Passing grade requirements:

1. **System performs cyclic, synchronous measurement of process value**
   How is it done: We will configure a hardware timer (TIM3 maybe) in the STM32 to trigger an interrupt every 200ms. This period accounts for the BH1750 sensor's integration time of around 120ms.
   Inside the timer interrupt callback, we are gonna set a flag to tell the main loop to read the sensor and run the PID algorithm immediately.

2. **System allows for process control in safe range of process value**
   How is it done: We implement a software "saturation" block in our controller code.
   Before writing the calculated value to the PWM timer, it checks if it exceeds some set limit. Also the resistor R3 (look hardware schematic) is sized to limit the maximum LED current to a safe level even at 100% duty cycle.

3. **System ensure steady-state error at 5% of the control range**
   How is it done: We use the integration in our PI controller.

4. **System allows to set a reference value (set-point) with serial port or other UI**
   How is it done: We will set the reference value with the rotary encoder from the 8th requirement.

5. **System allows reading current values of measurement, reference, and control signals**
   I would combine it with the 12th requirement and make the python script that displays the values sent through the ethernet cable.

Additional requirements (The selected 9):

1. **Organization of source code**
   How is it done: We just do .c and .h pairs for each functional block.

2. **Use of a version-control system**
   How is it done: There is a git already, so we just need to work frequently.

3. **System ensure steady-state error at 1% of the control range**
   How is it done: We need to tune PI controller gains (especially K) to be aggressive so it closes small gaps without oscillations. I hope the existing integrating part will make the error less than 1%. :)

5. **Dedicated scripts or simulation models**
   How is it done: We will do a MATLAB script that will model our control loop. We may also use it later to generate plots for the report.

6. **Additional user output device**
   How is it done: My proposition is to connect an OLED display (maybe the SSD1306) to the I^2C bus, then use some library to send Lux value to appear on the screen while also updating it a few times per second.

7. **Additional control output device**
   How is it done: I think we may add a second LED controlled by a button, we will use it as a disturbances generator.
8. **Additional user input device**
   How is it done: We use a rotary encoder and connect it to a timer in encoder mode (probably TIM2) so we can change the brightness using it.

9. **Serial communication protocol uses a checksum**
   How is it done: We will add this anyways in the ethernet communication manually, we will make it check the checksum and give feedback or something.

12. **Use of network communication instead of a serial port**
    How is it done: We just connect the Nucleo to a laptop and use a LwIP library that is built in to STM32 so we can send UDP packet containing the lux data do the laptop. We also run a python script to display values.