CODE 4LL

Relazione progetto

Automazione in magazzino

A.S. 2021/22 Classe 5J
CODE4LL

1. INTRODUZIONE

1.1. Obiettivo del progetto:

Lo scopo finale del gruppo Code4LL è la realizzazione di un software dedicato all'automatizzazione degli spostamenti e posizionamenti di merci all'interno di un generico magazzino, riducendo tempi e lavoro necessari nello svolgimento di tali mansioni da parte di esseri umani, concedendo inoltre una rappresentazione 3D che funge sia da interfaccia per l'operatore, sia da anteprima dell'operazione che verrà svolta.

All'interno del prototipo che si intende esporre è prevista un'interfaccia di comando via web con cui è possibile comandare un robot a distanza e una rappresentazione 3D del magazzino all'interno del quale è possibile muovere il mezzo virtuale in seguito ad una fase di autenticazione dell'operatore. Si intende rendere la simulazione utilizzabile sia da computer che da dispositivi mobile, grazie all'applicazione installata tramite apk.

1.2. Definizioni, acronimi ed abbreviazioni

IDE (Integrated development environments):

Unity – Rappresentazione 3D del magazzino

Blender – Realizzazione Modellino 3D Rasptank

Visual Studio – Stesura codice per Unity

Pycharm – Stesura codice per Rasptank

Linguaggi di programmazione:

C# - Programmazione rappresentazione 3D Unity

Python – Programmazione Rasptank e connessione al DB

PHP – Connessione al DB da Unity (Progetto finale)

HTML, CSS, Javascript – Realizzazione interfaccia web di comando

Componenti principali RaspTank:

Raspberry PI 4 model B

Motor HAT V2

Raspberry Camera

Sensori a infrarossi

Sensori ultrasonici

Altri strumenti:

MobaXterm – SSH client necessario per connettersi da remoto con Rasptank

Fing – Visualizzazione indirizzi IP dispositivi connessi alla rete

Firebase - Piattaforma per la realizzazione di app per dispositivi mobili realizzata da Google

Protocolli coinvolti:

SSH: Secure Shell

SFTP: Ssh File Transfer Protocol

IP: internet Protocol

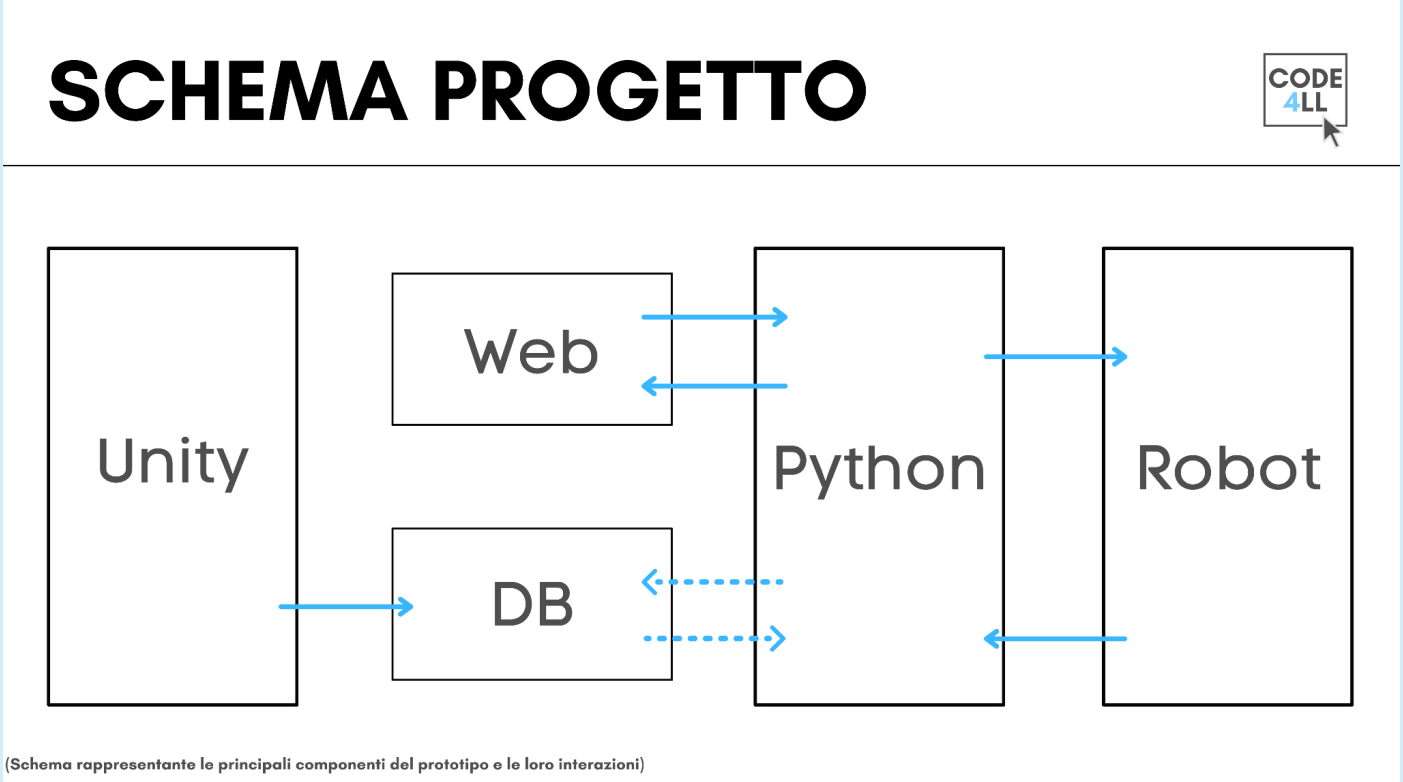
HTTP: HyperText Transfer Protocol

1.3. Campo di applicazione

In accordo con l'obiettivo prefissato, il software finale prevede un'applicazione in varie forme di magazzino o locali in cui si presenti la necessità di ridurre il lavoro assegnato all'uomo. In quanto simulazione generica, con la realizzazione di macchinari strutturalmente simili, è possibile applicare il sistema all'interno di ambbienti che prevedono il trasporto di merci di dimensioni e pesi variabili, i quali verranno accoppiati con robot in grado di effettuare tali compiti, in termini di forza e velocità.

