

|  |  |
| --- | --- |
| Dipartimento di  INGEGNERIA GESTIONALE, DELL’INFORMAZIONE E DELLA PRODUZIONE | |
| Corso di laurea in  INGEGNERIA INFORMATICA | |
| Classe n. | |
| Sviluppo e analisi prestazionale di una Serra Automatizzata con Raspberry | |
| Candidato: *Andrea Lenzi* | Relatore: *Chiar.mo Prof. Davide Brugali* |
| Matricola n.  1066922 |  |
|  |  |
|  |  |
| Anno Accademico  2021/2022 | |

**Indice**

**Elenco delle figure**

[Figura 1: Icona Raspberry Pi 9](#_Toc117505658)

[Figura 2: Componenti Raspberry Pi3 10](#_Toc117505659)

[Figura 3: Sensore Temperatura e Umidita DHT11 11](#_Toc117505660)

[Figura 4: Sensore Umidita del terreno 12](#_Toc117505661)

[Figura 5: Sensore di intensità luminosa BH1750 12](#_Toc117505662)

[Figura 6: Componenti reattivi utilizzati 14](#_Toc117505663)

**Capitolo 1**

**Introduzione**

* 1. **Introduzione alla automazione**

Quando si parla di automazione, in genere ci si riferisce all’insieme di tecnologie e sistemi in grado di gestire macchinari e linee di produzione in modo intelligente. Ciò significa, programmare una macchina fisica affinché esegua un’azione automatizzata in totale autonomia. Parte del significato di automazione si trova nel concetto di robotica: una branca dell’automazione che prevede l’esecuzione di un lavoro attraverso appositi sistemi automatici, ovvero i robot, in grado di sostituire al meglio l’attività umana. In questo senso l’automazione robotica prevede uno svolgimento più rapido, sicuro ed efficiente di alcune attività specifiche.

Le evidenti qualità dell’automazione riguardano la facilità di gestire, modificare e adeguare non solo l’infrastruttura IT (Information Technology), ma molti dei processi produttivi. A spingere verso questo cambiamento c’è da un lato la necessità di ridurre il divario con la digital trasformation e dall’altra la volontà, da parte delle aziende, di posizionarsi come innovatori nel proprio settore di appartenenza.

Tra i vantaggi troviamo una maggiore produttività del lavoratore che può dedicare il proprio tempo ad altre attività lavorative mentre il software lavora su operazioni ripetitive. Automatizzare tali attività consente di ridurre i margini di errore e i rischi alimentando una maggiore affidabilità. Infine, codificare i processi li rende più sicuri.

Sebbene i vantaggi siano evidenti ci sono degli elementi da valutare e affrontare in questa importante trasformazione, il più importante di questi è il costo. Le creazioni di soluzioni automatizzare richiedono tempo, risorse economiche e valutazioni tecniche. In merito a quest’ultimo punto, infatti, è opportuno verificare quali siano i processi da automatizzare. L’automazione non è una strategia applicabile ovunque è perciò necessario valutare rischi e vantaggi al fine di eseguire procedure più sicure, efficienti e meno costose.

* 1. **Introduzione al progetto**

L’obbiettivo di questa tesi è quello di, attraverso un progetto proposto, svolto e pubblicizzato dalla scuola agraria I.I.S. Mario Rigoni Stern, sviluppare un prototipo di serra automatizzata attraverso l’utilizzo di un Raspberry Pi, al fine di: innanzi tutto, verificare se questa soluzione possa essere o meno una valida alternativa alla coltivazione tradizionale, sia a livello economico che qualitativo del prodotto, e in secondo luogo, avvicinare le nuove generazioni ad un possibile nuovo approccio, quello della digital trasformation, più vicino ai giorni d’oggi, dove la tecnologia e la automazione dei processi produttivi sono alla base di gran parte delle aziende e start up.

Lo scopo è quindi quello di pensare, progettare e analizzare le prestazioni di una serra automatizzata che sia il più intuitiva possibile e sfruttabile da qualsiasi tipo di utenza, da un ingegnere informatico ad un neofito curioso e, anche grazie ai ragazzi frequentanti l’istituto agrario che parteciperanno al corso associato al progetto, valutare se questa possa essere o meno una soluzione alternativa alla coltura tradizionale in un futuro prossimo.

* 1. **Organizzazione della tesi**

Questo elaborato è suddiviso in tre macro-capitoli fondamentali per rendere più chiare le fasi del progetto e confortevole la lettura.

1. *Capitolo 2: Panoramica degli strumenti e tool utilizzati*, dove saranno elencati tutte le componenti fondamentali e tool principali utilizzati, sia per lo sviluppo del prototipo, che per l’implementazione software dello stesso.
2. *Capitolo 3: Realizzazione del progetto*, capitolo fondamentale dove saranno descritte nei minimi dettagli tutte le fasi del progetto, dalla costruzione fisica del prototipo, alla struttura elettronica per poi arrivare all’implementazione dello script utilizzato.
3. *Capitolo 4: Analisi prestazionali e obbiettivi,* dove verranno in un primo momento trattati i costi della manutenzione e confrontati con la coltura tradizionale e successivamente verrà descritto il progetto come esperienza didattica (se i ragazzi hanno capito e sono riusciti a replicare il prototipo).
   1. **Progetto Open Source e Repository GitHub**

Come scelta personale, ho voluto pubblicare come Open Source, un repository GitHub, dove sarà possibile visionare e condividere tutto il lavoro (sia a livello fisico che a livello Software) che è stato alla base di questa esperienza. Esso sarà disponibile presso il mio profilo personale: CaptainLento/Raspberry-Smart-GreenHouse oppure nel presente link: https://github.com/CaptainLento/Raspberry-Smart-GreenHouse.

