

****

**COTS & Design Pattern Document**

| **Riferimento** | C02\_Cots&DesignPatternDocument |
| --- | --- |
| **Versione** | 0.4 |
| **Data** | 06/01/2025 |
| **Destinatario** | Prof.ssa F.Ferrucci, Prof F.Palomba |
| **Presentato da** | B. Gallo, A. De Filippo, G. Montella, M. Zurolo, B. Colella, A. Ceruso, V. Vernellati |
| **Approvato da** | L. Todisco, C. Venditto |

Storia delle revisioni

| **Data** | **Versione** | **Descrizione** | **Autori** |
| --- | --- | --- | --- |
| 19/11/2024 | 0.1 | Prima Stesura | BC |
| 25/11/2024 | 0.2 | Stesura Design Pattern | BG, BC, ADF, GM, MZ, AC |
| 26/11/2024 | 0.3 | Stesura COTS | VV |
| 06/01/2025 | 0.4 | Revisione generale | MZ, GM |

Membri del team

| **Nome** | **Acronimo** | **Informazioni di contatto** |
| --- | --- | --- |
| Leopoldo Todisco | LT | l.todisco4@studenti.unisa.it |
| Carlo Venditto | CV | c.venditto@studenti.unisa.it |
| Biagio Gallo | BG | b.gallo9@studenti.unisa.it |
| Geraldine Montella | GM | g.montella19@studenti.unisa.it |
| Alessia De Filippo | ADF | a.defilippo31@studenti.unisa.it |
| Mario Zurolo | MZ | m.zurolo3@studenti.unisa.it |
| Vito Vernellati | VV | v.vernellati@studenti.unisa.it |
| Antonio Ceruso | AC | a.ceruso9@studenti.unisa.it |
| Benedetta Colella | BC | b.colella@studenti.unisa.it |

Sommario

[**1 - Introduzione 4**](#_heading=h.7f145huf8h1f)

[1.1 - Scopo del sistema 4](#_heading=h.1t3h5sf)

[1.2 - Individuazione Cots 5](#_heading=h.qfs9v5u1hzo6)

[1.3 - Individuazione Design Pattern 6](#_heading=h.28to7x84kq91)

[1.3.1 - Façade 6](#_heading=h.xkd8qvu58c64)

[1.3.2 - Observer 7](#_heading=h.7rq4f66qij59)

[1.4 - Linee guida 8](#_heading=h.rmzd7cbfao90)

[1.5 - Definizioni, acronimi e abbreviazioni 9](#_heading=h.sfg5k3axkwel)

[1.6 - Riferimenti](#_heading=h.m58c58fjca8u) 9

# 

# 1 - Introduzione

## 1.1 - Scopo del sistema

Bridge è una piattaforma digitale concepita per promuovere l’accoglienza e l'integrazione di rifugiati e migranti in Italia, facilitando l'incontro con risorse e opportunità messe a disposizione dalla comunità locale.

Mira a migliorare l'integrazione sociale ed economica dei rifugiati attraverso la gestione e l'assegnazione ottimizzata di alloggi e corsi di formazione. I rifugiati e i volontari potranno interagire direttamente tramite una chat dal vivo, inoltre il rifugiato avrà a disposizione un chatbot multilingue per ricevere supporto immediato e informazioni utili.

Bridge si propone così come un punto di riferimento per la creazione di connessioni reali e solide tra chi arriva e chi accoglie, favorendo l'inclusione sociale e l'autosufficienza dei rifugiati.

## 1.2 - Individuazione Cots

**Java Spring Boot**: Framework avanzato basato su Spring, progettato per semplificare la configurazione e lo sviluppo di applicazioni Java scalabili, con supporto integrato per API REST, microservizi e dipendenze.

**Java Spring**: Framework robusto per lo sviluppo di applicazioni Java, che facilita la creazione di applicazioni scalabili e manutenibili attraverso la sua architettura modulare e le numerose funzionalità integrate.

**React**: Libreria JavaScript per la costruzione di interfacce utente interattive e reattive, che permette di creare componenti riutilizzabili e migliorare l'esperienza utente complessiva.

**MongoDB Atlas**: Database NoSQL orientato ai documenti che utilizza il formato BSON (Binary JSON) per la memorizzazione dei dati, offrendo una struttura flessibile e supporta operazioni di read e write ad alta velocità.

