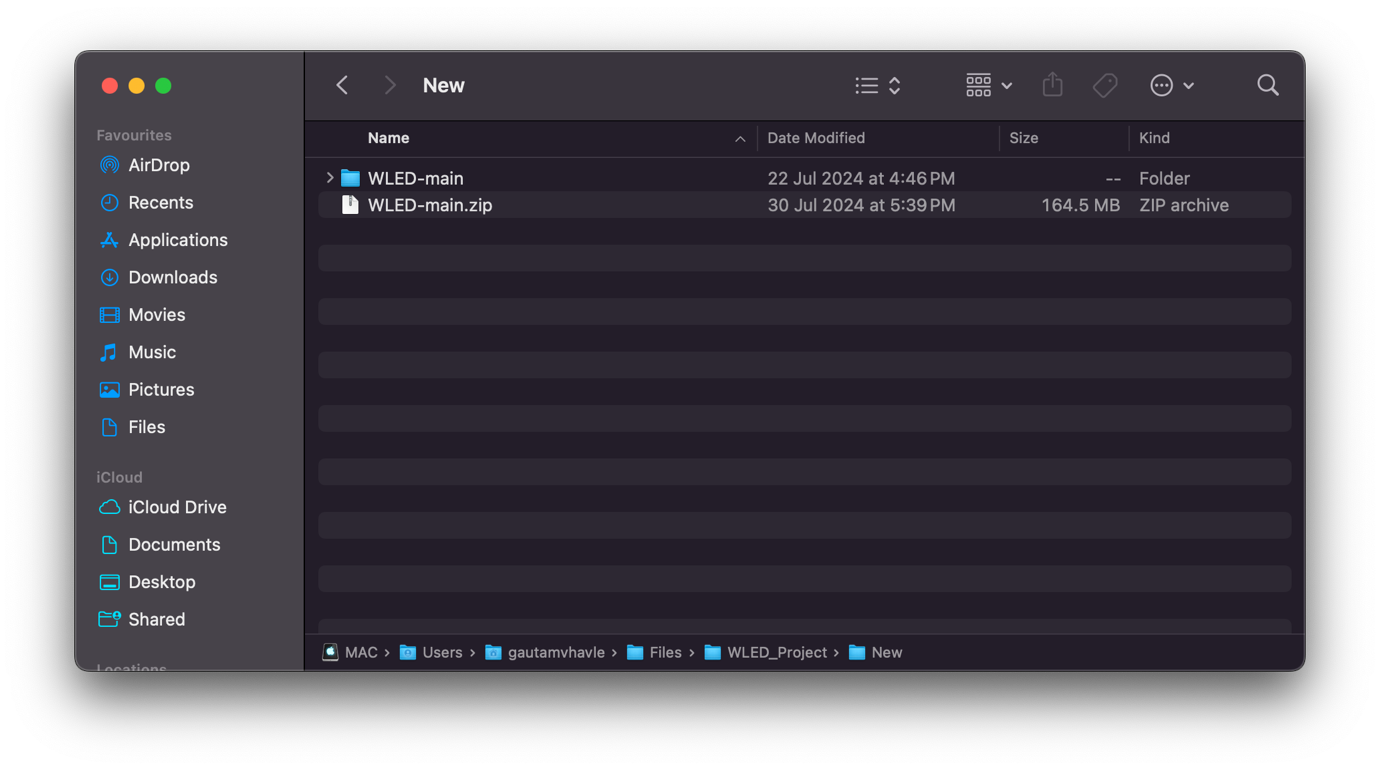
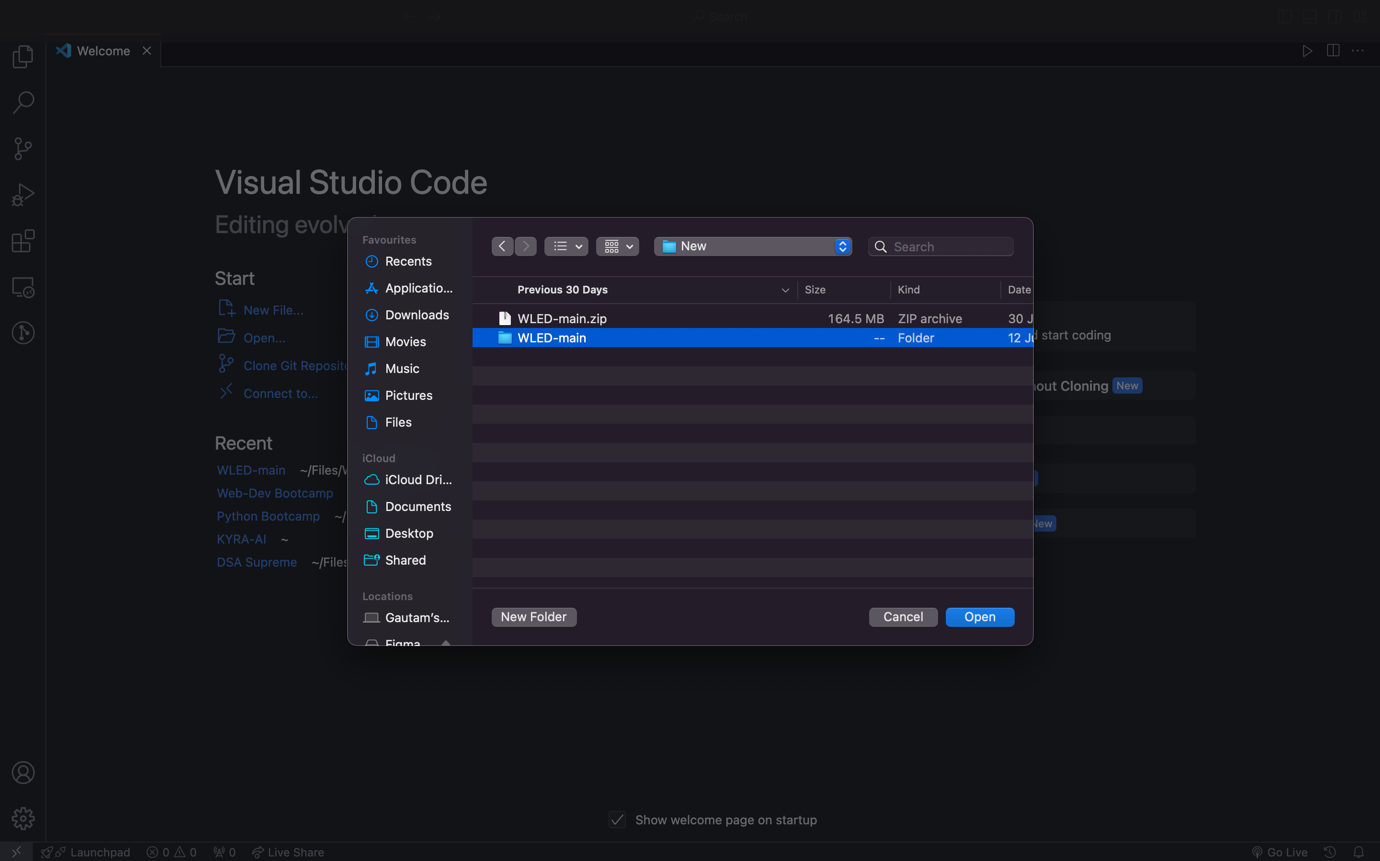
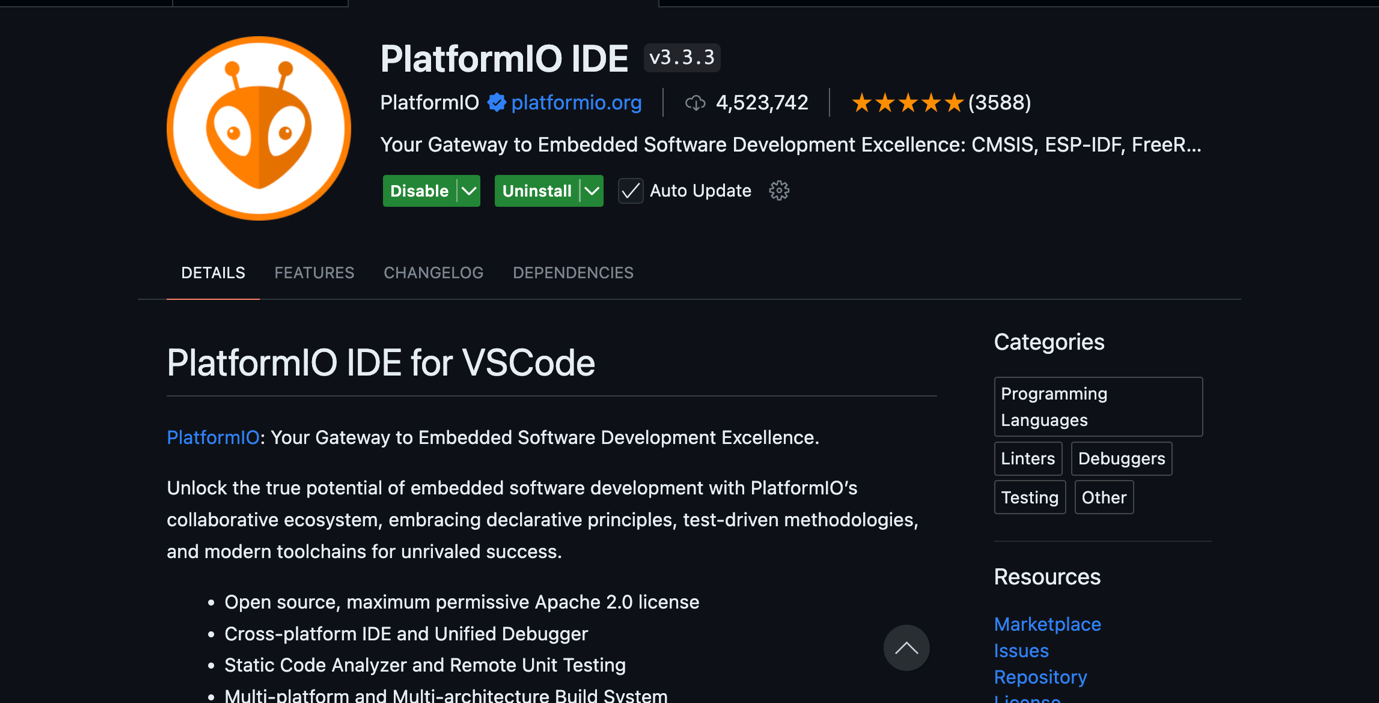