**Capitolo 2**

**Panoramica degli strumenti e tool utilizzati**

**2.1 Raspberry Pi3**

Il Raspberry Pi è un computer a scheda singola sviluppato nel Regno Unito nel 2012. La scheda è stata progettata per ospitare sistemi operativi basati sul kernel Linux. È stato concepito un sistema operativo appositamente dedicato, chiamato Raspberry Pi OS.

A livello hardware il progetto si basa su un system-on-a-chip che incorpora un processore, una GPU, e una memoria da 256 Megabyte a 1 Gigabyte dipendentemente dalla versione del dispositivo. Il progetto non prevede né hard disk né unità a stato solido, affidandosi solamente a una scheda SD per il boot e per la memoria non volatile.

Per quanto riguarda il software invece, la Raspberry Pi Foundation diffonde ufficialmente sistemi operativi basati su GNU/Linux, fra cui NOOBS e Raspbian, NOOBS contiene un installer semplificato mentre Raspbian propone un procedimento di installazione testuale come Debian. Si tende a preferire un ambiente con funzionalità minime e prediligendo un utilizzo puramente incentrato sulle prestazioni della macchina.

Per quanto riguarda questo progetto, si è deciso di utilizzare un Raspberry Pi3 con SO Raspbian, la scelta e data dal fatto che la terza versione di Raspberry porta con sé prestazioni abbastanza elevate senza avere un costo fuori dalla portata di tutti, questo lo ha reso la migliore soluzione tra le opportunità fornite dalla casa produttrice.

**2.1.1 Confronto tra Raspberry Pi e Arduino**

La differenza principale tra Arduino e Raspberry Pi consiste nel fatto che il primo è un microcontrollore, che può eseguire solo il codice C compilato, mentre l’altro funziona (se richiesto) anche come sistema autonomo. Tuttavia, il Raspberry Pi non è sempre la scelta migliore per qualsiasi tipo di progetto, come vedremo analizzando vantaggi e svantaggi e le possibilità di applicazione.

Arduino gode delle seguenti caratteristiche:

* Fornisce un processore integrato comprensivo di periferiche (entrate, uscite e interfacce),
* L'ambiente di sviluppo basato sull'hardware che fornisce un'interfaccia di programmazione con diverse librerie già pronte, che facilitano notevolmente la programmazione.
* L'ambiente di sviluppo integrato è un grande aiuto per il controllo dell'hardware, ma è difficile che i neofiti della programmazione imparino qualcosa sulla scrittura del classico codice (uno fornisce un setup dell'hardware già pronto, cosa che semplifica il processo dispendioso di assemblaggio della base tecnica per i progetti.

Mentre Raspberry Pi si differenzia in quanto:

* Fornisce un setup dell'hardware già pronto, cosa che semplifica il processo dispendioso di assemblaggio della base tecnica per i progetti.
* Dispone già di tutti i componenti necessari per far funzionare il minicomputer autonomamente. Inoltre la maggior parte dei modelli riesce a connettersi di default alla rete (il che renderebbe facile la connessione ad un eventuale DB interno alla scuola).
* Il software non è già incluso nel pacchetto e deve essere perciò scaricato e configurato dall'utente.

Sebbene anche Arduino sarebbe stata una ottima soluzione per questo caso di studio, si è deciso di sfruttare la architettura di Raspberry Pi per vari motivi: La connessione ad internet e l’interfaccia grafica che il minicomputer fornisce, semplificano molto sia il lavoro di setup che la comprensione da parte dei ragazzi del progetto, e soprattutto, volendo insegnare a dei neofiti delle basi di programmazione, il codice python usato su Raspberry diviene sicuramente più utile in un futuro, rispetto al singolo sketch di Arduino (sempre in ottica puramente didattica).



Figura : Icona Raspberry Pi

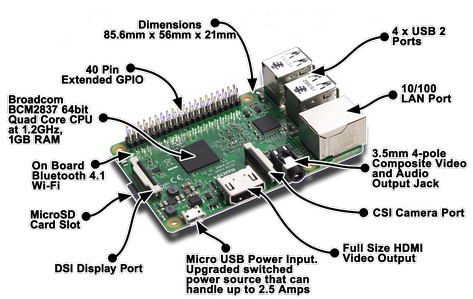


Figura : Componenti Raspberry Pi3

**2.1.2 Come Raspberry si interfaccia con l’esterno**

La maggior parte dei sensori, motori, luci e altre periferiche che rendono possibili questi progetti si collegano ai pin GPIO dei Pi. Ogni modello ha 40 pin GPIO; Affinché il progetto diventi realtà si deve conoscere il pinout del Raspberry Pi, ossia la mappa e la spiegazione di quello che può fare ogni pin. Il Raspberry gode di quattro tipologie di pin fondamentali descritte seguentemente:

*I pin GPIO (General Purpose Input Output).* Il GPIO è l’aspetto più semplice, ma accessibile del Raspberry Pi. I pin GPIO sono digitali, vuol dire che possono avere solo due stati, spento o acceso. Possono avere una direzione per ricevere o inviare corrente (input, output rispettivamente) e questo è tutto completamente controllabile tramite linguaggi di programmazione come Python, JavaScript, node-RED ed altri. I pin funzionano con una tensione di 3,3 V e con un assorbimento massimo di corrente di 16 mA. Infine, alcuni pin GPIO hanno anche funzioni secondarie che permettono loro di interfacciarsi con diversi tipi di dispositivi che usano i protocolli I2C, SPI o UART.

*I2C - Inter-Integrated Circuit.* L’I2C è un protocollo seriale a due fili a bassa velocità usato per collegare dispositivi mediante lo standard I2C. I dispositivi che usano questo standard hanno una relazione di tipo master-slave. Presenti sulla scheda sono due i collegamenti I2C.

*SPI - Serial Peripheral Interface.* SPI è un secondo protocollo per il collegamento di dispositivi compatibili disponibile sul vostro Raspberry Pi. Funziona similmente all’I2C in quanto esiste una relazione master-slave tra Raspberry Pi e i dispositivi collegati.

*UART - Universal Asynchronous Receiver / Transmitter.* Conosciuti anche solamente come “Serial”, i pin UART (trasmittente GPIO14 e ricevente GPIO15) forniscono un accesso console/terminale per l’installazione da remoto (headless), che significa potersi collegare al Pi senza tastiera o un dispositivo di puntamento.

Per il nostro progetto sfrutteremo, oltre ai GPIO semplici, solamente il bus I2C, cercando di limitare il più possibile complicazioni legate al conoscimento specifico del Raspberry e le implementazioni che ne seguono.

**2.2 Controllori utilizzati nel progetto**

Per far sì che la nostra macchina fosse messa in comunicazione con l’esterno, è stato necessario l’utilizzo di controllori. Essendo il nostro scopo riprodurre delle coltivazioni non in uso nei nostri territori, dopo una ricerca si è concluso che, al fine di monitorare al meglio le piante, è necessario controllare: la luce che queste assorbono (sia in quantità che in che frequenza), la quantità di acqua di cui queste hanno bisogno e l’umidita e la temperatura a cui hanno la necessita di restare.

Non cerchiamo una precisione molto elevata nelle misure dei sensori, né dei tempi esaustivi tra un controllo e l’altro, anche perché abbiamo cercato di limitare il budget il più possibile e in condizioni ottimali non ci aspettiamo variazioni di misure elevate in tempi brevi.

Ovviamente per lo sviluppo di un progetto più in grande scala si sarebbero potuti aggiungere altri sensori che avrebbero garantito un maggior controllo dell’ambiente circostante (come, per esempio, un misuratore del PH del terreno), ma essendo il primo prototipo si è deciso di mantenere sotto controllo solamente i fattori ambientali precedentemente descritti, nonché quelli ritenuti, non solo dal sottoscritto ma anche dagli studenti frequentanti la scuola agraria, più utili.

Una nota importante da sottolineare e la seguente; Il Raspberry possiede solamente pin digitali quindi per tutte le implementazioni di sensori in formato analogico e stato necessario l’utilizzo di un ADC (convertitore analogico-digitale) che ha permesso la corretta lettura di questi (sarebbe anche stato possibile evitare l’uso di un ADC e utilizzare un semplice circuito come un resistore e un condensatore in parallelo, sicuramente più economico ma meno intuitivo e diretto).

**2.2.1 Sensore temperatura e umidita aria DHT11**

Il sensore DHT11 è un sensore di temperatura e umidità con uscita dei dati in formato digitale di facile gestione con la scheda Raspberry attraverso una apposita libreria fornita da Adafruit (produttrice e venditrice di numerosi componenti elettronici e strumenti per la loro implementazione). Il sensore viene alimentato attraverso il pin da 5V e utilizza una tecnica digitale esclusiva che unita alla tecnologia di rilevamento dell’umidità, ne garantisce l’affidabilità e la stabilità. I suoi elementi sensibili sono connessi con un processore 8-bit single-chip.

Le sue piccole dimensioni e suo basso consumo permettono al sensore DHT11 di essere adatto per molti tipi di applicazioni, inoltre, il modulo con tre pin in linea rende facile la connessione alla scheda Raspberry.

A livello prestazionale, ha un range di misurazione dell’umidità che va dal 20 al 90 per cento, con una precisione di 5 per cento RH (Relative Humidity) e di temperatura da 0 a 50°C, con una precisione di 2°C; queste caratteristiche, e il costo non elevato ha fatto pensare che fosse la scelta più adatta al nostro progetto.

