Plant Pal

CPE301 Final Project

Matthew Amorelli, Chantelle Cabanilla, Jason Kimoto

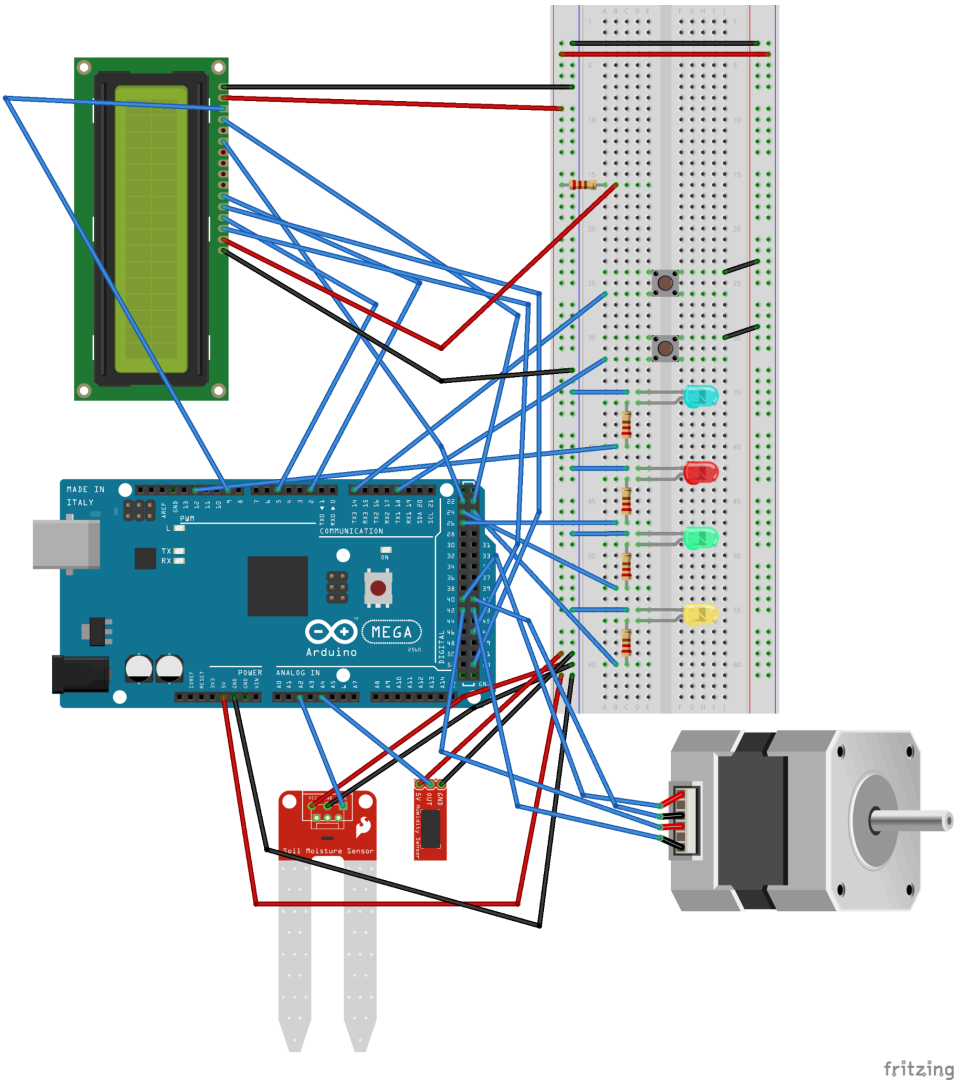
December 12th, 2025

CPE301

Video demonstration

https://www.canva.com/design/DAG7brSPlas/uuqeHY7Frq8NCPiTR1VAwA/edit?utm_content=DAG7brSPlas&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