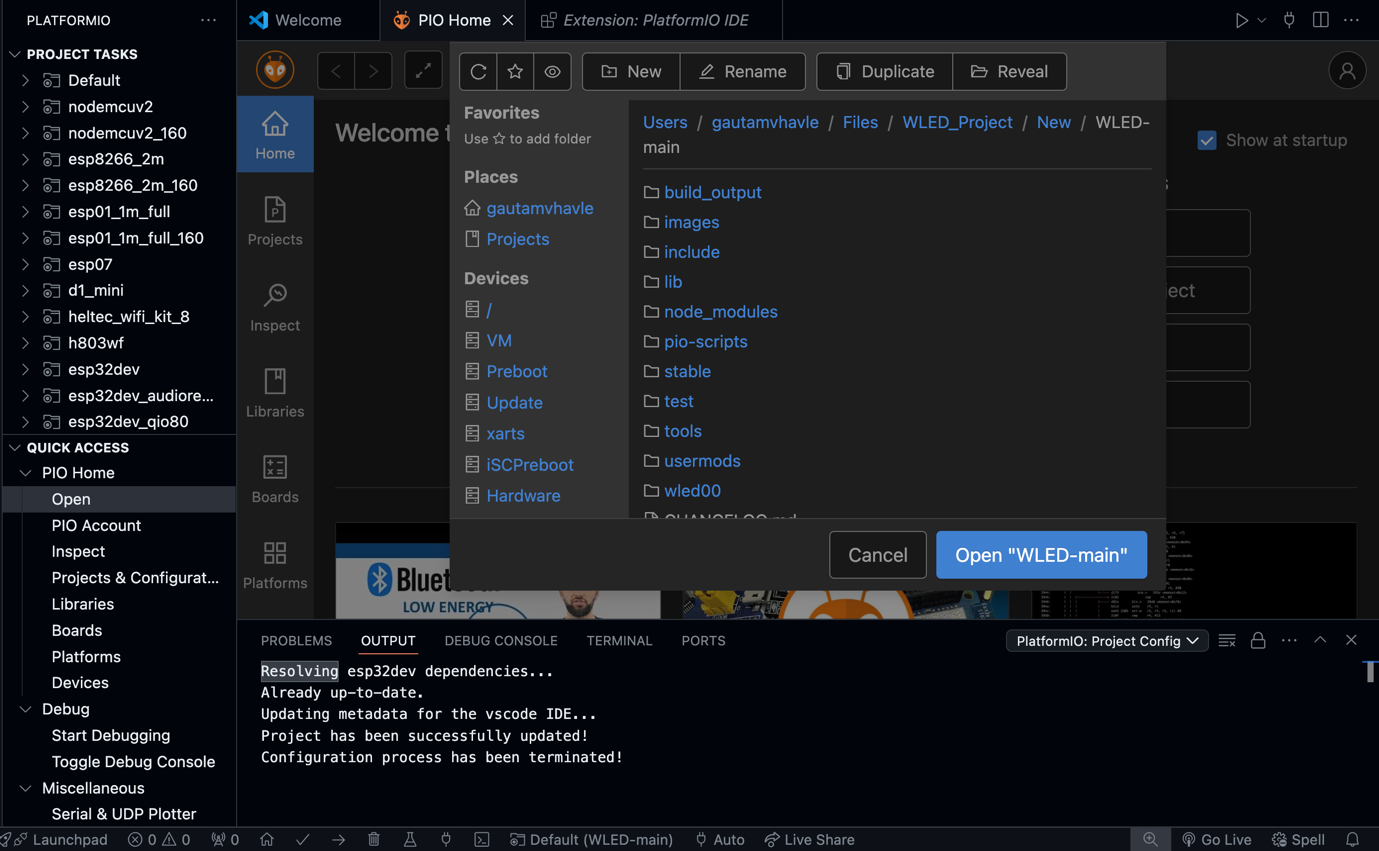
Documentation for WLED Project, By Gautam V  
  