2. DESCRIZIONE GENERALE DEL PROGETTO

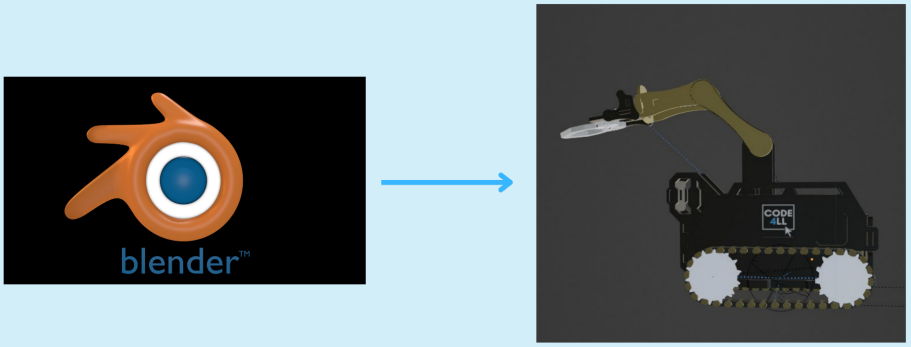
2.1. Inquadramento e rappresentazione del sistema (con immagini e/o diagrammi a blocchi) (Ravaglia, Ranieri, Calmetti, Nassif)



(Schema rappresentante le principali componenti del prototipo e le loro interazioni)

L'unione del sistema realizzato si basa sulla comunicazione tramite tabella del database, in cui risiede la coda delle operazioni.

Tramite l'applicativo Unity è realizzato il magazzino 3D all'interno del quale si può visualizzare il robot virtuale e impartire comandi



Con l'ausilio di Blender è realizzato il modellino 3D del robot, modellando i disegni 2D delle componenti ed unendole tra loro.

Il kit di Adeept permette il montaggio di un rov ruotato che comprende le caratteristiche necessarie individuate, spiegate nel dettaglio nelle prossime sezioni. Anche il codice che permette i movimenti nel robot è trattato in seguito.

(Adeept Rasptank)

2.2. Macro funzionalità

Il prodotto finale che il gruppo Code4LL intende realizzare fornisce all'utente finale una rappresentazione 3D del locale, con la quale è possibile interfacciarsi al mezzo ed impartire comandi intuitivamente, dopo averne visualizzata l'anteprima mostrata. Il programma realizzato rende facilmente fruibile il servizio anche da parte di operatori non esperti nel settore informatico, in quanto accompagnati da un'interfaccia utente che ne semplifica le interazioni con il sistema.

2.3. Caratteristiche degli utenti e vincoli di dipendenza

Gli utenti a cui è indirizzato il progetto discusso non necessitano di profonde conoscenze nei campi IT e di programmazione in quanto l'interfaccia realizzata semplifica l'utilizzo del sistema realizzato. In caso di necessità di cambiamenti nella componente fisica e virtuale del prodotto è richiesta una profonda padronanza dei linguaggi utilizzati: principalmente Python e C#, così come di componentistica in caso si ritenesse necessario alterare la struttura hardware del robot. In caso di necessità è possibile convocare nuovamente il gruppo di lavoro Code4LL per applicare l'aggiornamento necessario al sistema.

2.4. Ipotesi di partenza, assunzioni e dipendenze

L'intenzione iniziale del gruppo Code4LL prevede la realizzazione di un progetto di automazione, nello specifico nello spostamento di oggetti, che può verificarsi in qualsiasi tipo di magazzino/locale, tramite l'utilizzo di un robot in grado di muoversi autonomamente una volta ricevuti appositi input, riguardanti i pacchi/materiali da recuperare indicandone la posizione. In una futura evoluzione il robot sarà in grado di seguire un percorso indicato da una linea al suolo, raggiungendo i punti richiesti dall'operatore.

Le richieste/quest da eseguire vengono impartite tramite il simulatore 3D, realizzato con il software Unity per fornire una semplice modalità di interfaccia con il robot e per visualizzarne un'anteprima d'azione in seguito ad una fase di autenticazione.

Il robot, comunicando con il simulatore tramite un database (Mysql, istanza di MariaDB tramite Xampp), In cui vengono inserite in coda le azioni da svolgere da parte del programma Unity con il linguaggio PHP, e risposte di conferma/errore da parte del Rasptank, in base al verificarsi di situazioni impreviste o di una corretta esecuzione della richiesta tramite il codice Python con cui avviene la programmazione su Raspberry. Quanto contenuto all'interno del database può anche formare lo storico delle operazioni effettuate nell'arco di un certo periodo di tempo, come una giornata lavorativa.

L'idea di innovazione meditata, data la complessità e la quantità di tempo ridotta, è ridimensionata all'interno del progetto di realizzazione del primo prototipo, comprendente le componenti principali del sistema indipendenti.

Una volta analizzati i requisiti necessari per realizzare un macchinario efficiente e funzionante all'interno del sistema, così come per soddisfare le richieste di un ipotetico cliente, sono state individuate le capacità che il robot deve possedere, di cui 3 principali:

- Seguire una linea
- Evitare urti con oggetti o persone
- Curve agili e movimenti reattivi

Parte del gruppo si è in seguito dedicato alla ricerca dei componenti o possibili kit in grado di rispondere a tutte le esigenze individuate. Il kit di realizzazione del Rasptank di Adeept risulta essere il miglior candidato.

Nel frattempo il resto dei partecipanti ha iniziato la formazione necessaria alla realizzazione delle componenti 3D, tramite le numerose documentazioni ed esempi presenti sul Web riguardo Unity, Blender e Visual Studio per la realizzazione del codice C# che guiderà la simulazione.