Immagine che contiene elettronico

Descrizione generata automaticamente

Figura : Sensore Temperatura e Umidita DHT11

**2.2.2 Sensore umidita del terreno**

Si tratta di un sensore capacitivo analogico che misura i livelli di umidità del suolo mediante rilevamento capacitivo, ovvero la capacità viene variata in base alla quantità d'acqua presente nel suolo. La capacità viene convertita in livello di tensione fondamentalmente da 1,2 V a 3,0 V massimo. Il vantaggio del sensore capacitivo di umidità del suolo è che sono realizzati in un materiale resistente alla corrosione che gli conferisce una lunga durata.

Anche lui come il DHT11 descritto precedentemente ha un range di misurazione dell’umidità che va dal 20 al 100 per cento, con una precisione di 5 per cento RH (Relative Humidity) ma, essendo esposto a corrosioni, dopo un determinato periodo di tempo e consigliato cambiare il sensore per avere dei valori non errati.

Immagine che contiene testo

Descrizione generata automaticamente

Figura : Sensore Umidita del terreno

**2.2.3 Sensore intensità luminosa BH1750**

Il sensore BH1750 presente sul breakout GY-302 è in grado di trasformare l’intensità della luce in lux che lo colpisce in un segnale digitale disponibile tramite interfaccia I2C presente nel Raspberry (il lux, simbolo lx, è l’unità di misura per l’illuminamento). Attraverso questo sensore siamo in grado di capire se c’è luce e con che intensità questa è presente.

Il modulo viene alimentato dal pin VCC con tensione 5V fornita dal Raspberry e collegato ai pin SCL e SDA che permettono l’accesso al bus I2C. Il sensore restituisce un intero senza segno a 16 bit.

Immagine che contiene elettronico

Descrizione generata automaticamente

Figura : Sensore di intensità luminosa BH1750

**2.3 Motori e altri componenti**

Per un ottimo sviluppo di una serra realmente automatizzata, non è sufficiente solamente avere dei controlli che ti permettono di misurare l’ambiente circostante ma, è necessario associare questi, ad un set di funzioni e motori che mantengono le condizioni prestabilite stabili e ad una infrastruttura che permetta il reale mantenimento di questi fattori ambientali classificati come ottimali per la specifica pianta coltivata.

**2.3.1 Come rispondere a determinati input**

Ciò su cui ci siamo concentrati di più e stato quello di evitare tre scenari che abbiamo classificato come disastrosi per qualsiasi tipologia di pianta seminata nella serra, che sono: evitare categoricamente temperature troppo basse e percentuali di umidita troppo alte, mantenere un’umidita del terreno costante e preservare livelli di luce, sia in termini di tempo che in quantità, accettabili.

Per quanto riguarda la temperatura, al fine di evitare temperature troppo basse si è deciso di optare per delle resistenze che, sfruttando il calore prodotto per l’effetto Joule, scaldano l’ambiente circostante. Viceversa, per evitare temperature troppo alte e livelli di umidita troppo elevati la soluzione migliore e stata quella di inserire due ventole che, se messe in funzione, abbassando sia i livelli di umidita nell’aria che quelli di temperatura percepita.

Al fine di preservare livelli accettabili di luminosità, abbiamo fornito il prototipo di una luce il più simile a quella emessa dal sole (la luce diurna del sole si aggira attorno ai 5000 Kelvin), in modo che questa si accendi all’occorrenza in mancanza di luce o semplicemente perché il livello di luminosità percepita diventi troppo basso.

Per mantenere il terreno umido e ricco di acqua da cui le piante possono attingere i propri nutrienti è venuta in soccorso una pompa a immersione che, messa in un apposito serbatoio esterno, apre e chiude un rubinetto quando l’umidita del terreno scende sotto determinati livelli.

Tengo a sottolineare che, ovviamente, i valori da mantenere stabili variano da coltura a coltura e non esiste uno standard fisso valido per ogni piantagione, aspetto importante affrontato nella implementazione del codice. L’unico lavoro manuale sarà quindi quello di: scegliere una pianta ad inizio coltura e specificarla al software e, successivamente, mantenere controllato solo il serbatoio esterno dell’acqua.

Ogni elemento precedentemente descritto in questo sotto capitolo è stato controllato tramite i pin GPIO presenti sulla scheda Raspberry, ovviamente abbiamo privilegiato componenti che funzionassero ad una tensione di 3.3V o 5V già presenti sulla scheda, evitando l’utilizzo di relay per acquisite altre tensioni.

