|  |  |
| --- | --- |
| Immagine che contiene testo, Elementi grafici, design  Il contenuto generato dall'IA potrebbe non essere corretto.  **Manuale tecnico** | Università degli Studi dell’Insubria – Laurea Triennale in Informatica  a.a. 2024/2025  *Progetto Laboratorio B*  **Sviluppato da:**  **ERICA FACCIO,**  matricola n° 751654, Varese  **GIOVANNI ISGRO',**  matricola n° 753536, Varese  **MATTIA SINDONI,**  matricola n° 750760, Varese |

**Sommario**

[1. Introduzione 2](#_Toc207460707)

[2. Architettura generale 2](#_Toc207460708)

[3. Requisiti tecnici 2](#_Toc207460709)

[4. Struttura del progetto 3](#_Toc207460710)

[5. Database 3](#_Toc207460711)

[6. Protocollo Client-Server 5](#_Toc207460712)

[7. Sicurezza 5](#_Toc207460713)

[8. Installazione e configurazione 5](#_Toc207460714)

[9. Manutenzione 6](#_Toc207460715)

# **1. Introduzione**

Il manuale tecnico di TheKnife è rivolto a sviluppatori e amministratori di sistema che necessitano di comprendere a fondo il funzionamento della piattaforma e contiene informazioni su architettura, protocollo di comunicazione, database, sicurezza e manutenzione.

L’obiettivo è garantire che chiunque debba installare, configurare o estendere il sistema disponga di tutte le informazioni necessarie.

# **2. Architettura generale**

TheKnife segue un’architettura **client-server**.

* **Client**: applicazione JavaFX che gestisce l’interfaccia grafica
* **Server**: applicazione Java che gestisce la logica applicativa
* **Database**: PostgreSQL, contenente tabelle relazionali normalizzate.

La comunicazione tra client e server avviene tramite socket TCP e protocollo testuale.

# **3. Requisiti tecnici**

* **Software**: Java 17+, Maven 3.8+, PostgreSQL 13+
* **Hardware**: minimo 4 GB RAM, processore dual-core, 2 GB spazio libero; consigliato: 8 GB RAM, CPU quad-core, SSD
* **Ambiente di sviluppo consigliato**: IntelliJ IDEA o Eclipse con supporto Maven

# **4. Struttura del progetto**

Il progetto è suddiviso nei seguenti moduli:

* **theknife\_client**: include interfaccia grafica (FXML), controller JavaFX, classe Communicator per la comunicazione con il server
* **theknife\_server**: include gestione client con thread dedicati, DBHandler per interazione col database, classe User per la gestione della sicurezza
* **doc/javaDoc**: documentazione automatica generata con Javadoc

# **5. Database**

Tabelle principali:

* **utenti**: memorizza dati anagrafici, credenziali, ruolo
* **ristoranti**: contiene le informazioni sui locali e il riferimento al proprietario
* **recensioni**: contiene le recensioni elaborate dagli utenti e sono collegate sia a utenti sia a ristoranti
* **risposte**: contiene eventuali risposte dei ristoratori alle recensioni
* **preferiti**: associazione utente–ristorante

Ogni tabella è provvista di chiavi primarie e relazioni tramite chiavi esterne.

Ogni ristorante è associato a un utente proprietario.

Ogni recensione è collegata a un utente e a un ristorante.

Ogni risposta è collegata a una recensione.

Ogni preferito è un'associazione utente-ristorante.

[INSERIRE: DIAGRAMMA ARCHITETTURA GENERALE]

[INSERIRE: DIAGRAMMA ER DATABASE]

# **6. Protocollo Client-Server**

Le comunicazioni avvengono tramite socket TCP e messaggi testuali. Ogni messaggio è costituito da un comando seguito da eventuali parametri. Le risposte del server includono codici standardizzati.

|  |  |  |  |
| --- | --- | --- | --- |
| **Comando** | **Parametri** | **Risposta** | **Note** |
| login | username, password | ok | username | password | Autenticazione utente |
| register | dati anagrafici, ruolo | ok | missing | error | Crea nuovo account |
| getRestaurants | filtri, pagina | elenco ristoranti | Ricerca filtrata con paginazione |
| getReviews | idRistorante, pagina | elenco recensioni | Recensioni ristorante |
| addReview | idRistorante, stelle, testo | ok | unauthorized | not found | Aggiunta recensione |
| addRestaurant | dettagli ristorante | ok | missing | price\_format | Solo ristoratore |

# **7. Sicurezza**

Le password sono cifrate utilizzando l'algoritmo BCrypt. Il server effettua controlli di validità sui dati ricevuti dal client (campi obbligatori, controlli sul formato numerico e sui permessi di accesso).

# **8. Installazione e configurazione**

Per installare e configurare TheKnife:

1. Creare il database con init-db.sql
2. Compilare il server con Maven e avviare il JAR indicando parametri di porta e credenziali DB
3. Configurare il file config.ini del client con indirizzo IP e porta del server
4. Avviare il client tramite file JAR.

# **9. Manutenzione**

La manutenzione prevede:

* Backup periodici con pg\_dump
* Monitoraggio dei log lato server per individuare anomalie
* Aggiornamento del database in caso di nuove versioni tramite script SQL incrementali
* Controllo performance