In seguito all'acquisto, la costruzione del robot ha importanza primaria, in quanto da quest'ultima dipendono i lavori svolti sulle diverse componenti software del progetto: codice Python e rappresentazione 3D su Unity, insieme alla creazione del modello 3D del rov su Blender.

2.5. Sviluppi futuri

In progetti futuri, il sistema ideato può essere applicato in diversi ambiti lavorativi, sollevando l'uomo dall'incarico di muovere grandi masse e ridurne gli sforzi, oltre a poter supervisionare le posizioni dei diversi materiali soggetti all'automazione realizzata. Il robot costruito rappresenta un generico macchinario per effettuare il tipo di operazioni descritto, ciò non vieta di effettuare modifiche ad esso o al codice che lo guida, se non cambiarne completamente la struttura per gestire situazioni molto differenti che richiedono operazioni/ investimenti in termini di automazione e aumentare l'efficienza.

In futuro saranno sicuramente possibili migliorie, ottimizzazioni nel codice del robot, accompagnabili da componentistiche dotate di maggiore precisione e prestazioni in base al budget a disposizione.

3. DESCRIZIONE DEI SINGOLI SOTTOSISTEMI

3.1. Descrizione sottosistema 1: Rasptank

Marco Calmetti

RaspTank è una piattaforma robot mobile cingolata basata su Raspberry Pi, è dotata di un braccio robotico a 4 DOF in grado di afferrare piccoli oggetti. Il ROV è munito di un braccio robotico dotato di pinza. La pinza può essere aperta e chiusa per afferrare oggetti nonchè ruotata. Nella parte frontale è alloggiata una Raspberry Pi Camera e un modulo sensore ultrasuoni HC-SR04: questi due componenti sono montati su un blocco mosso da un servocomando e possono essere diretti verso l'alto o verso il basso. Sulla parte anteriore sono posti 3 fotoaccoppiatori per il riconoscimento di una linea: è quindi possibile implementare un robot di tipo Line Follower sia sfruttando i fotoaccoppiatori sia utilizzando la visione artificiale.

3.2. Descrizione sottosistema 2: Python

Ravaglia Gianluca

Python è un linguaggio di programmazione di "Alto livello", orientato agli oggetti (OOP) ed utilizzabile per sviluppare software di vario tipo, grazie ai numerosissimi pacchetti esistenti e alla imponente community che lavora dietro le quinte.

Il nostro robot necessita quindi di un programma che funga da mente, che gli consenta di reperire le informazioni e di agire di conseguenza. Essendo che il Rasptank è incentrato su un Raspberry (model Pi 4 B) il programma in questione è scritto proprio utilizzando Python.

Le parti principali del codice prevedono innanzitutto il recupero dei movimenti da effettuare da database (scritti nella apposita tabella dal codice eseguito su Unity), utilizzando il pacchetto 'pyodbc' con cui è possibile collegarsi ad una istanza di SQL Server.

In secondo luogo, 2 processi sono avviati in una fork: ovvero parallelamente. I moduli 'Threading' e 'Multiprocessing' ci consentono di eseguire in contemporanea 2 processi di codice: il primo che guida il robot nelle situazioni sotto controllo, lo conduce normalmente a spostare un ipotetico pacco e il secondo processo parallelo mantiene controllato invece il segnale del sensore ultrasonico, prevedendo un'interruzione dei compiti del robot in caso di ostacolo, utile al fine di evitare danni materiali e garantire la sicurezza del personale operativo nello stesso locale del robot.

3.3. Descrizione sottosistema 3: Unity

Ranieri Enrico

Unity è un motore grafico multipiattaforma utilizzato per lo sviluppo di giochi, applicazioni e video 3D, nello specifico noi di CODE4LL lo abbiamo usato per la gestione del magazzino virtuale.

Visual Studio è un ambiente di sviluppo integrato, multilinguaggio, sviluppato da Microsoft che supporta la creazione di progetti per varie piattaforme ed è possibile creare ed utilizzare estensioni e componenti.

C# è il linguaggio utilizzato principalmente da Unity, in quanto esso lavora con degli script di quest'ultimo che permettono di eseguire determinate azioni all'interno dell'ambiente virtuale, per cui è il linguaggio con cui abbiamo steso il codice su Visual Studio.

Nel nostro caso, Unity, oltre al simulare le azioni di spostamento pacchi eseguite dal robot, funge anche da interfaccia utente, in quanto è tramite l'applicazione Unity che l'utente seleziona le varie azioni interne al magazzino, le quali poi verrano caricate tramite visual studio sul database per poter essere reperite tramite python dal robot fisico.

3.4. Descrizione sottosistema 4: Blender

Mastropaolo Alessio, Nassif Yassine

E' un software gratuito e multipiattaforma di modellazione, rigging, animazione, montaggio video, composizione, rendering e texturing di immagini tridimensionali e bidimensionali, il quale è stato utilizzato nella creazione del modello 3D del robot e delle sue animazioni.

La realizzazione del modello è stata divisa in tre macrofasi in base al movimento autonomo di ognuna di esse: cingoli, corpo e braccio. Le misure per la realizzazione di questa parte sono state prese direttamente dal robot reale e disegnate di conseguenza in scala.

- Cingoli: Vengono disegnati i cingoli che permettono al robot di muoversi all'interno del magazzino. Prevede il movimento dei due cingoli in modo sincrono o asincrono.

-Corpo: Viene progettato il corpo del robot che permette di collegare cingoli e braccio. Inoltre è stato realizzato anche il movimento della cam del robot.

-Braccio: Viene rappresentato il braccio del robot, il quale permette lo spostamento dei pacchi all'interno del magazzino. Questa macrofase prevede l'implentazione di 3 animazioni che permettono l'elevazione, l'abbassamento e la flessione del braccio.

4. STRUMENTI PER LO SVILUPPO DELL'APPLICAZIONE

4.1. Descrizione 1: Rasptank

Marco Calmetti

Il Rasptank è un kit per la costruzione di un ROV cingolato proposto da Adeept.