  
1. Download the repo and extract it as a folder.  
  
2. Open the folder in vs code:

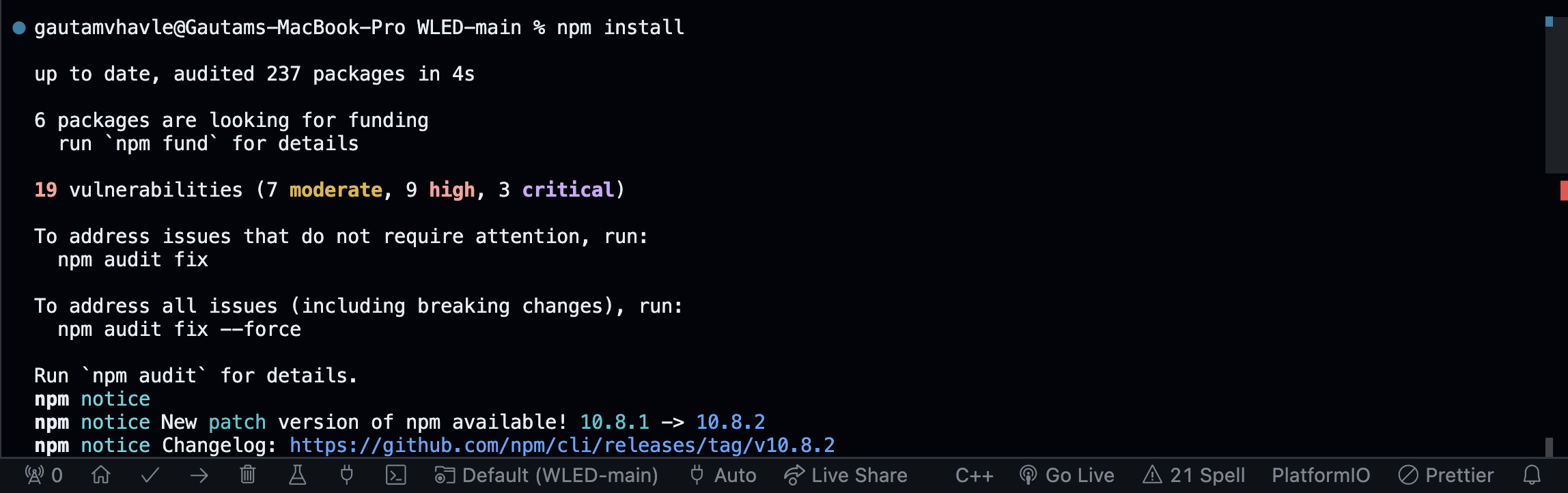


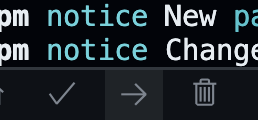
3. Install PlatformIO extension

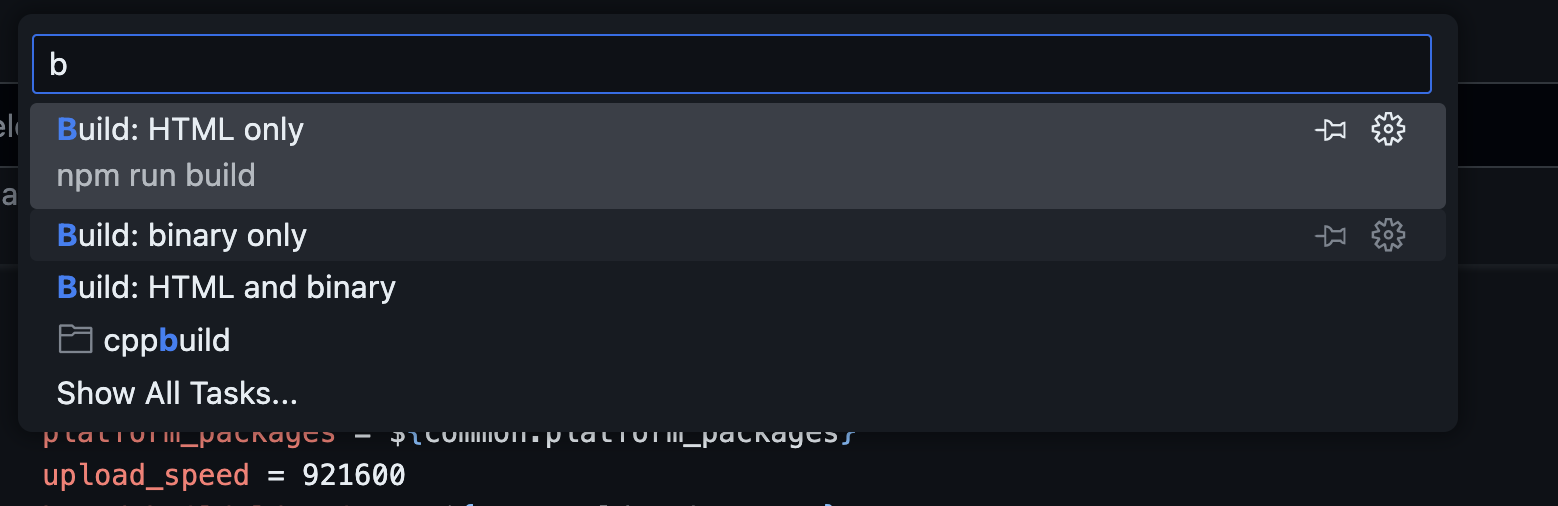


4. Open that project in PlatformIO as a existing project

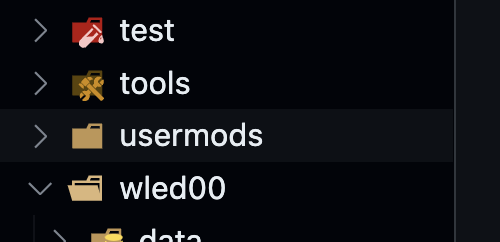


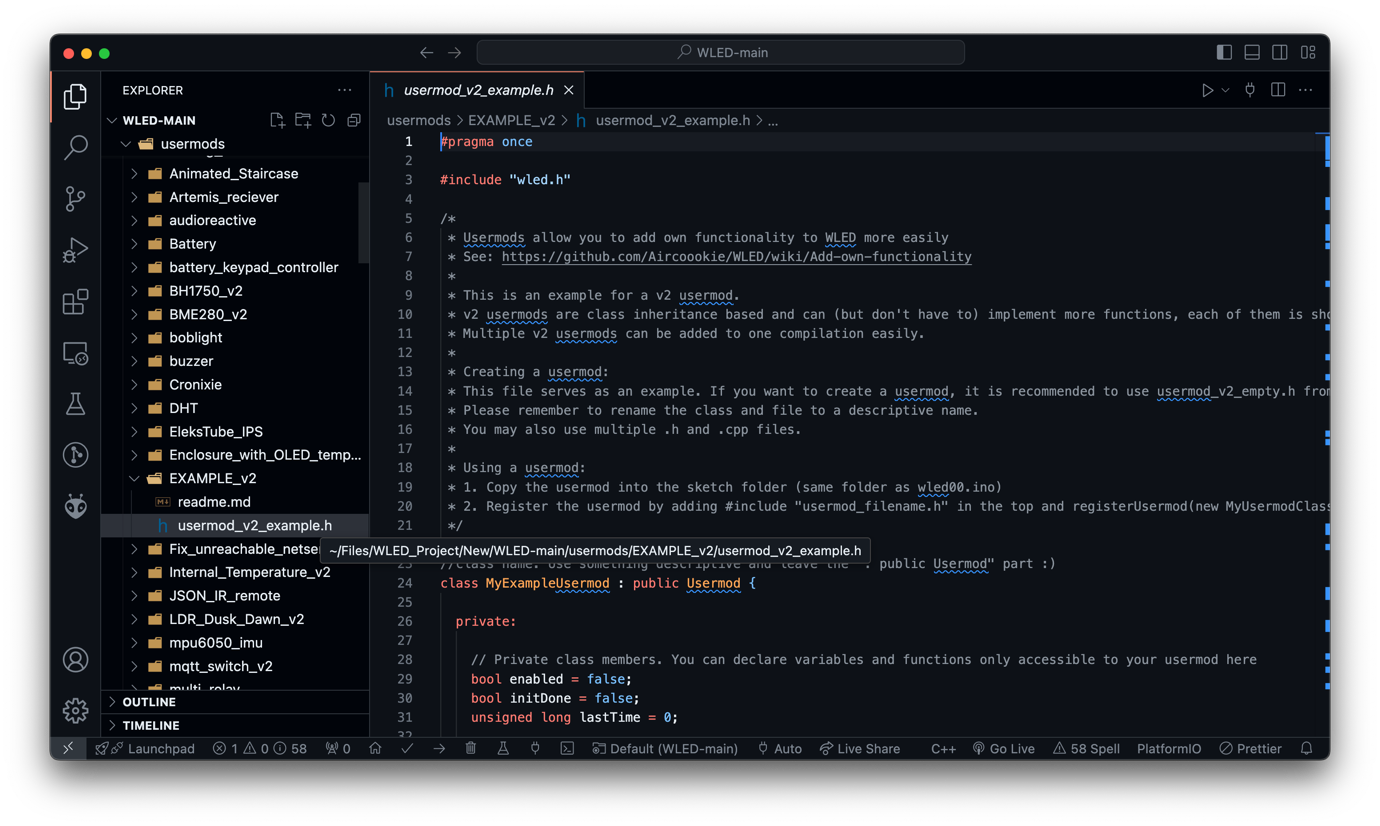
5. Once you are in the WLED folder, run ‘npm install’ to install all dependencies.  
  
  
  