Schematic diagram



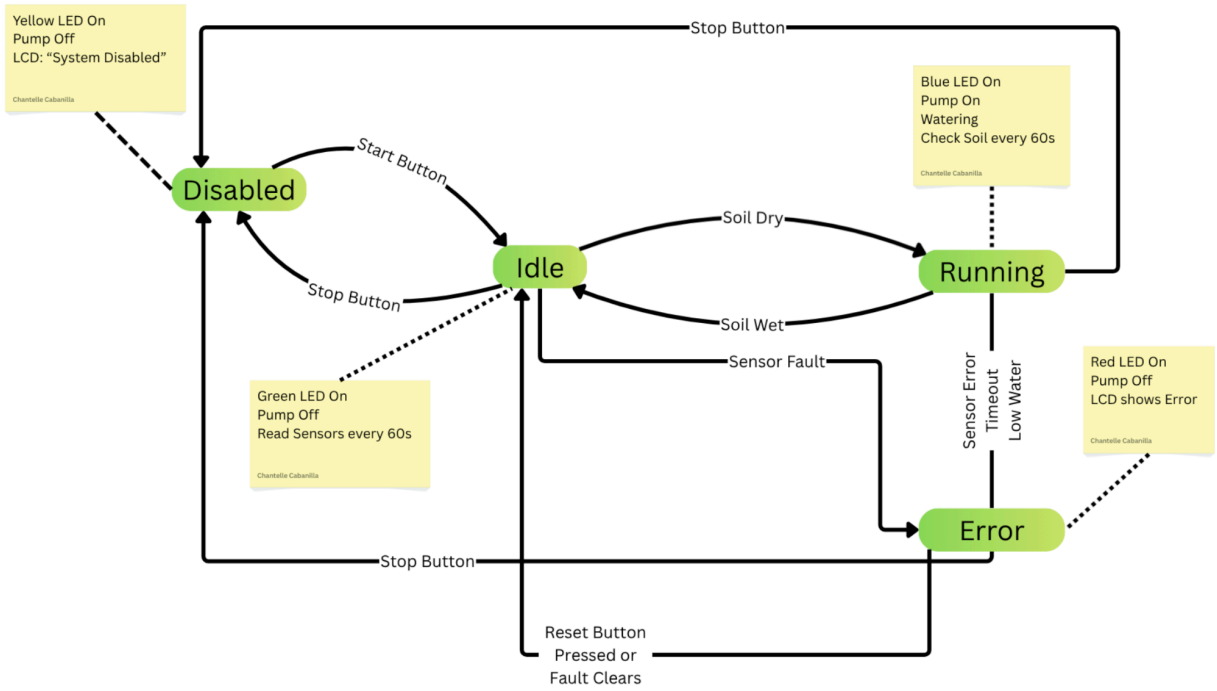
This schematic shows how all components in Plant Pal are connected to the Arduino Mega. The soil moisture sensor is wired to an analog input so the system can read soil wetness using manual ADC. The DHT11 temperature and humidity sensor is connected to a digital pin for environmental readings. The RTC module is connected through the Mega's I²C pins (SDA/SCL), allowing the system to keep accurate timestamps.

The LCD display uses the standard parallel wiring of RS, EN, and data pins D4–D7 to show sensor values and system messages. Each LED is connected to its own digital pin with resistors

for visual indicators. The Start and Reset buttons are wired to digital inputs (with the Start button on an interrupt-capable pin).

The stepper motor connects through the ULN2003 driver board, which interfaces it with four digital control pins on the Mega. All components are grounded and powered appropriately to ensure reliable operation.

Description of Diagram and State Changes



Disabled → Idle (Start Button Pressed)
The system starts completely off. If the Start button is pressed, it basically powers up the monitoring part of the system and switches into Idle so it can start checking the soil.

Idle → Disabled (Stop Button Pressed)
If the Stop button is pressed while the system is just sitting in Idle, everything shuts off again and it goes right back to the Disabled state.

Idle → Running (Soil Dry Detected)
While it's in Idle, the system checks the soil every so often. If it notices the soil is dry, it turns the pump on and moves into the Running state to begin watering.

Running → Idle (Soil Wet Detected)
If the system is watering and the soil finally reads as wet, then the pump is turned off and it returns to Idle to continue normal monitoring.

Idle → Error (Sensor Fault)
If something goes wrong with the moisture sensor while the system is idle—like the readings don't make sense or it's not responding—the system doesn't try to operate blindly. Instead, it switches into Error so the user knows the sensor needs attention.

Running → Error (Sensor Error, Timeout, or Low Water)
If the system is watering and anything goes wrong during that time (sensor stops responding, watering takes too long, or there isn't enough water in the reservoir), the pump is stopped immediately and the system moves into Error for safety.

Error → Idle (Reset or Fault Resolved)
If the problem is fixed or the Reset button is pressed, the system leaves the Error state and goes back to Idle so it can start checking the soil again.

Error → Disabled (Stop Button Pressed)
If the system is already in Error and the user presses Stop, everything is turned off and it goes right back to Disabled.

Running → Disabled (Stop Button Pressed)
If the user presses Stop during watering, the pump shuts off right away and the system goes into the Disabled state.

Environmental Impact Section

Energy Efficiency
For energy use, the system doesn't really pull that much. The pump only turns on when it has to, and the sensor checks are spaced out so it's not constantly running things in the background. Since the parts don't draw much power anyway, it doesn't take a lot to keep the system running. Honestly, it ends up using less energy than a watering setup that just turns on on a timer even when the plant doesn't actually need water.

Design Safety
Safety-wise, we kept it simple. Everything runs at low voltage, and we added checks so the system doesn't damage itself if something feels off. If the sensor starts giving weird readings, the water runs out, or the pump has been running longer than it should, the system just shuts the pump off and switches into an error state. The LEDs and LCD make it really obvious when something is wrong so you don't have to guess.

Affordability
All the parts we used were cheap and easy to get. It's basically the Arduino, a few LEDs, resistors, a small pump, and the moisture sensor. Nothing fancy. It keeps the cost low, and it also means if something breaks, you can replace the part without spending a ton of money.

Sustainability
The main sustainability benefit here is avoiding water waste. Since the system only waters when the soil actually needs it, it helps prevent overwatering. Most of the hardware can be taken apart and reused for another project later, so you're not just throwing out a bunch of electronics. It's definitely better than a system that waters on a timer no matter what.

Accessibility
The interface is simple enough that anyone can figure it out. The LEDs and LCD give clear feedback about the system's state, and the buttons make controlling it easy. After everything is set up, it doesn't require much from the user, which is great for people who don't have time to constantly check their plants or don't know how to read soil conditions.