Le componenti all'interno del kit sono:

Un sensore ultrasuoni tipo HC-SR04

Una Raspberry Pi Camera con cavetto flat

5 servocomandi da 9g completi di viti e squadrette (ci sono anche quelle non necessarie)

4 piccoli pcb su cui sono montati 3 led WS2812 per ciascuno

2 motoriduttori

2 portabatterie per batterie 18650

un HAT(Adeept Motor Shield) per Raspberry al quale vengono collegati tutti i componenti

i cavetti di collegamento

un cavetto di prolunga per un servo

un nastro per facilitare l'estrazione delle batterie

una brochure con le istruzioni

Caratteristiche del Rasptank:

E' dotato di un braccio robotico 4-DOF;

Riconoscimento di oggetti, rilevamento, rilevamento del movimento: basato su openCV, può tracciare oggetti di una forma o colore specifici;

Rilevamento linea: può percorrere il percorso impostato;

Trasmissione video in tempo reale: può trasferire le immagini in tempo reale scattate dalla fotocamera Raspberry Pi su un computer remoto;

Controllato a distanza tramite APP: è possibile controllare a distanza il robot tramite i pulsanti sulla tastiera o i pulsanti virtuali sulla GUI;

Dotato di 12 LED RGB seriali WS8212;

Il sistema operativo è installato all'interno del Rasptank è il Raspbian (basato su Linux).

4.2. Descrizione 2: Python

Ravaglia Gianluca

Gli strumenti utilizzati per poter realizzare il programma in python sono essenzialmente 2:

- Pycharm: tramite questo ambiente di sviluppo integrato (IDE) realizzato da JetBrains appositamente per Python, abbiamo potuto visualizzare con maggiore comodità e features il codice, potendo risalire alle definizioni di funzioni e con una migliore evidenziazione del codice nelle sue parole chiavi e classi. E' risultato quindi molto utile nello studio e nella stesura del codice.

- MobaXterm: Un SSH Client (Secure Shell Client), ovvero un software che utilizza il protocollo SSH per connettersi ad un computer remoto. Questa applicazione ci ha permesso di connetterci da remoto al Raspberry contenuto nel robot, evitando la connessione con tutte le periferiche necessarie utilizzando indirizzo IP, nome utente e password. Presenta anche un proprio IDE(molto basico) per la stesura del codice Python direttamente nei file contenuti nel robot, potendo quindi effettuare modifiche e renderle effettive nell'immediato. Una volta salvati i file, tramite linea di comando del sistema operativo basato su Debian, possiamo lanciare il file che viene eseguito, dando vita al nostro Rasptank.

4.3. Descrizione 3: Unity

Ranieri Enrico

- Unity: Unity è un motore grafico multipiattaforma utilizzato per lo sviluppo di giochi, applicazioni e video 3D il quale ci ha permesso di replicare virtualmente le azioni svolte dal robot fisico anche attraverso la creazione di animazioni, come ad esempio il movimento del braccio robotico per prendere e lasciare i pacchi.

- Visual Studio: tramite questo ambiente di sviluppo integrato (IDE) realizzato da Microsoft, il quale permette di utilizzare un numero elevatissimo di linguaggi utilizzando le apposite estensioni, è stato possibile creare un progetto multipiattaforma per la gestione del magazzino virtuale.

4.4. Descrizione 4: Blender

Mastropaolo Alessio, Nassif Yassine

Blender è un software gratuito e multipiattaforma di modellazione, rigging, animazione, montaggio video, composizione e molto altro. Dispone inoltre di funzionalità per mappature UV, simulazioni di fluidi, di rivestimenti, di particelle, altre simulazioni non lineari e creazione di applicazioni/giochi 3D. E' stato utile nella realizzazione del robot tridimensionale e delle sue animazioni (cingoli, braccio e PiCam) in modo da simulare al meglio la realtà.

5. STRUMENTI E METODI UTILIZZATI PER LA COMUNICAZIONE

5.1. Descrizione 1

La prima forma di comunicazione riguarda quella instaurata tra la rappresentazione 3D di Unity e l'istanza di MariaDB su server locale. I dati che verranno utilizzati come input per i movimenti del robot, sono generati dalla simulazione 3D una volta selezionata un'operazione da impartire al robot. Le informazioni sviluppate dal programma, tramite collegamento al database locale con apposita libreria di C#, vengono scritte nella tabella dedicata alla memorizzazione della coda delle operazioni da svolgere.

5.2. Descrizione 2

La seconda comunicazione è stabilita invece tra il database citato e il programma Python che guida il Rasptank. Similmente a quanto accade con C#, con apposita libreria: 'mysql-connector' il programma interpretato dal Raspberry è in grado di connettersi da remoto al database, Utilizzando l'indirizzo IP del computer utilizzato per ospitare l'istanza del database. I record scritti da C# nella coda delle operazioni vengono letti per essere eseguiti in maniera sequenziale, completando una task per volta fino al completo svuotamento della tabella. In seguito il programma rimane in 'polling', ovvero richiede continuamente informazioni al database, riprendendo ad operare una volta che una nuova riga è inserita nel database, ovvero una nuova operazione da svolgere.

5.3. Descrizione 3

Una terza forma è implementata tra il codice Python e il robot stesso: input digitali calcolati dal programma sono inviati ai vari sensori/motori che traducono quanto ricevuto in output reali. Avviene anche il contrario, alcuni sensori ricevono segnali analogici che vengono convertiti in digitale ed utilizzati nell'esecuzione del codice presente nella memoria del Raspberry, cuore del Rasptank.

5.4. Descrizione 4

L'ultima comunicazione è stabilita tra il programma Python e l'interfaccia web. I dati creati e scambiati tra Raspberry e sensori/motori del robot vengono inoltrati all'applicazione web tramite l'utilizzo del socket su cui è in esecuzione il processo del web server, ovvero presso l'indirizzo ip del Raspberry nella rete associato al numero di porta 5000. Questa comunicazione è stabilita solo se a conoscenza di indirizzo ip e numero di porta del server, oltre a dover far parte della medesima rete del robot.