6. Once all dependencies are installed, uncomment ” default\_envs = esp32dev “ to compile your program for esp32.

7. Instructions to compile your program:  
  


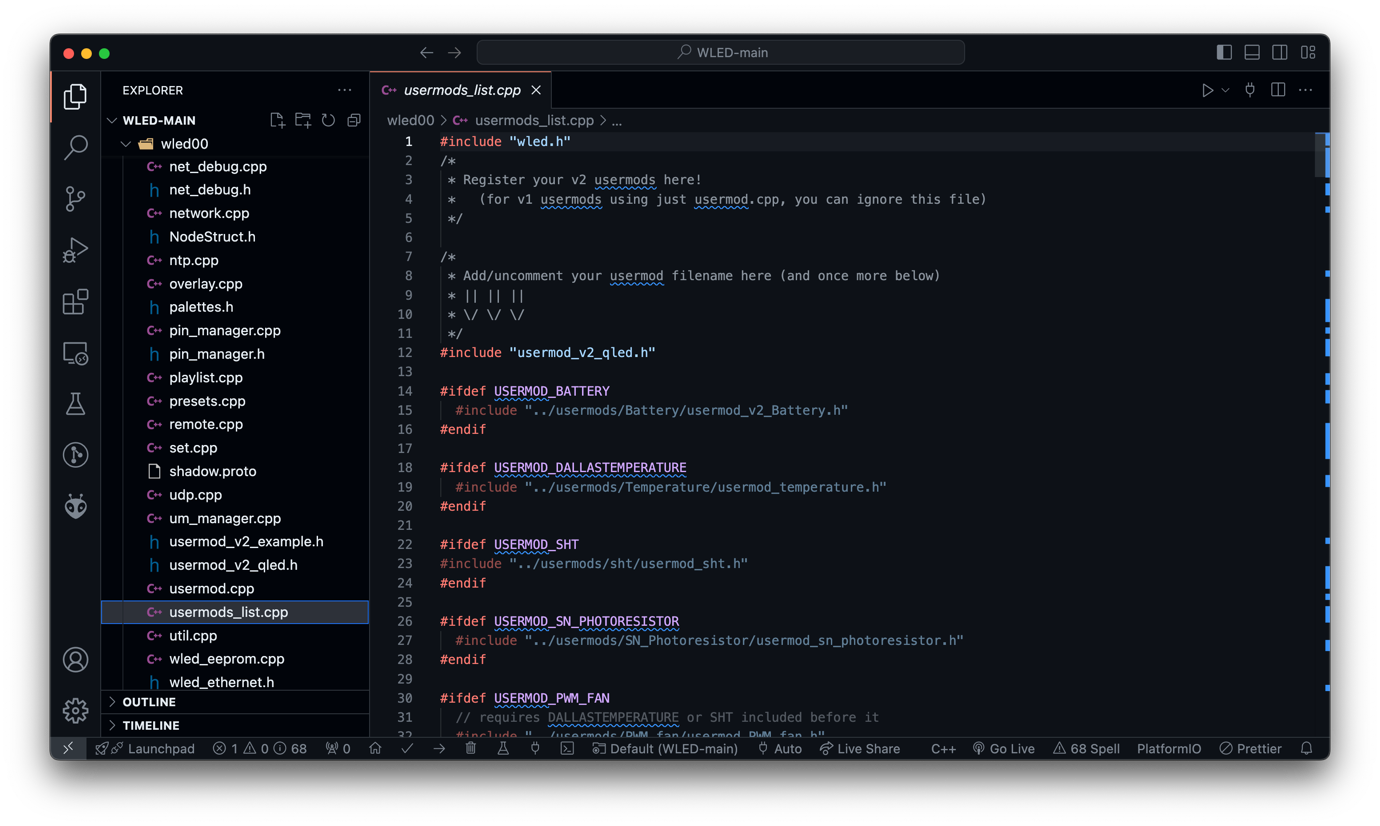
Usually, when we want to build the program, build button option is supposed to work, but in this case. Select build button, from the top select build binary only. The better option is to select upload, which is faster and if ESP is not connected it won’t flash anything onto the chip.

