Università degli Studi di Salerno
Corso di Ingegneria del Software

Hotel Campus
System Design Document
Versione 1.0

Data: 23/11/2024

Coordinatore del progetto:

Nome	Matricola

Partecipanti:

Nome	Matricola
Luca Del Bue	0512116173
Salvatore Di Martino	0512116932

Scritto da:	Team members

Revision History

Data	Versione	Descrizione	Autore
23/11/2024	1.0	Prima stesura	Team Members

Indice

1. Introduzione

1.1. Scopo del Sistema

Lo scopo del sistema è permettere la gestione dell'attività di una struttura alberghiera. Si intende gestire le camere, i servizi offerti e le prenotazioni effettuate dai clienti. Queste operazioni verranno eseguite rispettivamente dal gestore delle camere e servizi, ossia il direttore, e dal gestore delle prenotazioni. Inoltre, i clienti avranno la possibilità di registrarsi, accedere, finalizzare una prenotazione e visualizzarle.

1.2. Obiettivi di progettazione

1.2.1. Criteri di usabilità

Il sistema deve fornire un meccanismo per evitare l'inserimento di input non validi, come nel caso dell'inserimento delle date al momento della prenotazione e del numero di persone che usufruiscono dei servizi. Questa operazione è svolta mediante l'invio di messaggi di errore da parte del sistema all'utente, comunicando quali dati di input non sono validi, e imponendo dei vincoli sui valori da inserire garantendo consistenza e correttezza.

L'obiettivo del sistema è quindi quello di fornire un'interfaccia utente semplice ed intuitiva attraverso l'implementazione di un design responsive, in modo che la web application possa essere utilizzata su qualsiasi dispositivo, e fornendo in ogni pagina una barra di navigazione garantendo una facile navigabilità tra la varie aree del sistema. Tutto ciò deve essere realizzato rispettando i vincoli citati inizialmente.

1.2.2. Criteri di affidabilità

Il sistema realizzato deve essere robusto, effettuando delle verifiche sui dati inseriti in modo tale da gestire gli input non validi, aggiungendo un ulteriore livello di controllo oltre alla prima verifica effettuata dopo l'inserimento degli input da parte del cliente. Questo consente di gestire situazioni non previste.
Inoltre il sistema deve rispettare dei requisiti di sicurezza, adottando protocolli che garantiscono una connessione sicura e crittografando i dati sensibili degli utenti, come la password dell'account. Infine, la password deve rispettare un vincolo sul numero di caratteri e sul contenuto.

L'obiettivo di progettazione è realizzare un sistema sicuro e affidabile, che effettui i controlli necessari e che protegga le credenziali di accesso degli utenti.

1.3. Riferimenti

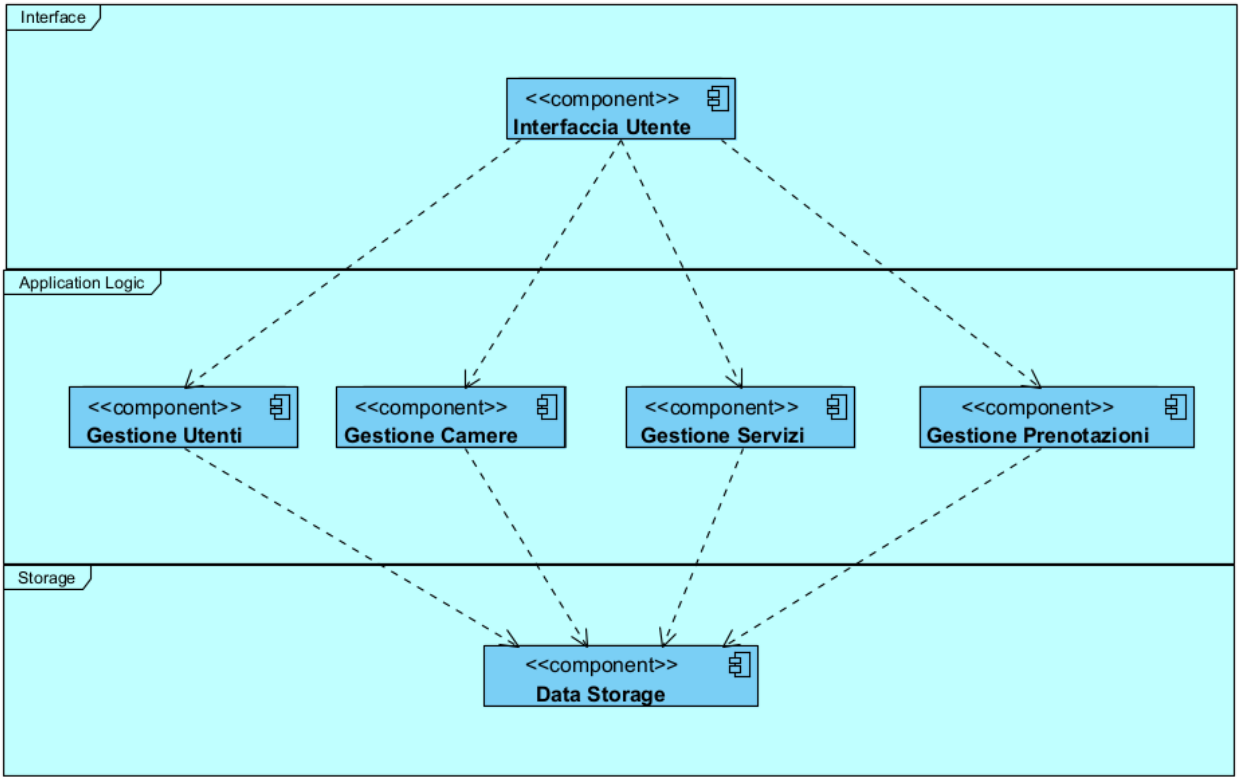
RequirementsAnalysisDocument_HotelCampus