6. PROGETTAZIONE DELL'APPLICAZIONE

6.1. Requisiti funzionali

Dopo un'attenta analisi dell'idea del progetto e delle possibili casistiche verificabili in un ipotetico ambito lavorativo si sono visualizzate le funzionalità necessarie del sistema, di cui le principali sono:

- Interfaccia web per consentire ad un operatore di comandare a distanza il rov, presente sulla stessa rete del dispositivo da cui è acceduta l'applicazione web. Questo deve consentire di svolgere ogni tipo di movimento e di selezionare alcune particolari modalità automatiche, come 'trackline' Per seguire automaticamente una linea posta sul suolo.
- Robot fisico che presenti le seguenti componenti:
  - Braccio mobile.
  - "Mano" in grado di afferrare oggetti.
  - Sensori ad ultrasuoni frontali per non urtare cose e persone.
  - Sensori a infrarossi diretti verso il suolo che possano rilevare una differente riflessione della luce, individuando così un tracciato predefinito.
  - Ruote o cingoli che permettano il movimento del dispositivo
  - Unità di elaborazione centrale(individuata in un Raspberry PI 4)
  - Fotocamera per possibili sviluppi futuri, permettendo un riconoscimento di movimenti o gesti.
- Rappresentazione 3D su Unity del magazzino/ambiente di lavoro con:
  - Locale chiuso.
  - Scaffali (su cui verranni posati i materiali)
  - Modellino del robot che rispetti le reali proporzioni e componenti di quest'ultimo (Realizzato tramite Blender ed importato su Unity)
  - Animazioni del robot che mostrino l'anteprima delle operazioni che saranno svolte effettivamente dall'hardware del sistema come: movimenti nel magazzino, curve, presa e rilascio dei pacchi sugli scaffali.
- Database finalizzato alla memorizzazione di:
  - Informazioni riguardanti i pacchi come i materiali contenuti e dimensioni
  - Coda delle operazioni impartite dall'app. 3D, eseguite in maniera sequenziale una volta prelevate dal programma Python le righe del database.
  - Tabella di Log in cui sono memorizzati i dati riguardanti le operazioni svolte durante la giornata e codici di stato sulle operazioni, rappresentanti una corretta o errata esecuzione delle operazioni.

6.2. Requisiti non funzionali

Per quanto riguarda i requisiti non funzionali, ovvero quei criteri che possono fornire una valutazione al sistema, sono individuati i principali punti, raggruppati nelle aree indicate:

- **Sicurezza**(informatica e del personale): Il sistema deve essere sicuro ed accessibile solo da personale autorizzato, è necessario proteggere l'accesso a distanza ai comandi del robot per evitarne usi non previsti e pericolosi e limitando l'utilizzo ai soli dispositivi presenti all'interno della rete, impedendo a chiunque si trovi nella rete pubblica internet di accedere al robot, di cui è necessario conoscere il socket su cui fornisce interfaccia web. Anche l'utilizzo della simulazione 3D è previsto solo in seguito al superamento di un form di autenticazione, il quale richiede email e password personali. Per garantire sicurezza sul posto di lavoro anche nel caso di modalità automatiche impartite al robot, quest'ultimo deve essere in grado di individuare possibili ostacoli(cose e persone) al fine di evitare danni alla produzione aziendale o al personale impiegato nel locale di collocazione del sistema, rispettando così le leggi vigenti di sicurezza sul lavoro.
- **Adattabilità**: per un semplice e maggiore impiego futuro, il sistema realizzato da Code4LL risulta altamente scalabile, composto da diverse componenti indipendenti tra loro e facilmente integrabili. La

rappresentazione 3D può essere modificata al fine di mostrare un diverso ambiente di lavoro ed è possibile importare modellini di qualsiasi tipo in caso risultasse differente il macchinario installato da rappresentare virtualmente. Il codice Python deve essere modificabile con approfondita conoscenza di quest'ultimo e dei linguaggi web utilizzati. L'utilizzo di un database intermedio è la chiave di questa ampia integrabilità di componenti differenti tra loro: molte tecnologie supportano la possibilità di connettersi a database di vari generi, costituendo come uno standard per la memorizzazione di informazioni, permettendo lo scambio di esse, per esempio, tra programmi scritti in linguaggi completamente diversi tra loro.

- **Prestazioni**: Il sistema deve iniziare l'esecuzione di nuove operazioni eseguite dall'utente nel giro di poco tempo, non appena il programma legge nuove informazioni interne al database e le utilizza come input dei movimenti del robot all'interno dell'ambiente.
- **Disponibilità**: Tramite connessione alla medesima rete locale (se in possesso di credenziali di autorizzazione) deve sempre essere possibile connettersi al robot ed alla sua interfaccia web, tramite cui interagire. In caso di problemi riguardanti la connessione ad internet si vuole realizzare un robot a cui sia possibile connettersi wireless in assenza di connessione ad internet, creando una rete locale in cui il gateway è il robot stesso.
- **Manutenibilità**: Il codice, commentato in modo appropriato e seguendo le best practices standard della programmazione deve risultare comodamente modificabile al fine di modificare graficamente le interfacce presenti o cambiarne la logica applicativa per possibili evoluzioni o applicazioni in ambienti di utilizzo differenti.
- **Portabilità**: Il sistema dovrà prevedere la possibilità di essere utilizzato su piattaforme differenti tra loro, per cui l'utilizzo di una web app ne risulta perfetto al fine di non avere vincoli in termini di fascia hardware o sistema operativo utilizzato. Necessaria è soltanto la connessione ad internet o al robot in caso quest'ultima fosse assente.