All the usermods are mentioned in this file:

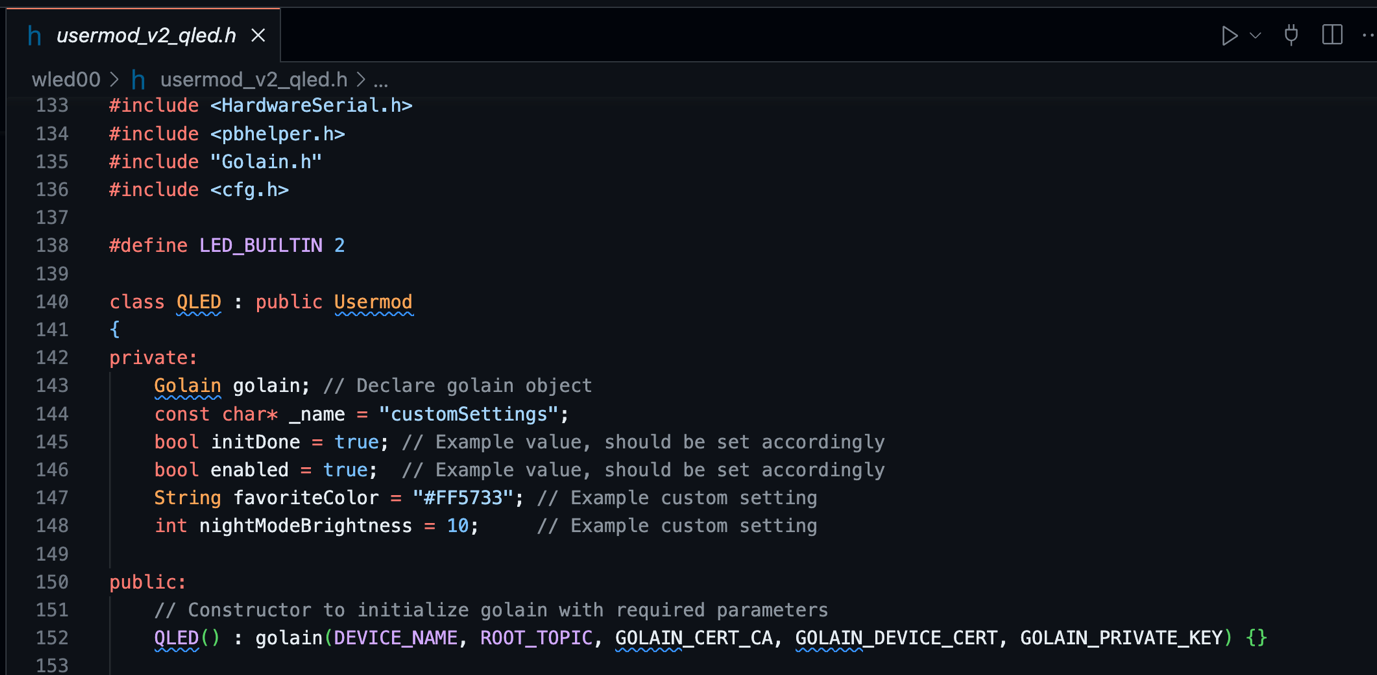


I want you to focus on this file especially: “usermod\_v2\_example.h” this contains everything how you can run your own usermod, with the readme. This is the latest method. And you are supposed to copy that usermod example file in the wled00 folder. You can check out the example which I made in the repo.  