2. Architettura Software Proposta

2.1. Decomposizione in Sottosistemi

Partendo dal modello di analisi prodotto (*RequirementsAnalysisDocument_HotelCampus*), il sistema verrà decomposto seguendo lo stile architetturale a tre livelli: interface, application logic e storage.

Component Diagram



Interface

Interfaccia Utente è il sottosistema che offre le classi e le operazioni necessarie all'interazione con l'utente. Inoltre, coordina la logica applicativa, delegandola ai sottosistemi del layer sottostante, ed effettua le verifiche sui dati in input.

Application Logic

Gestione Utenti fornisce il servizio relativo alla autenticazione, creazione ed eliminazione degli account. Gestione Camere e Gestione Servizi permettono di inserire e rimuovere le camere e i servizi. Gestione Camere si occupa anche di verificare la disponibilità delle camere. Gestione Prenotazione ha il compito di aggiungere ed eliminare le prenotazioni effettuate dai clienti.

Storage

Data Storage è il sottosistema responsabile della memorizzazione degli oggetti persistenti.

In particolare l'architettura individuata è un'architettura chiusa, dove ogni strato può invocare operazioni solo dallo strato immediatamente inferiore. L'obiettivo di progettazione è alta manutenibilità e flessibilità.

2.2. Mapping Hardware/Software

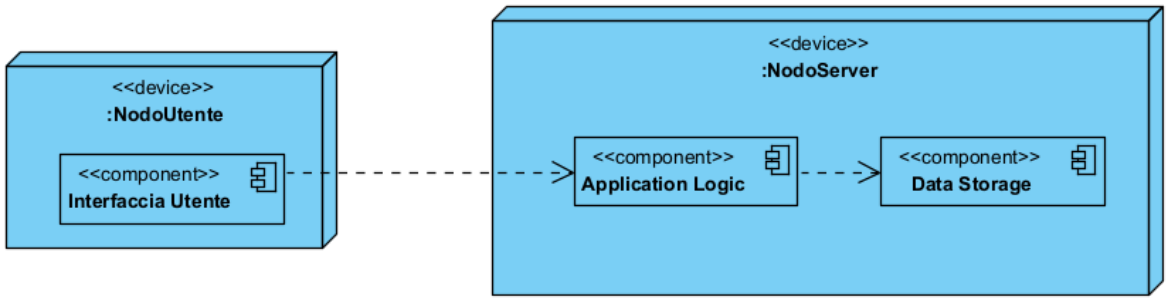
Il sistema Hotel Campus è per sua natura un sistema distribuito, siccome gli utenti operano da macchine diverse da remoto. Distinguiamo due tipi di nodi:

- NodoUtente, che fornisce l'interfaccia utente. Esegue il sottosistema Interfaccia Utente attraverso un web browser.
- NodoServer, che esegue la logica applicativa, i controlli necessari, la gestione della persistenza e i servizi offerti. Esegue tutti gli altri sottosistemi.

Il sistema verrà realizzato usando Java EE in combinazione con il framework Spring Boot. Spring Boot è un framework Java che semplifica lo sviluppo di applicazioni web, riducendo la configurazione manuale. Supporta microservizi, offre server integrati e una gestione semplificata delle dipendenze, permettendo un avvio rapido e un deployment facilitato, grazie alla configurazione automatica e al supporto di molte librerie.

In particolare, per implementare i sottosistemi eseguiti sul NodoServer verranno utilizzate le classi Controller, che elaborano e rispondono alle richieste del browser, e Thymeleaf, un template engine per Java che renderizza pagine dinamiche lato server.

Deployment Diagram



Il component Application Logic racchiude i sottosistemi del relativo layer.

2.3. Gestione dei dati persistenti

Identificazione dei dati persistenti
Il sistema gestisce il seguente insieme di oggetti da memorizzare in maniera persistente:

- Utente
- Camera
- Servizio
- Servizio Prenotato
- Prenotazione

Selezione della strategia di persistenza
Per memorizzare questi oggetti verrà utilizzato un database relazionale, invece che altre strategie, perchè supportano query concorrenti, forniscono meccanismi di transazione per garantire l'integrità dei dati e la possibilità di effettuare backup.
I database relazionali utilizzano SQL per definire e manipolare le tabelle e supportano vincoli come l'integrità referenziale.
Il sistema dovrà memorizzare per le camere e per i servizi le relative immagini. Siccome memorizzare le immagini in formato BLOB in un database è un'operazione onerosa e riduce le prestazioni, queste saranno inserite in una directory nello spazio di indirizzamento del server, e il valore dell'attributo associato all'immagine corrisponderà al path del file.