6.3. Struttura dell'applicazione

L'applicazione, indirizzata ad automatizzare il lavoro all'interno di un generico magazzino, è formata da 3 componenti principali, composte dalle caratteristiche che seguono:

- Rappresentazione 3D: si intende realizzare tramite l'applicativo Unity una Simulazione 3D di quanto dovrà svolgere il robot e per impartire le operazioni ad esso. Cliccando su appositi spazi, programmati con funzioni di tipo Onclick, è possibile muovere il robot virtuale e scrivere sulla tabella dedicata di un database i dati necessari per far svolgere la stessa azione al robot. Tramite il programma blender è realizzato il modello 3D del robot, partendo da fogli di Autocad su cui sono disegnate le componenti assemblate, queste sono modellate e rese rispettose delle reali proporzioni del rov. La simulazione prevede un form di autenticazione tramite cui un ipotetico operatore deve identificarsi per avere accesso all'applicativo in grado di comandare l'hardware realizzato. I dati richiesti sono email e password personali, memorizzati su Firebase, integrato nel progetto Unity per la realizzazione dell'interfaccia iniziale.
- Database: Il database intermedio è la componente fondamentale per stabilire una comunicazione tra la componente virtuale e reale del progetto. Tralasciando le entità dedicate alla sola memorizzazione di informazioni riguardanti i pacchi e la loro posizione, una tabella dedicata, denominata 'Operation', memorizza le informazioni riguardanti le operazioni da svolgere, formando quindi una coda delle operazioni da svolgere, in quanto la componente reale non è sincronizzata nei movimenti con quella reale. Una volta che le informazioni sono prelevate dallo script python, le righe della tabella sono eliminate. La query di delete eseguita avvia un trigger memorizzato nel database, il quale aggiunge nella tabella 'OperationLog' una riga che descrive l'operazione eseguita e se questa è andata a buon fine. Questa tabella funge appunto da log delle operazioni, mantiene memorizzate le azioni svolte per un certo arco di tempo, come una giornata di lavoro.
- Robot: è assemblato il kit fornito dalla ditta cinese Adeept, il quale corrisponde al migliore compromesso per rispettare le necessità individuate in fase di progettazione. Il rov costruito è guidato dal codice Python memorizzato sulla scheda SD del Raspberry. Il programma, tramite modulo 'mysql-connector' è in grado di connettersi al database e leggere i record presenti nella tabella delle operazioni del database mysql. In seguito, ottenendo i dati scritti in precedenza da Unity, li traduce

nelle proprie unità di misura per realizzare input adatti per i vari sensori, motori o servo di cui è composto il robot.

- Tramite stampante 3D si intende realizzare dei piccoli ripiani per poter simulare la disposizione nel magazzino, all'interno del quale è tracciata una linea sul suolo, realizzato su un foglio di carta ampio a sufficienza per poter dimostrare quanto può svolgere il sistema realizzato.

7. PROGETTAZIONE DEL DATABASE

7.1. Analisi

Si intende realizzare un database che memorizzi le informazioni riguardanti i contenuti del magazzino automatizzato. Le informazioni necessarie riguardano i pacchi contenuti sui ripiani, ID univoco del pacco, dimensioni, peso e breve descrizione del contenuto. Ognuno di essi contiene un riferimento al ripiano in cui si trovano, record di un'altra tabella con cui si instaura una relazione di tipo 1:N.

Ulteriori 2 tabelle sono utilizzate per la condivisione delle operazioni da svolgere al robot e per realizzare il log delle operazioni svolte nel corso di un certo arco di tempo, come una giornata lavorativa e il relativo codice di stato dell'operazione che indica il corretto o errato svolgimento dell'azione.

Sono quindi individuate 4 entità da implementare nel database:

- Package:
  L'entità Package rappresenta un pacco contenuto all'interno del magazzino, è caratterizzata da: ID univoco, peso(kg), dimensioni (CmxCmxCm), descrizione del contenuto e chiave esterna con riferimento alla chiave primaria del ripiano.
- Shelf:
  La tabella shelf riguarda i singoli ripiani degli scaffali nel magazzino e contiene le seguenti informazioni: ID univoco, lato magazzino (Right/Left), fila scaffale (Bottom/Center/Up), lato scaffale(Bottom/Up) e posizione all'interno della fila (0÷8).
- Operation:
  Contiene la coda delle operazioni impartite dal programma su Unity al robot, contiene un ID univoco dell'operazione e le coordinate in cui svolgere l'operazione, similmente a quanto contenuto nell'entità 'Shelf'
- OperationLog:
  Contiene lo storico delle operazioni svolte: Oltre ad ID identificativo, contiene un campo testuale in cui sono concatenate le informazioni eliminate da una riga in 'operation' una volta completata, così da formare piccoli testi che descrivano lo svolgimento dell'operazione.

Vincoli:

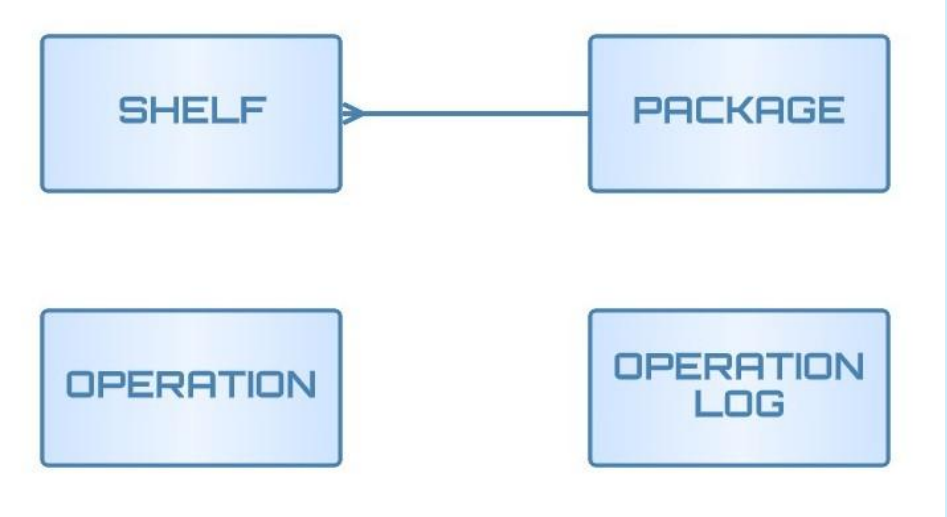
Oltre ai vincoli di integrità referenziale tra foreign key e primary key delle tabelle sono presenti i seguenti vincoli sui campi indicati:

- Lato magazzino: Solo valori 'R', 'L' e 'S' (Right, Left e Start)
- Fila scaffale: Solo valori 'B', 'C', 'U', 'S' (Bottom, Center, Up, Start)
- Lato scaffale: Solo valori 'B', 'U', 'S' (Bottom, Up, Start)
- Punto: Solo valori compresi tra 0 e 8
- Peso: <= a 100 KG (peso massimo simbolico in accordo con le capacità del robot)

Relazioni:

All'interno del database del progetto è attualmente presente una singola relazione, di tipo 1-1 tra i pacchi e i ripiani, infatti il pacco contiene una chiave esterna che fa riferimento alla primary key dell'entità ripiano, così da poter ottenere tramite query le informazioni riguardanti la posizione tramite Join effettuata sulla corrispondenza delle chiavi. Tra la tabella delle operazioni da effettuare e il Log delle operazioni non è presente alcuna relazione, si tratta solo di una copia dei dati, concatenati in una stringa per fornire una descrizione di quanto avvenuto nell'arco della giornata e a quali orari.

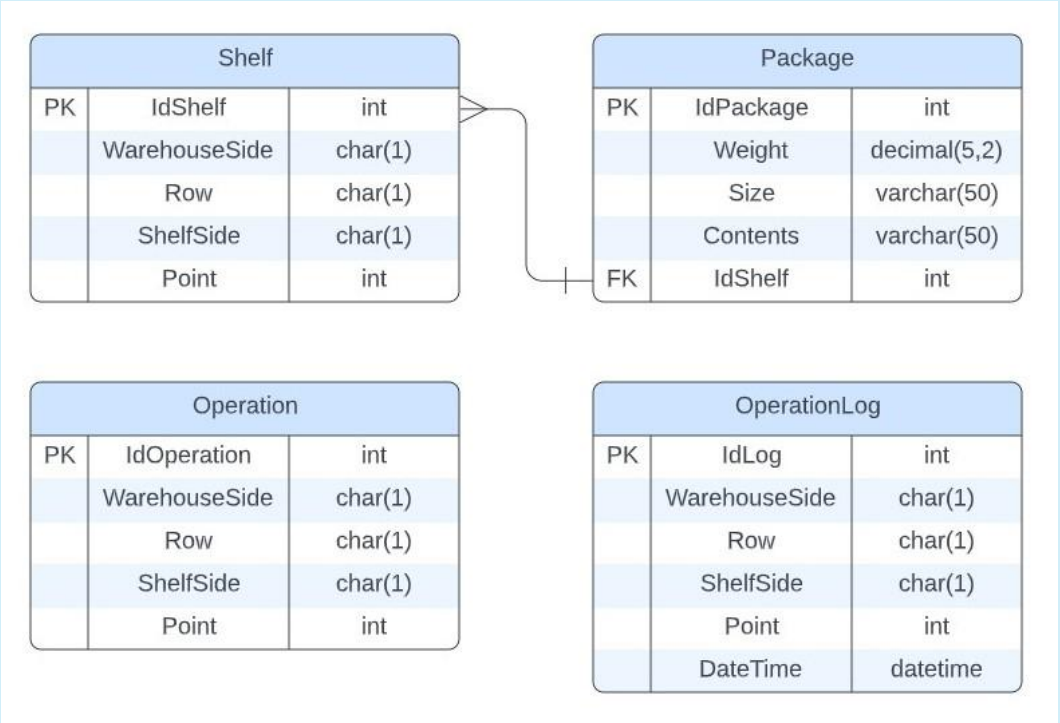
7.2. Modello E-R



7.3. Dizionari

Entità	Descrizione	Attributi	Identificatore
Package	Rapperesenta un pacco contenuto all'interno del magazzino	Weight, Size, Contents, IdShelf	IdPackage
Shelf	Rappresenta il singolo ripiano dello scaffale nel magazzino	WarehouseSide, Row, ShelfSide, Point	IdShelft
Operation	Rappresenta l'operazione impartita da Unity al robot	WarehouseSide, Row, ShelfSide, Point	IdOperation
OperationLog	Rappresenta l'operazione già svolta	WarehouseSide, Row, ShelfSIde, Point, DateTime	IdLog