Create and Add/declare your usermod file in this usermod\_list.cpp

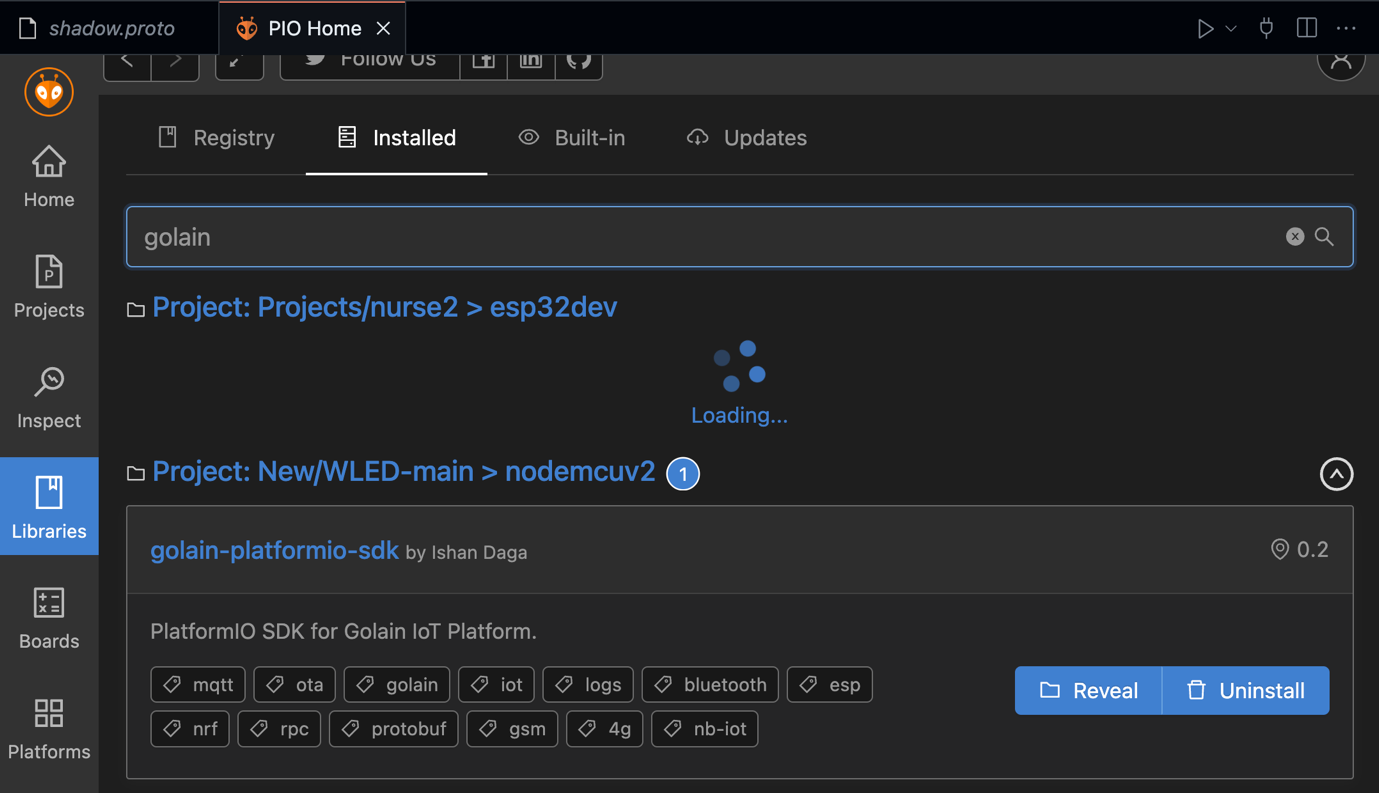


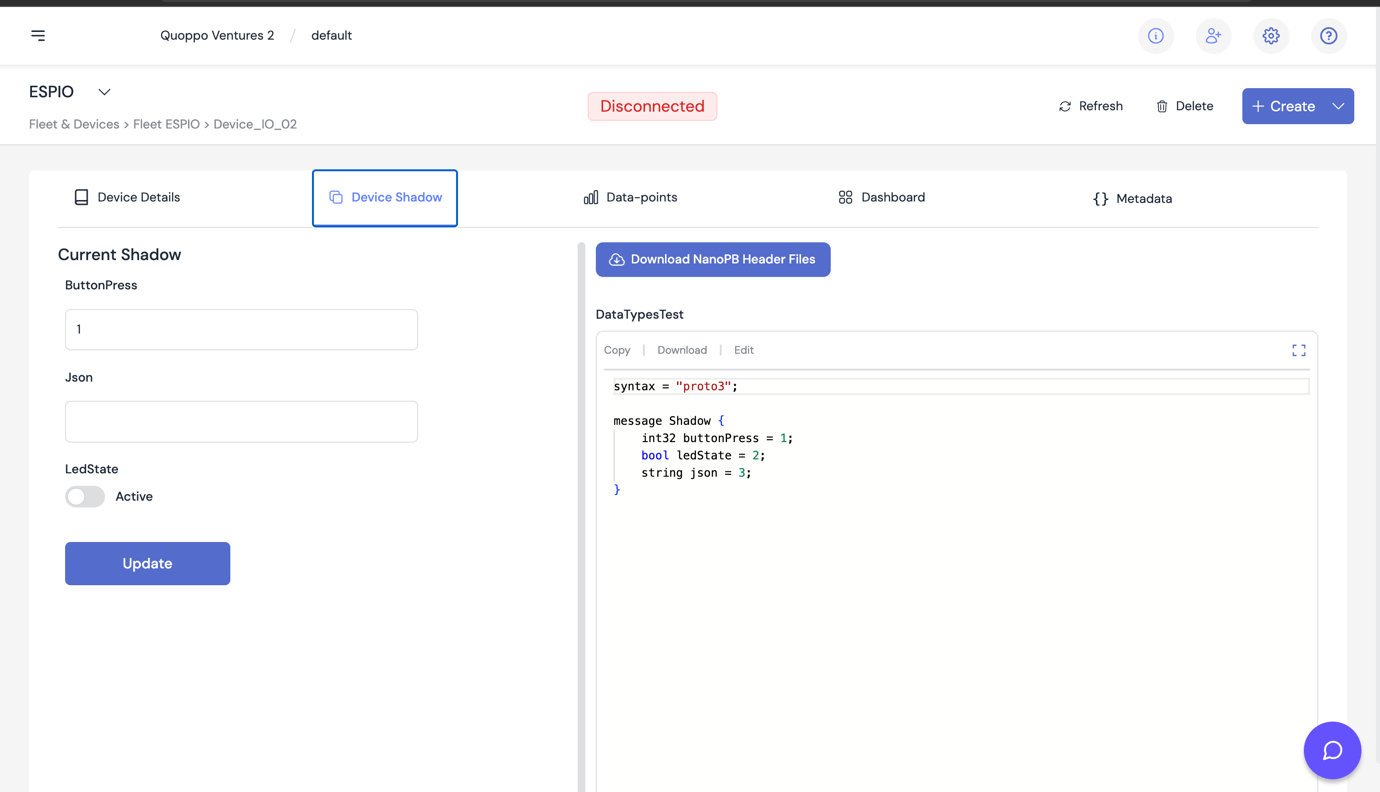
Create a class name in my case it is QLED as described in the screenshot:

