Università degli Studi di Napoli Parthenope

PROGETTO DI RETI DI CALCOLATORI

"UNIVERSITÀ"

Marika Esposito 0124002508

Diego Pennacchio 0124002673

Valerio Piccolo 0124002597

CODICE GRUPPO: hmvuo20h9bv

Docenti: Prof. Emanuel Di Nardo

A.A. 2023/2024

SOMMARIO

CAPITOLO 1

DESCRIZIONE DEL PROGETTO

Introduzione

Il progetto si focalizza sulla progettazione e sviluppo di un'applicazione client/server parallelo per la gestione di esami universitari. L'obiettivo principale è quello di creare un sistema efficiente che faciliti la comunicazione tra la segreteria, gli studenti e il server universitario, consentendo una gestione ottimale degli esami universitari.
Per sviluppare ed implementare il codice abbiamo utilizzato il linguaggio di programmazione Phyton.

Contestualizzazione

Nel contesto universitario, la gestione degli esami è cruciale per garantire un flusso ordinato di informazioni tra la segreteria e gli studenti. Il progetto mira a fornire una soluzione pratica ed affidabile, per soddisfare le esigenze sia da parte del client che del server.

Obiettivi

Gli obiettivi fondamentali del progetto sono molteplici e comprendono:

- **Interfaccia Intuitiva per la Segreteria**: Creare un'interfaccia utente intuitiva perla segreteria, consentendo un facile inserimento e gestione delle informazioni sugli esami. Questo aspetto è fondamentale per garantire un'efficace interazione con il sistema.
- **Verifica e Prenotazione per gli Studenti:** Fornire agli studenti la possibilità di verificare la disponibilità degli esami e di effettuare prenotazioni in modo chiaro e intuitivo. Questo processo contribuisce a semplificare la pianificazione accademica degli studenti.
- **Implementazione di un Server Universitario Robusto**: Creare un server universitario robusto, in grado di gestire con efficienza e affidabilità le richieste provenienti sia dalla segreteria che dagli studenti. Un server solido è essenziale per garantire la continuità operativa del sistema.

CAPITOLO 2

DESCRIZIONE E SCHEMA DELL'ARCHITETTURA

Architettura del Sistema

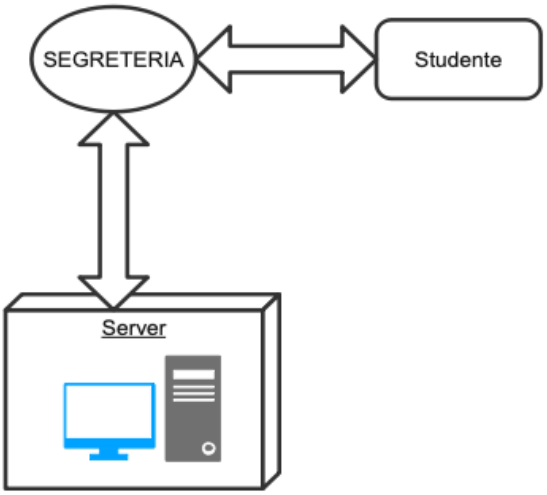
Il cuore del progetto risiede nell'architettura del sistema, concepita con cura per garantire un ambiente fluido e interattivo. Il modello adottato, come abbiamo già detto in precedenza, è basato su un'architettura client/server che sfrutta le *socket*, promuovendo la connettività e la collaborazione tra le diverse componenti. In particolare, la segreteria, gli studenti e il server universitario sono i principali attori coinvolti, ciascuno con ruoli distinti ma interconnessi.

Lo schema architetturale è basato su tre entità:

- **Il server dell'università**
- **La segreteria**
- **Lo studente**

Diagramma dell'Architettura

Per fornire una comprensione visiva dettagliata dell'architettura, è stato sviluppato un diagramma che esplicita chiaramente le interazioni tra le diverse componenti del sistema. Il diagramma illustra il flusso delle informazioni durante le diverse fasi della gestione degli esami, evidenziando le connessioni e i protocolli utilizzati.
Ogni componente è rappresentata in modo distintivo, evidenziando le operazioni svolte e le interazioni con le altre parti del sistema. Questo diagramma fornisce una visione intuitiva dell'architettura complessiva, agevolando la comprensione del funzionamento del sistema.