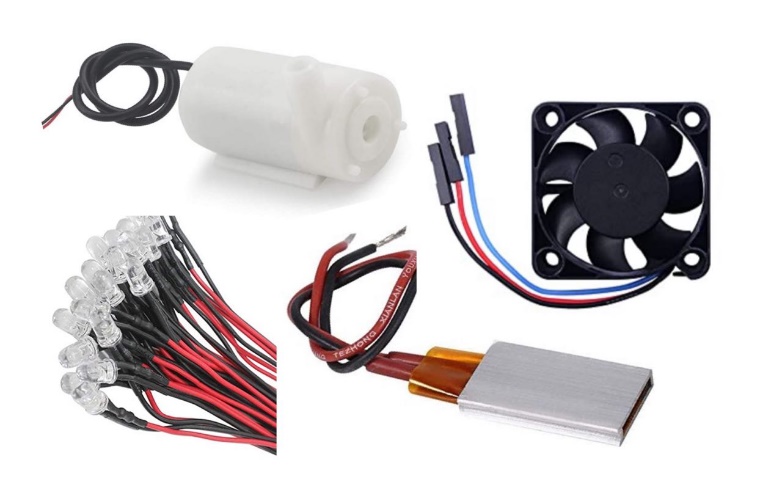


Figura : Componenti reattivi utilizzati

**2.3.2 Strumenti usati per la struttura del prototipo**

Volendo mantenere il costo relativamente basso, ho dovuto costruire a mano la struttura che avrebbe accolto il nostro progetto; l’obbiettivo della struttura era quello di, isolare il prototipo dall’esterno (perché altrimenti sarebbe stato inutile progettare dei motori per mantenere temperatura e umidita costanti) e allo stesso tempo mantenere un certo fascino e facilita di replicazione perché, come descritto nell’introduzione di questo elaborato, è stato importante far capire agli studenti come replicare questo progetto sui Raspberry offerti dalla scuola e, in secondo luogo, il prototipo sarebbe stato esposto nel laboratorio di informatica fino a temo indeterminato.

A fronte di ciò, ho deciso di costruire una struttura in legno con un doppio fondo, utile per nascondere eventuali parti di cablaggio non estremamente belle alla vista, e plexiglas, che permette, sia di far passare la luce (evitando sprechi), che di visionare lo sviluppo della pianta e incuriosire le persone. Le dimensioni della struttura e i particolari sono descritte nel capitolo 3.1 *Struttura e dimensioni del prototipo.*

**2.4 Software utilizzati per implementazione**

Una volta finita la struttura e i collegamenti non resta altro che sincronizzarli per ottenere un lavoro completo, a questo fine è necessario porre un codice che notifica i controllori quando prendere le misure e ai vari motori e componenti quando attivarsi e quando spegnersi; per fare ciò abbiamo sfruttato il linguaggio python, attraverso l’IDE, già preinstallato su Raspberry dal suo SO, Thonny.

**2.4.1 Perché Python?**

Come già detto in precedenza ciò che abbiamo cercato di ottenere con questo progetto è stato quello di raggiungere il risultato più efficiente possibile nel modo più semplice da capire per dei neofiti; principalmente per quest’ultimo motivo il linguaggio di programmazione python è stato il consigliato, perché, oltre ad essere uno dei linguaggi nativi di Raspberry, e uno dei linguaggi più semplici da spiegare a livello didattico a degli studenti (tant'è che viene utilizzato in molte scuole come punto iniziale per programmare, privilegiando questo linguaggio anche al C e a Java). È ricco di librerie, tra cui anche quelle offerte da Adafruit che sono state utilizzate per le implementazioni di alcuni controllori.

L’IDE (integrated development environment) Thonny è un ambiente di sviluppo già preinstallato sui Raspberry contenenti Raspbian quindi la scelta era pressoché obbligata. Sapendo benissimo che lo Script sarebbe potuto essere stato scritto ed eseguito attraverso lo shell dei comandi Linux abbiamo previlegiato un ambiente apposito per la più facile comprensione e comodità che i ragazzi nel corso di questo percorso avrebbero riscontrato.

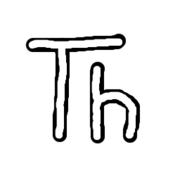


Figura : Icona IDE Thonny

**Capitolo 3**

**Realizzazione del progetto**

**3.1 Struttura e dimensioni del prototipo**

Il primo obbiettivo che mi sono posto per lo sviluppo di un prototipo e stato quello di ottenere una struttura che potesse accogliere la serra automatizzata nella sua interezza; per questo motivo ho voluto rispettare quattro punti fondamentali che, in ordine di importanza sono: isolamento dall’esterno, volontà di essere una struttura indoor, dimensioni non troppo elevate e mantenimento di un certo fasino per essere tenuta come oggetto da esposizione.

Ho voluto che la serra fosse un progetto indoor perché, oltre alla sua pura utilità di coltivazione, possa essere utilizzato anche come oggetto da esposizione (esempio nel laboratorio di Informatica), questo mi ha portato a voler progettare una struttura non solo efficiente ma anche elegante nella sua interezza. Inoltre, essendo posizionato all’interno di un edificio, non sarà necessario un dispendio elevato di energia per il mantenimento di determinate temperature nei periodi freddi dell’anno e inoltre, non sarà soggetta a intemperie.

Lo stesso motivo energetico lo si può applicare anche al primo punto dell’elenco, dato che, mantenuto l’isolamento il più possibile efficiente, una volta raggiunti i valori ottimali per la piantagione coltivata, non sarà necessario un ulteriore dispendio di corrente, o almeno non nel breve termine, limitando così sia i costi che gli sprechi elettrici.