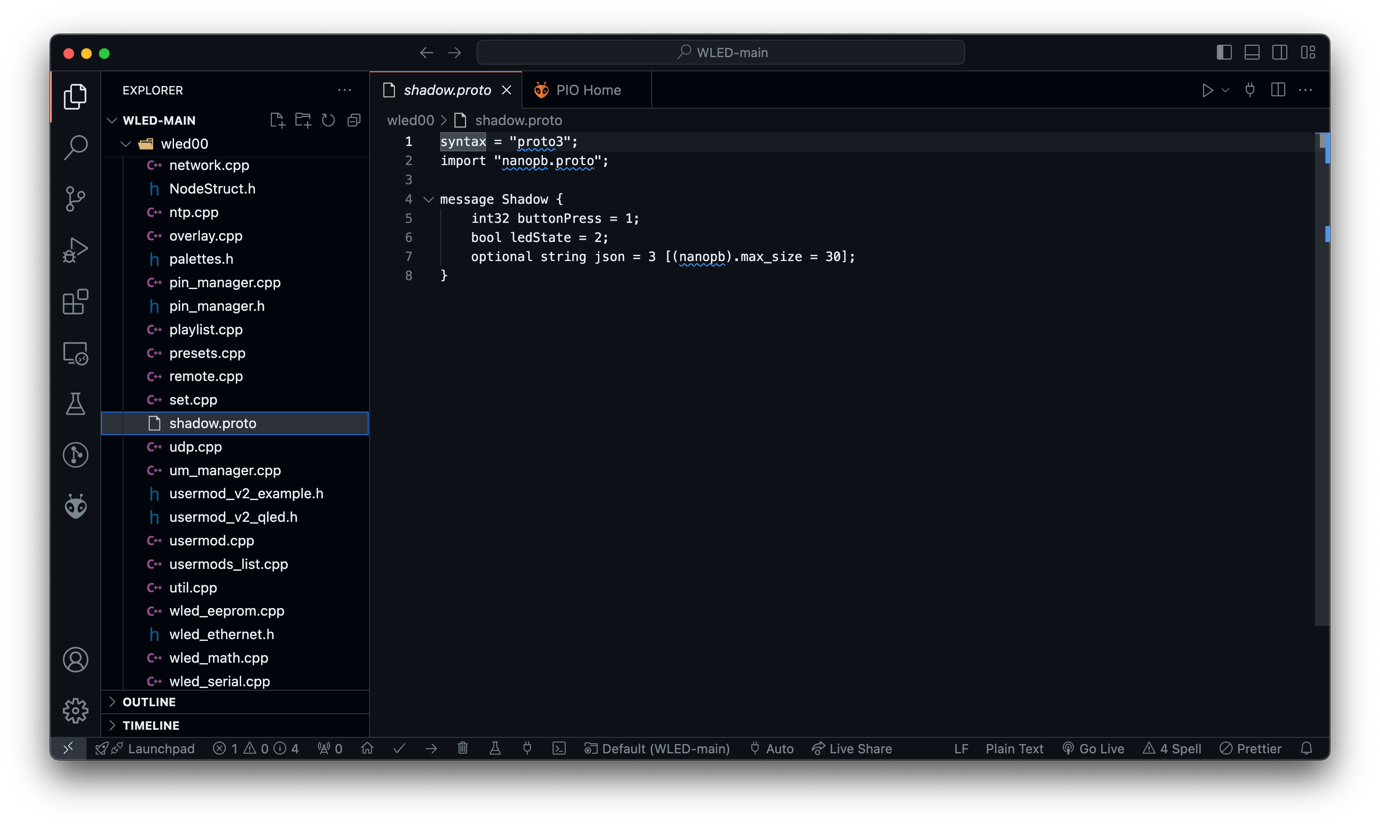


And call it in the last function in the last lines like this:

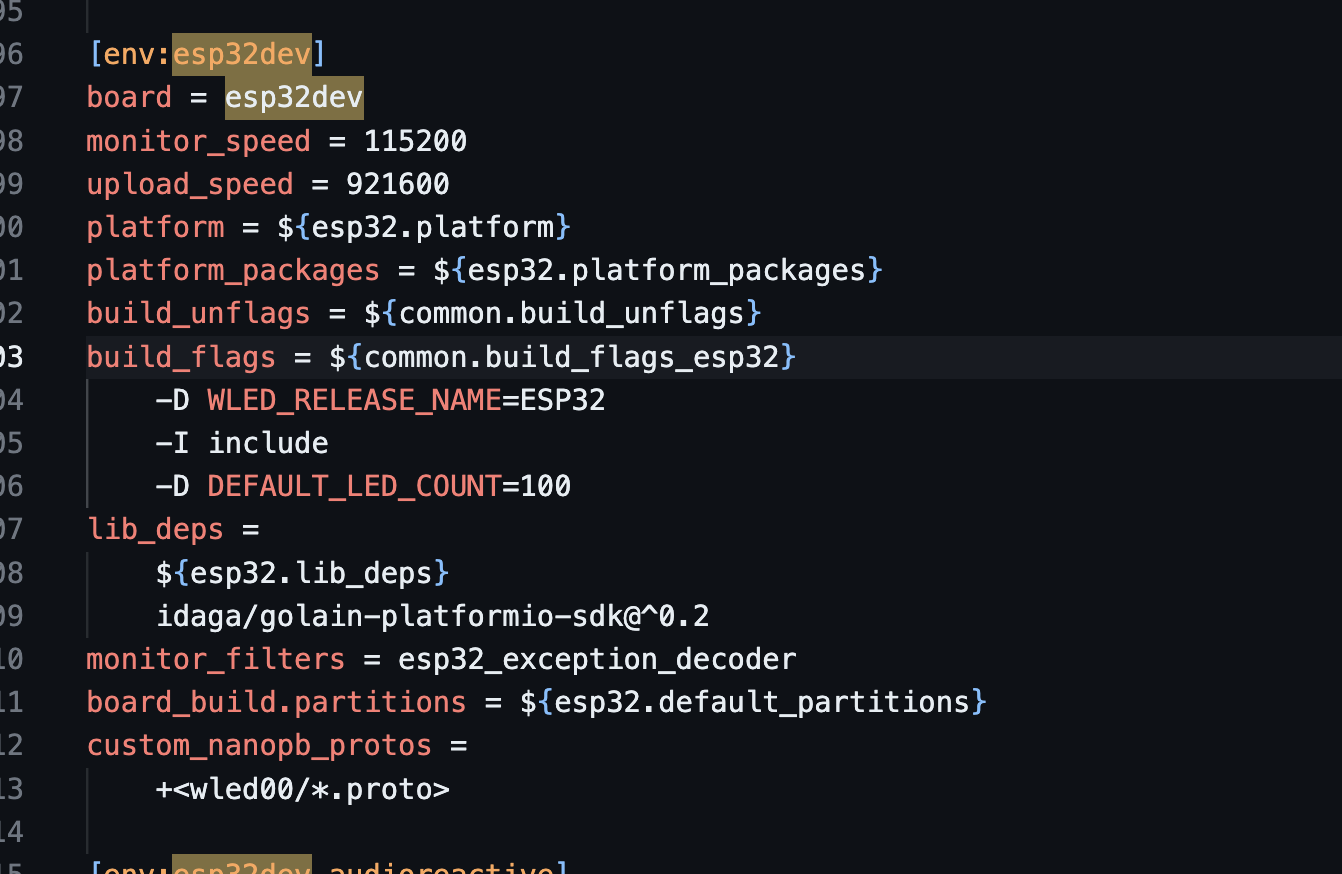


Before compiling make sure you have the dependencies installed, in my case and if you are working on golain integration I’ve to install Golain lib.  
  