Tecnologie utilizzate
Per la gestione del database verrà utilizzato MySQL come DBMS. Gli sviluppatori interagiranno con il database attraverso JPA, una specifica di Java per la gestione della persistenza e del mapping relazionale-oggetti, utilizzando oggetti Java invece di scrivere direttamente query SQL.

2.4. Controllo degli accessi e sicurezza

Hotel Campus è un sistema multi-utente, quindi i diversi attori hanno permessi e responsabilità differenti definite dall'insieme di operazioni che possono compiere.

Matrice di controllo degli accessi
Per documentare in modo sintetico i diritti di accesso, viene creata una matrice di controllo degli accessi che mostra le operazioni consentite sugli oggetti entità per ciascun attore.

Attore \ Oggetto	Registro Utenti	Cliente	CatalogoCamere	ElencoServizi	Utente
Utente Ospite	creaUtente		getCamereDisponibili getCamere	getServizi	
Utente	autentica		getCamereDisponibili getCamere	getServizi	
Cliente		getPrenotazioni creaPrenotazione	verificaDisponibilità		
Direttore	Elimina getUtente getUtenti		creaCamera rimuoviCamera	creaServizio rimuoviServizio	setRuolo
Gestore Prenotazioni	getClienti	eliminaPrenotazione cercaPrenotazioni getPrenotazioni			

Per implementare il controllo degli accessi all'interno del sistema verranno utilizzate le liste di controllo degli accessi, dove si associa ad ogni classe le operazioni che gli attori possono compiere.

L'autenticazione sarà effettuata attraverso l'utilizzo di credenziali, ossia email e password, scelte dall'utente in fase di registrazione. Ad ogni utente, una volta autenticato, sarà associata una sessione per eseguire le varie funzionalità.

2.5. Controllo Software

Osservando il dynamic model prodotto in fase di analisi dei requisiti, viene scelto un tipo di controllo centralizzato, siccome i diagrammi di sequenza individuati sono di tipo fork, ossia il controllo è delegato ad un unico oggetto centrale.
In particolare verrà implementato un flusso di controllo event-driven, basato sul paradigma MVC (Model – View – Controller).

Funzionamento generale
Il Web Server attende richieste dal Web Browser. Alla ricezione di una richiesta, il WebServer la elabora e la inoltra al controller, implementando così un flusso di controllo basato su eventi. Il WebServer alloca un nuovo thread per ogni richiesta, consentendo la gestione parallela delle richieste.

Strategie per la gestione della concorrenza
Gli oggetti boundary che verranno implementati non devono definire campi, ma solo contenere dati temporanei relativi alla richiesta corrente in variabili locali. Questo evita problemi di concorrenza poiché gli oggetti boundary sono condivisi tra thread.
Gli oggetti di controllo non verranno condivisi tra thread. Ogni sessione dovrà avere al massimo un oggetto di controllo, impedendo richieste concorrenti che coinvolgono lo stesso oggetto all'interno della stessa sessione.
Gli oggetti entità permetteranno l'accesso e la modifica dello stato solo attraverso metodi dedicati.

Poiché il sistema verrà implementato in un contesto enterprise, le transazioni e la concorrenza verranno gestite automaticamente dal container.

2.6. Condizioni di confine

Le condizioni di confine riguardano l'inizializzazione, la terminazione e i fallimenti del sistema. Per identificare i casi d'uso riguardanti queste attività, viene introdotta la figura dell'amministratore di sistema, sottoforma di attore.

Il sistema non richiede l'utilizzo di software aggiuntivo per la configurazione da parte dell'amministratore. Essendo una web application, l'inizializzazione e la terminazione vengono gestite direttamente tramite l'interfaccia del web container.

2.7.Servizi dei Sottosistemi

Consideriamo i sottosistemi Gestione Utenti, Gestione Camere, Gestione Servizi e Gestione Prenotazioni.

Gestione Utenti

- Definiamo il servizio *Autenticazione* per la verifica delle credenziali di accesso e per la registrazione dell'utente.
- Definiamo il servizio *Ruolo ed Eliminazione* per la selezione del ruolo e la rimozione dell'account.

Gestione Camere

- Definiamo il servizio *Camera* per creare, eliminare e ottenere informazioni sulle camere della struttura.
- Definiamo il servizio *Disponibilità* per la verifica delle camere disponibili.

Gestione Servizi

- Definiamo il servizio *Servizio* per creare, eliminare e ottenere informazioni sui servizi offerti dalla struttura.

Gestione Prenotazioni

- Definiamo il servizio *Prenotazione* per creare, eliminare e ottenere informazioni sulle prenotazioni effettuate dai clienti.