7.4. Schema logico



7.5. Creazione tabelle in SQL

```sql
CREATE TABLE Shelf(
   IdShelf INT AUTO_INCREMENT,
   WarehouseSide CHAR(1) NOT NULL,
   Row CHAR(1) NOT NULL,
   ShelfSide CHAR(1) NOT NULL,
   Point INT NOT NULL,
   CONSTRAINT PK_Shelf_IdShelf PRIMARY KEY (IdShelf),
   CONSTRAINT CK_Shelf_WarehouseSide CHECK(WarehouseSide IN('R', 'L', 'S')),
   CONSTRAINT CK_Shelf_Row CHECK(Row IN('B', 'C', 'U', 'S')),
   CONSTRAINT CK_Shelf_ShelfSide CHECK(ShelfSide IN('S', 'B', 'U')),
   CONSTRAINT CK_Shelf_Point CHECK(Point >= 0 AND Point < 9)
);

CREATE TABLE Package(
   IdPackage INT AUTO_INCREMENT,
   Weight DECIMAL(5,2) NOT NULL,
   Size VARCHAR(50) NOT NULL,
   Contents VARCHAR(50) NOT NULL,
   IdShelf INT,
   CONSTRAINT PK_Package_IdPackage PRIMARY KEY (IdPackage),
   CONSTRAINT FK_Package_IdShelf_Shelf FOREIGN KEY (IdShelf) REFERENCES Shelf(IdShelf),
   CONSTRAINT CK_Package_Weight CHECK(Weight > 0 AND Weight <= 100)
);

CREATE TABLE Operation(
   IdOperation INT AUTO_INCREMENT,
   WarehouseSide CHAR(1) NOT NULL,
   Row CHAR(1) NOT NULL,
   ShelfSide CHAR(1) NOT NULL,
   Point INT NOT NULL,
   CONSTRAINT PK_Operation_IdOperation PRIMARY KEY (IdOperation),
   CONSTRAINT CK_Operation_WarehouseSide CHECK(WarehouseSide IN('R', 'L', 'S')),
   CONSTRAINT CK_Operation_Row CHECK(Row IN('B', 'C', 'U', 'S')),
   CONSTRAINT CK_Operation_ShelfSide CHECK(ShelfSide IN('S', 'B', 'U')),
   CONSTRAINT CK_Operation_Point CHECK(Point >= 0 AND Point < 9)
);
```

```
CREATE TABLE OperationLog(
   IdLog INT AUTO_INCREMENT,
   WarehouseSide CHAR(1) NOT NULL,
   Row CHAR(1) NOT NULL,
   ShelfSide CHAR(1) NOT NULL,
   Point INT NOT NULL,
   DateTime DATETIME NOT NULL,
   CONSTRAINT PK_OperationLog_IdLog PRIMARY KEY (IdLog),
   CONSTRAINT CK_Operation_WarehouseSide CHECK(WarehouseSide IN('R', 'L', 'S')),
   CONSTRAINT CK_Operation_Row CHECK(Row IN('B', 'C', 'U', 'S')),
   CONSTRAINT CK_Operation_ShelfSide CHECK(ShelfSide IN('S', 'B', 'U')),
   CONSTRAINT CK_Operation_Point CHECK(Point >= 0 AND Point < 9)
);

CREATE TRIGGER triggerLog AFTER DELETE ON Operation FOR EACH ROW
   INSERT INTO OperationLog(WarehouseSide, Row, ShelfSide, Point, DateTime)
   VALUES (OLD.WarehouseSide, OLD.Row, OLD.ShelfSide, OLD.Point, CURRENT_TIMESTAMP)
```

8. SICUREZZA ED AUTENTICAZIONE

8.1. Analisi del sistema di autenticazione

Un form di autenticazione viene mostrato all'accesso della simulazione 3D, realizzato integrando Firebase a Unity, all'interno del quale sono memorizzati gli utenti autorizzati ad accedere all'applicazione una volta inseriti email e password personali, fornendo appositi output in caso di dati mancanti o errati.

8.2. Crittografia

La crittografia è applicata nella connessione remota al robot tramite l'SSH client MobaXTerm: la connessione avviene conoscendo l'ip del dispositivo all'interno della rete locale, username e password impostati all'interno del sistema operativo del Raspberry. La comunicazione viene stabilita tramite il protocollo SSH sulla well-known port 22 del protocollo TCP, per cui la comunicazione avviene in maniera sicura, crittografando i messaggi tramite SSL/TLS (Secure Socket Layer/Transport Layer Security).

Anche Firebase applica una forma di crittografia, nascondendo i dati con apposite funzioni di hash, fornendo una chiave personale in base 64. In questo modo i dati sono visualizzabili in chiaro se in possesso dei permessi necessari, altrimenti i dati non sono leggibili in quanto crittografati con la funzione di hash citata.

8.3. Ulteriori specifiche

Ulteriori applicazioni di sicurezza sono presenti nella connessione all'interfaccia web ottenuta da Adeept. Per poter utilizzare l'applicazione web è necessario far parte della medesima LAN e conoscere indirizzo IP e numero di porta del web server, che in questo caso coincide con la porta 5000.

9. CONCLUSIONI

9.1. Riflessione generale sull'attività del progetto

L'attività di progetto svolta al fine di partecipare alla borsa di studio offerta da Xion Technology S.R.L si è rivelata molto formativa per i membri del gruppo Code4LL, nome coniato proprio per questa occasione.

Il progetto ha dato ad ogni componente la possibilità di comprendere, di entrare nell'ottica di quelle che sono le caratteristiche, ma soprattutto le difficoltà da affrontare nello sviluppo del software.

In primo luogo l'approccio al lavoro in team, modalità di lavoro a cui l'ambiente scolastico indirizza relativamente poco. Comunicare, collaborare ed adattarsi ai ritmi degli altri è fondamentale nel mondo IT, così come in gran parte delle realtà lavorative. Entrare in sintonia con persone a cui raramente si rivolge la parola o con approcci allo studio diametralmente opposti sono fondamentali per instaurare un ambiente di lavoro produttivo e piacevole.

Abbiamo potuto comprendere quanto sia importante dare il proprio contributo al fine di realizzare un ampio progetto e molto vario nelle sue componenti, dalla programmazione pura alla sistemistica, dalla realizzazione di modelli 3D alle loro animazioni. Quando le realizzazioni di queste componenti dipendono tra loro si può capire come sia importante rispettare le proprie date di scadenza, al fine di non intaccare i tempi di lavoro dei compagni di gruppo.

Ci siamo resi conto soprattutto di come la realizzazione di software a progetto sia la migliore forma per imparare le nozioni necessarie nell'ambito informatico: ricercare qualcosa di sconosciuto e tentare l'applicazione fino al funzionamento permette di interiorizzare al meglio la logica di quanto realizzato. L'istruzione scolastica fornisce poche informazioni riguardo argomenti così specifici, per cui ideare un sistema, senza sapere esattamente come funziona rende molto complessa una programmazione corretta di quanto si vuole realizzare nei limiti di tempo prestabiliti. Tutto ciò alimenta l'ansia, il timore di non riuscire a completare per tempo il progetto del gruppo: reggere mentalmente le situazioni di stress è molto importante in azienda, soprattutto per chi ricopre ruoli di supervisione o management di gruppi di lavoro su cui ricadono pressochè tutte le responsabilità, soprattutto in caso di fallimento del progetto o di mancato rispetto dei vincoli economici e temporali prestabiliti.

10. ALLEGATI

10.1. Eventuali file allegati al documento (organigramma, codice etico di comportamento, WBS progetto)

Codice etico Code4LL

Organigramma gruppo Code4LL

Logo Code4LL

Logo negativo Code4LL

WBS progetto