Il terzo punto va di pari passo con i precedenti ossia, volendo creare un prototipo che non costi troppo e da esposizione, è stato necessario limitare le dimensioni anche perché come primo obiettivo ha quello di essere un esempio in piccola scala, da mostrare e insegnare a dei neofiti, che, all’occorrenza potrebbero ricreare in modo autonomo e adattarlo ad un progetto più ampio.

**3.1.1 Conformazione della struttura**

Come detto in precedenza ho voluto mantenere questi tre aspetti: dimensioni non elevate, isolamento ed eleganza, mantenendo dei costi affrontabili; al che mi ha portato a costruire una struttura prevalentemente in legno e plexiglas, dove e facilmente visibile la piantagione e allo stesso tempo le parti puramente elettroniche sono nascoste dalla struttura in legno, ma facilmente reperibile grazie al doppio fondo presente sul retro del prototipo.

Immagine che contiene testo

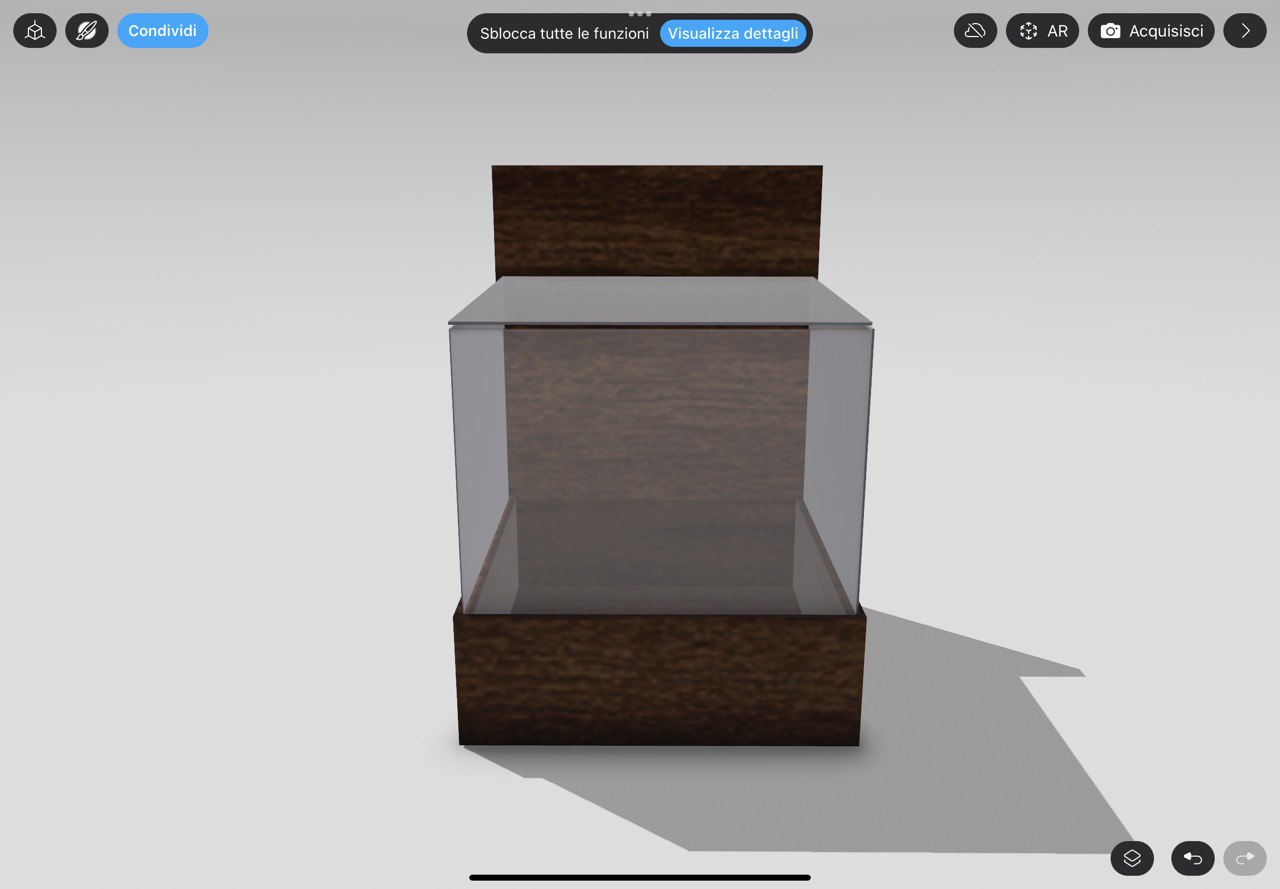
Descrizione generata automaticamente

Figura : Disegno 3D della infrastruttura

Ho dotato l’infrastruttura di un box esterno per evitare di avere una semplice bottiglia come recipiente per l’acqua. Inoltre, la parete dietro ha due funzionalità: evita che tutti i componenti siano in vista, soprattutto i cablaggi, garantendo così più pulizia nell’insieme, e facilita il posizionamento dei componenti che, a discapito di altri, devono essere posizionati internamente; ciò indirettamente facilita la comprensione dell’insieme da parte degli studenti che sono stati invitati a replicarlo