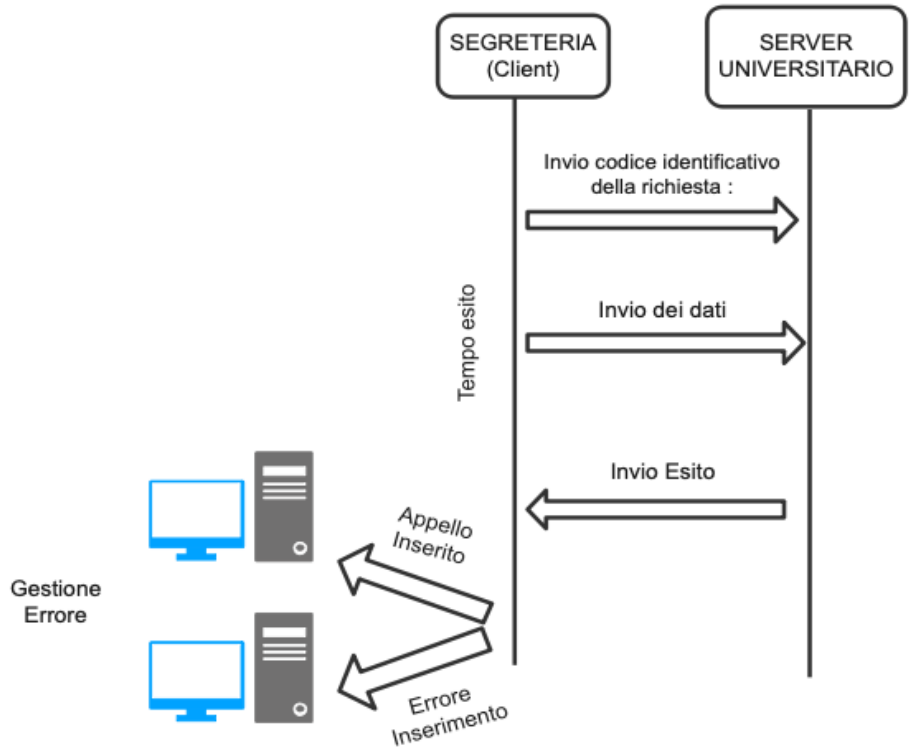


CAPITOLO 3

DETTAGLI IMPLEMENTATIVI DEI CLIENT/SERVER

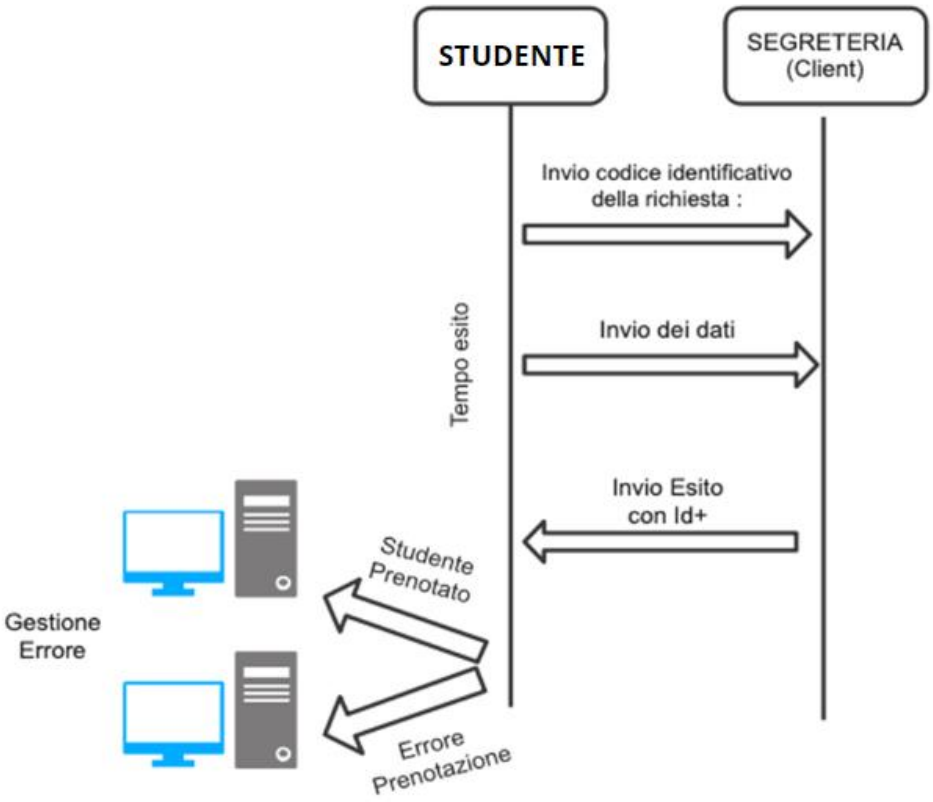
Primo Passo:

L'operazione inizia con la segreteria che sceglie quale operazione eseguire. Al server universitario vengono inviati il nome e la data dell'esame da inserire dell'appello. Successivamente il server dell'università riceve i dati e restituisce alla segreteria l'esito dell'operazione, qualsiasi sia il suo esito.



Secondo Passo:

L'operazione inizia con lo studente che sceglie il tipo di operazione. Alla segreteria vengono inviati i dati relativi all'operazione scelta dallo studente. Successivamente questi dati vengono inviati al server dell'università che genera un codice ID progressivo. Successivamente il server re-invia il codice ID alla segreteria che a sua volta lo invia nuovamente allo studente.