**Hibernate**: Framework ORM per la gestione della persistenza dei dati.

**Spring Security**: Modulo per autenticazione e autorizzazione, utilizzato per proteggere applicazioni web e API.

**Socket.IO**: Libreria per implementare comunicazione in tempo reale, essenziale per la gestione della live chat.

**OpenAI API**: Servizio utilizzato per alimentare il chatbot multilingue, consentendo interazioni in linguaggio naturale. L'API viene integrata tramite Spring Boot per orchestrare richieste e risposte, offrendo un supporto intelligente e immediato agli utenti.

**Mail API (es. JavaMail)**: Per inviare notifiche email agli utenti, utilizzata per eventi come interesse per un alloggio o notifiche relative ad accettazione per un alloggio.

**Checkstyle**: Strumento per mantenere la qualità e l'aderenza agli standard del codice, essenziale per il controllo del codice Java e il rispetto delle convenzioni.

## 1.3 - Individuazione Design Pattern

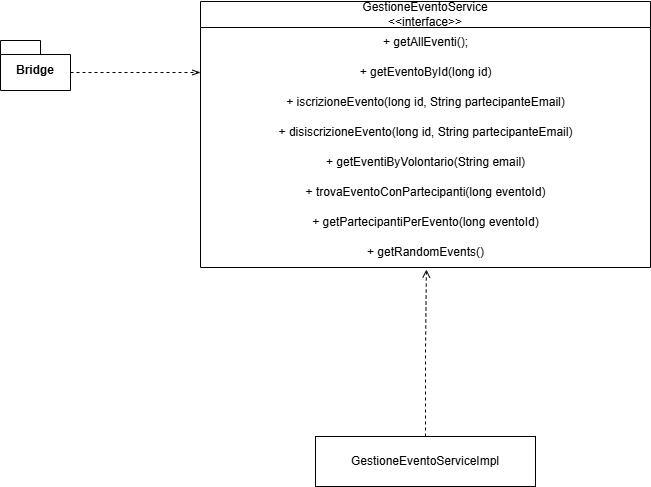
#### **1.3.1 - Façade**

Il design pattern Facade semplifica l'interazione tra un sottosistema e i suoi utilizzatori, riducendo la complessità e la dipendenza reciproca.

Fornisce un'interfaccia unificata che nasconde i dettagli interni del sottosistema, favorendo un basso accoppiamento.

Questo approccio consente di apportare modifiche al sottosistema senza impattare i componenti che lo utilizzano, migliorando la manutenibilità e l'isolamento del codice.

Nel sistema Bridge il design pattern Facade viene implementato per ogni sottosistema tramite delle interfacce service.



#### **1.3.2 - Observer**

L’Observer è un Design Pattern che consente di definire un meccanismo di sottoscrizione per notificare più oggetti circa l’avvenimento di eventi correlati ad uno specifico oggetto che viene “osservato”.

Nel sistema Bridge, tale design pattern sarà responsabile di notificare tramite delle email gli utenti quando si verificano determinati eventi.

Viene implementato attraverso la gestione degli eventi di Hibernate, poiché l'utilizzo delle tradizionali interfacce Subject e Observer avrebbe impedito di sfruttare appieno il principio di Dependency Injection offerto da Java Spring Boot.

### 

## 1.4 - Linee guida

Le linee guida per la scrittura del codice si attengono alla seguente convenzione, di cui sono riportati i dettagli al link che segue:

● **Sun Checkstyle:** <https://checkstyle.sourceforge.io/sun_style.html>

## 

## 1.5 - Definizioni, acronimi e abbreviazioni

● **COTS**: Commercial Off The Shelf

● **DAO**: Data Access Object

● **UML:** Unified Modeling Language

● **Checkstyle:** linee guida per la scrittura del codice

● **Design Pattern**

● **Java Spring Boot MVC:** framework Java

● **ReactJS**

● **API**: Application Programming Interface

## 1.6 - Riferimenti

●  [Statement of Work](https://drive.google.com/file/d/1lI7-PPEv1z373S7F6sfjZjYJFmtTBTfH/view?usp=drive_link)

●  [Matrice di tracciabilità](https://docs.google.com/spreadsheets/d/15nmHqjWaqj7P9LyKv4gSCzlvh9GUT5eg/edit?usp=drive_link&ouid=112179462567237080194&rtpof=true&sd=true)

●  [Requirements Analysis Document](https://docs.google.com/document/d/13qPqBlr6AQ5HTP-HkMtEVIdEKVk3_34R/edit?usp=drive_link&ouid=112179462567237080194&rtpof=true&sd=true)

● [System Design Document](https://docs.google.com/document/d/1KZoTyjVobbQkOVd_sM-IhN-aVyvmAe7Q/edit?usp=drive_link&ouid=115953621529974241157&rtpof=true&sd=true)