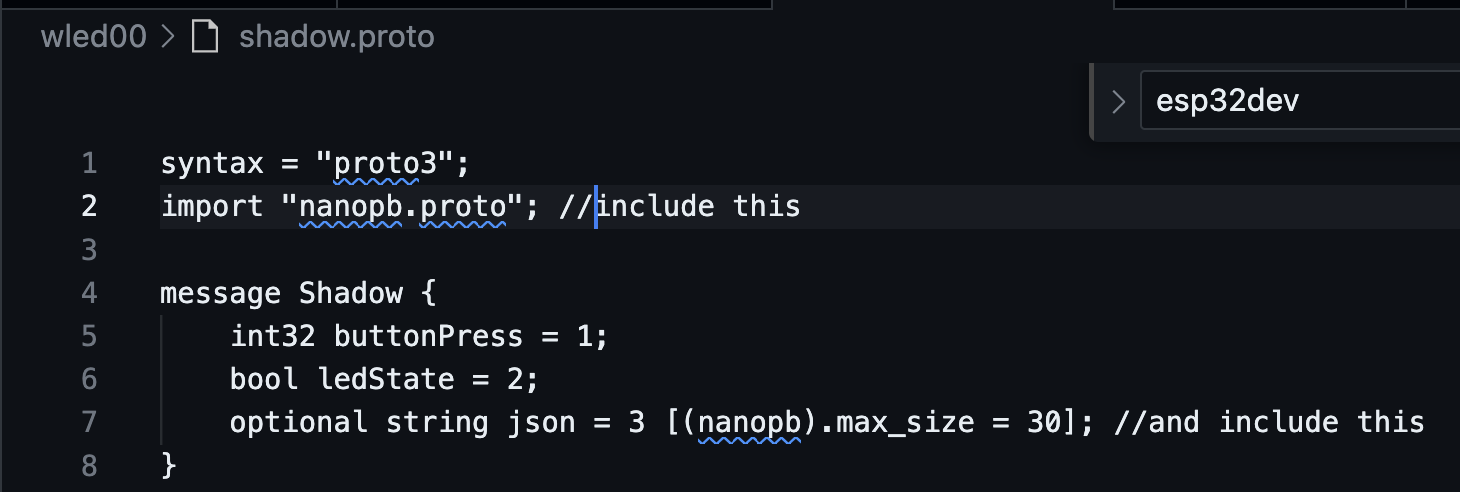
Once intalled, also include the necessary header files like certs and shadows, shadows are supposed to be placed in a seprate shadow.proto file like this, and it should be same as mentioned in the console foe proper functionality



Once done, ensure your platform.ini looks something like this:

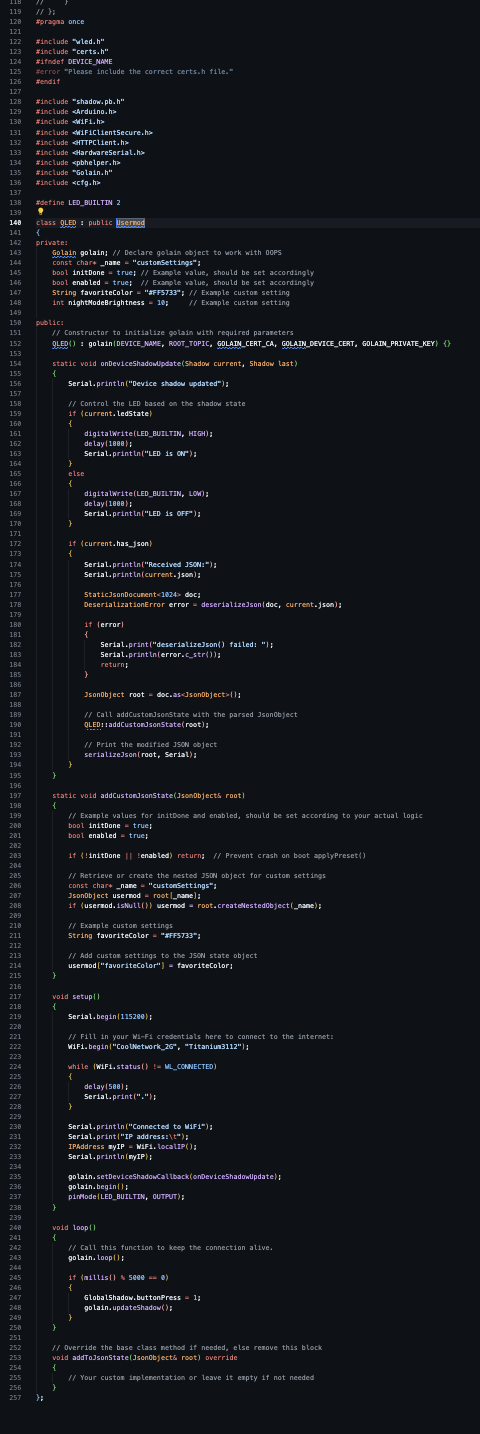


If all the dependencies are correct and required header files are provided, our program shoud be able to compile successfully.  
  
all the custom code is supposed to be pushed in the usermod file.



Observe the include statements in proto file. This Is how we can control the size of the string. If unspecified it may give errors, so enough size is supposed to be provided.

Using this code we have successfully brought the integration with golain, and we are able to fetch json stirings.



}}Blocker faced:  
  
1. It was hard working with OOPs while making this program.

2. troubleshooting was needed when implementing those json files getting parsed and calling the WLED functions as per the WLED JSON API documentation.  
3. Was not able to call functions easily from usermod.  
4. Needed more clarity on the global declarations which can be used in platform.ini and which were needed for compiling, there was no proper documentation available on their website or other sources.

Resources:  
<https://github.com/golain-io/golain-platformio-sdk>

<https://github.com/golain-io/golain-arduino-sdk>

<https://www.youtube.com/watch?v=S2x40jZ0Cfo&t=2279s\>

<https://www.youtube.com/watch?v=fRxyp5Yhivc&t=2901s>

<https://kno.wled.ge/advanced/compiling-wled/>

<https://kno.wled.ge/interfaces/json-api/>

<https://kno.wled.ge/interfaces/mqtt/>

<https://github.com/Aircoookie/WLED>