CAPITOLO 4

PARTI RILEVANTI DEL CODICE SVILUPPATO

Operazione 1:

L'operazione inizia con la segreteria che sceglie il tipo di operazione. Al server viene inviato il nome dell'esame da inserire tra gli appelli. Successivamente il server riceve i dati e sono disponibili per il proseguo delle operazioni.

PARTE DI CODICE CHE INCLUDE QUESTE FUNZIONALITÀ:

```
    #Metodo per inviare una richiesta al server
    def send_request(self, request):
        if self.client_socket is None:
            #Verifica se il client è connesso
            print("Non si riesce a connettere al server.")
            return

       #Invia la richiesta al server e riceve la risposta
        self.client_socket.sendall(request.encode())

        response = self.client_socket.recv(1024)
        print(response.decode())

    #Metodo per aggiungere un nuovo esame
    def add_exam(self, exam_name):
        request = f"ADD_EXAM {exam_name}" #Costruzione della richiesta
        self.send_request(request) #Invio della richiesta al server

    #Metodo per inoltrare la richiesta di prenotazione di un esame
    def forward_booking_request(self, exam_name, student_name):
        request = f"BOOK_EXAM {exam_name} {student_name}"
        self.send_request(request)

    #Metodo per inoltrare la richiesta degli esami disponibili
    def forward_available_exams_request(self):
        request = "AVAILABLE_EXAMS"
        self.send_request(request)

   #Metodo per inoltrare la richiesta di disponibilità di un esame
    def forward_exam_availability_request(self, exam_name):
        request = f"CHECK_EXAM_AVAILABILITY {exam_name}"
        self.send_request(request)

   #Metodo per chiudere la connessione
    def close_connection(self):
        self.client_socket.close()

#Funzione per avviare il client della segreteria
if __name__ == "__main__":
    secretary_client = SecretaryClient()
    secretary_client.connect() #Connessione al server
```

Operazione 2:

L'operazione inizia con lo studente che sceglie il tipo di operazione. Alla segreteria vengono inviati i dati relativi all'operazione scelta dallo studente. Successivamente questi dati vengono inviati al server dell'università che genera un codice ID progressivo. Successivamente il server re-invia il codice ID alla segreteria che a sua volta lo invia nuovamente allo studente.

PARTE DI CODICE CHE INCLUDE QUESTE FUNZIONALITÀ:

```c
void book_exam(char *studentName, char *examName, char *response) {
    if (booking_count < MAX_BOOKINGS) {
        bookings[booking_count].id = booking_count + 1; // ID progressivo
        strcpy(bookings[booking_count].studenteName, studentName);
        strcpy(bookings[booking_count].examName, examName);
        booking_count++;

        sprintf(response, "Prenotazione esame effettuata. ID Prenotazione: %d\n", bookings[booking_count - 1].id);
    } else {
        strcpy(response, "Non è possibile prenotare ulteriori esami.\n");
    }
}
```

Questo frammento di codice serve per gestire e inoltrare specifiche richieste della segreteria, evitando di eseguire ulteriori operazioni sulle richieste che soddisfano questa condizione.

```python
if request.startswith("SECRETARY_"):
    # Inoltra la richiesta della segreteria
    self.forward_secretary_request(request)
    continue  # Salta il resto del loop per attendere la risposta
```

CAPITOLO 5

MANUALE UTENTE CON LE ISTRUZIONI SU COMPILAZIONE ED ESECUZIONE

Istruzioni per la compilazione:

Per avviare correttamente il progetto, segui attentamente i seguenti passaggi:

1. **Apertura del Terminale:** Apri il terminale sul tuo sistema operativo. Assicurati di avere privilegi sufficienti per eseguire comandi come amministratore, se necessario.
2. **Apertura di Tre Finestre del Terminale:** Apri tre finestre del terminale sul tuo desktop o sistema operativo. Questo può essere fatto solitamente facendo clic destro sull'icona del terminale e selezionando "Apri nuova finestra" o utilizzando il menu del tuo sistema operativo.
3. **Avvio dei File Python in Ordine:** Avvia i seguenti file Python nell'ordine specificato:
   - **server.py:** Nella prima finestra del terminale, posizionati nella directory in cui è presente il file server.py utilizzando il comando cd seguito dal percorso della directory. Una volta all'interno della directory, esegui il comando:
   - python3 server.py
     Questo avvierà il server.
   - **segreteria.py:** Nella seconda finestra del terminale, posizionati nella stessa directory in cui è presente il file segreteria.py utilizzando il comando cd seguito dal percorso della directory. Una volta all'interno della directory, esegui il comando:
   - python3 segreteria.py

     Questo avvierà il modulo della segreteria.

   - **studente.py:** Nella terza finestra del terminale, posizionati nella stessa directory in cui è presente il file studente.py utilizzando il comando cd seguito dal percorso della directory. Una volta all'interno della directory, esegui il comando:
   - python3 studente.py

     Questo avvierà il modulo studentesco.