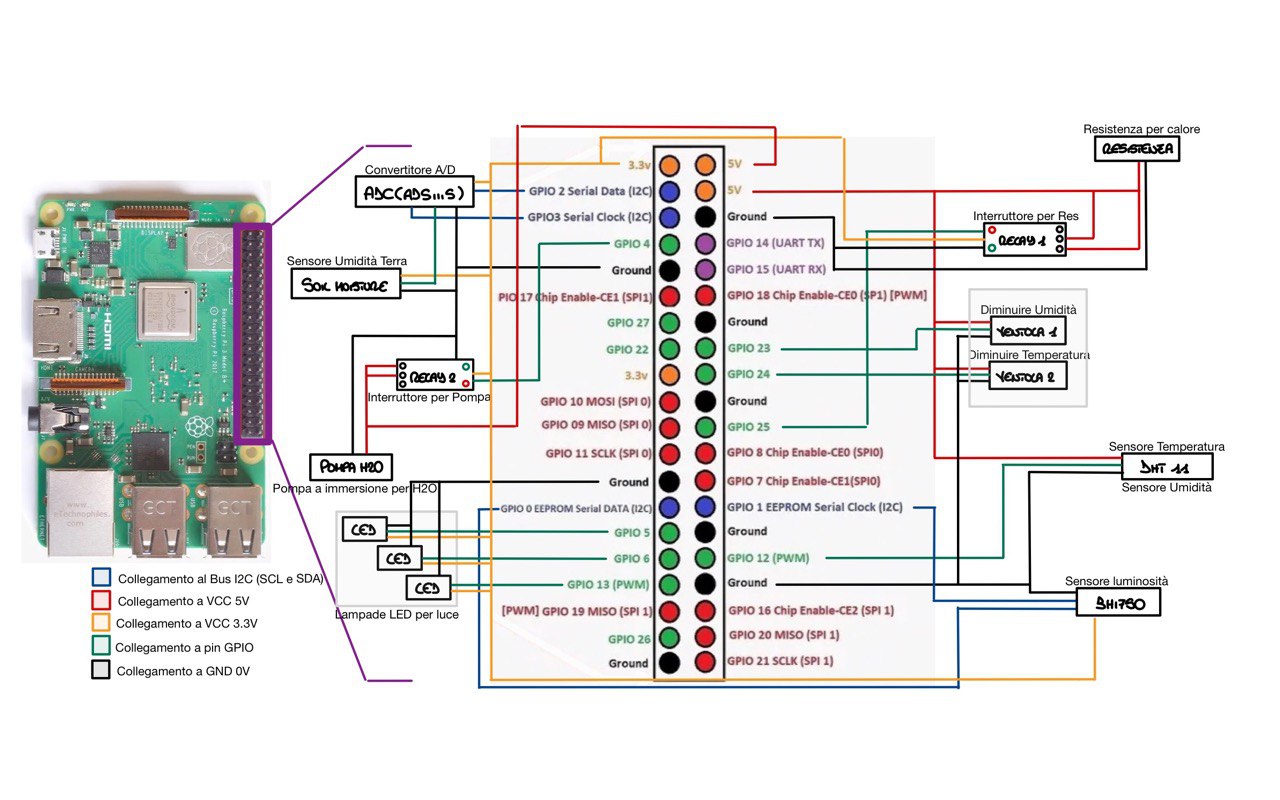
FOTO DORGETTO CON PLEX

Ammetto di non essere né un buon disegnatore CAD né un ottimo falegname quindi il risultato, benché soddisfacente, e dotato di alcune imperfezioni (niente che un maestro del mestiere non possa aggiustare).

**3.2 Cablaggio tra i moduli**

Per quanto riguarda il cablaggio tra i moduli, per comodità ho deciso di sfruttare, in un primo momento, un’estensore di Pin per breadboard sia per facilitare l’implementazione dei GPIO, che per avere a portata di mano a qualsiasi altezza della breadboard i VCC (5V e 3.3V) e il GND necessari per il circuito.

Tute le connessioni sono riportate nel seguente disegno (l’immagine e solamente di riferimento per i pin utilizzati su Raspberry così da poter facilitare la comprensione ed un eventuale riassemblaggio futuro). Ovviamente molti di questi pin sono interscambiabili ma, successivamente nello script andrebbero cambiati anche i riferimenti associati ad essi.



Una breve descrizione di tutti i pin utilizzati e di come i componenti sono stati collegati ad essi:

* Il controllore DHT11 (sensore di temperatura e umidita dell’aria) è stato collegato al VCC di 5V e al GND per la sua alimentazione e, successivamente, al pin GPIO 12 per il passaggio di dati; non è stato necessario usare la funzione PWM (pulse-width modulation) di cui gode questo pin. Per la sua implementazione è stata necessaria una libreria fornita da Adafruit meglio descritta nel sotto capitolo 3.2.2 *(Utilizzo di librerie e tool esterni).*
* Come risposta ai dati raccolti grazie al DHT11 abbiamo tre componenti; Due ventole, una per abbassare la temperatura e una per diminuire l’umidita presente nel sistema, collegate semplicemente attraverso i pin di alimentazione (VCC di 5V e GND) e rispettivamente ai pin GPIO 24 e 23 che, in questo caso, hanno solo la funzione di interruttore. Il terzo componente è una resistenza per alzare la temperatura se necessario; avendo solamente i pin di alimentazione 5V e GND è stato necessario l’utilizzo di un relay, collegato alla sua alimentazione attraverso i 3.3V e il GND presenti sulla scheda, sfruttato come interruttore grazie al pin GPIO 25.
* Spostandoci in basso a destra nella immagine troviamo il sensore di intensità luminosa BH1750, collegato al Pi tramite pin I2C attraverso i pin fisici 27 e 28, rispettivamente SDA e SCL, oltre alla sua alimentazione (VCC 3.3V e GND). Il bus I2C permette la trasmissione di dati non più solamente digitali, ma essendo una comunicazione seriale, da accesso ad una gamma di valori ben più ampia.
* Nella necessita di produrre luce ho collegato una serie di LED (tre ciascuno) ai pin GPIO 5, 6 e 13, anche qua non usando la funzione PWM nel pin 13, ed esattamente come gli altri componenti, li ho collegati alla propria alimentazione di 3.3V, (era possibile collegarli anche ad una tensione di 5V ma, la luce prodotta da essi era sufficiente).
* Essendo il sensore di umidita del terreno in formato analogico, è stata necessaria l’implementazione di un ADC (Convertitore analogico-digitale) per essere più specifici ho utilizzato un ADS1115, convertitore a 16bit. Anche questo componente lavora attraverso il bus I2C presente nella scheda di fatti, oltre ai collegamenti di alimentazione VCC 3.3V e GND sono stati usufruiti i GPIO 2 e 3, rispettivamente SDA e SCL; anche questo componente sfrutta una libreria specifica offerta da Adafruit anch’essa descritta nel capitolo 3.2.2 (Utilizzo di librerie e tool esterni). Il componente possiede 4 uscite digitali a cui ad una di queste abbiamo collegato il soil moisture sensor, oltre che ai soliti 3.3V e GND.
* Come risposta ai dati raccolti grazie a questo ultimo sensore ci viene in soccorso la pompa ad immersione che, come la resistenza per la temperatura, anche questa ha solamente i pin di alimentazione quindi, nello stesso identico modo, ho utilizzato un secondo relay, anch’esso come interruttore per la pompa comandato tramite il pin GPIO 4.

Una volta collegate tutte le componenti non resta altro che la vera e propria implementazione software del progetto.

**3.2.1 Tempistiche nell’assemblaggio**

L’assemblaggio dei componenti è stata sicuramente la parte più dispendiosa del progetto per vari motivi. Il primo sicuramente è quello del trasporto e reperibilità dei componenti stessi; non avendo mai lavorato su un Raspberry ma solamente sulla scheda Arduino, e avendo a disposizione pochi sensori, molti dei quali analogici, è stato necessario comprare il materiale sia per me, che per gli studenti che avrebbero dovuto replicare il tutto (per questi ultimi sono venute in soccorso le finanze scolastiche) e quindi necessariamente aspettare l’arrivo dei componenti.

Oltre a questo, è stato impegnativa anche la scelta dei singoli componenti, e degli strumenti ausiliari (ADC e relay). L’obbiettivo era di rendere l’assemblaggio una componente del progetto non complicata, per creare meno problemi di comprensione possibili ai neofiti che avrebbero dovuto riprodurlo al fine di concentrarsi principalmente sulla implementazione Software (non credo si riesca a trovare una soluzione più intuitiva adattabile per il nostro percorso).

Infine, come solito accade nella costruzione di prototipi, non nego che sono stati necessari anche pezzi di ricambio perché, avendo collegato nel posto sbagliato qualche cavo, esempio alimentazioni di 3.3V collegate ai 5V, alcuni componenti sono andati bruciati e conseguentemente inutilizzabili.

**3.3 Sviluppo software del progetto**

3.2.2 Utilizzo di librerie e tool esterni

Libreria adafruit.h

Scrivo tutto

Tengo a sottolineare che lo sviluppo hardware e quello software sono andati avanti di pari passo senza favorire l una rispetto all altra o viceversa, ma per correttezza e ordine nella scrittura di questo testo

tengo a sottolineare che realmente non è stata presente una vera e propria divisione temporale delle fasi progettuali anzi, nella realizzazione del prototipo si è cercato il più possibile di mantenere lo sviluppo dei tre macro-aspetti allo stesso passo implementativo:

Tutto quello che ovviamente e stato pensato può essere trasportato su una scala più grande fino anche ad ottenere realmente un progetto di impresa basato su questa idea.

**4 Performance del prototipo**

**4.1 obbitiitvi**

Bibliografia da fare

Automazione wikipedia <https://it.wikipedia.org/wiki/Au>

Raspbarry contro arduino <https://www.ionos.it/digitalguide/server/know-how/arduino-vs-raspberry-pi/>

tengo a sottolineare che realmente non è stata presente una vera e propria divisione temporale delle fasi progettuali anzi, nella realizzazione del prototipo si è cercato il più possibile di mantenere lo sviluppo dei tre macro-aspetti allo stesso passo implementativo:

<https://www.tomshw.it/hardware/pin-gpio-raspberry-pi/>