Università degli Studi di Salerno
Corso di Ingegneria del Software

CIRECinema
System Design Document
Versione 2.0

Nome	Matricola
Bufalino Chiara	0512102894
Massaro Edilio	0512103218
Squeglia Raffaele	0512103122
Tiberio Ildo	0512103134

Data	Versione	Descrizione	Autore

Sommario

1. Introduzione

1.1 Scopo del sistema

Lo scopo è quello di realizzare un sistema in grado di fornire un servizio di prenotazione online per il Cinema Multisala CIREcinema, permettendo all'utente, nel caso sia registrato, di poter prenotare posto e orario della proiezione, a sua discrezione.

1.2. Obiettivi di design

La nostra applicazione deve poter essere efficiente ed intuitiva. L'efficienza sarà garantita ove possibile attraverso tempi rapidi di risposta ai vari input possibili garantendo anche politiche di tolleranza all'errore e feedback verso l'utente. L'interfaccia dovrà risultare agevole e navigabile.

1.2.1 Criteri di Performance

Tempo di risposta	L'applicazione dovrà garantire una risposta rapida verso l'utente. Naturalmente sarà necessaria una connessione al server, quindi una velocità di 1s.
Throughput	Il sistema sarà in grado di gestire un numero utenti in una banda limitata dipendente dalle risorse offerte da Altervista, ove sarà hostata l'applicazione.
Memoria	Il quantitativo di memoria usata potrebbe essere abbastanza considerevole. Calcoliamo un numero finito di amministratori, un numero variabile di utenti ed il resoconto (memorizzato) dei loro acquisti.

1.2.2 Criteri di Affidabilità

Robustezza	L'applicazione gestirà eventuali input errati senza compromettere il funzionamento dell'intero sistemo tramite varie politiche di controllo e feedback verso l'utente.
Disponibilità	La disponibilità sarà di 24 ore su 24.
Tolleranza all'errore	Cercheremo di rendere le componenti del nostro sistema il più disaccoppiate possibile, per evitare

	che l'errore di una componente comprometta il resto del sistema.
Sicurezza	L'accesso al sistema è garantito tramite un sistema di login che categorizza gli utenti.

1.2.3 Criteri di Manutenzione

Estendibilità	Cercheremo di garantire un codice pulito così dal poter risultare rileggibile in futuro, nel caso dell'aggiunta di nuove funzionalità.
Modificabilità	Le modifiche potranno essere apportate al sistema, sempre garantito da un codice ben strutturato e pulito.
Leggibilità	Tramite una buona indentazione del codice ed eventuali commenti garantiamo la leggibilità del codice.
Tracciabilità dei requisiti	Grazie alla tracciabilità dei requisiti, vi sarà la possibilità di effettuare modifiche necessarie al corretto funzionamento del sistema valutandone i costi e i rischi che porteranno.

1.2.4 Criteri per l'Utente finale

Usabilità	CeriCinema garantirà un'ottima navigabilità fra le sue pagine e le sue funzioni, accompagnando l'utente nella scelta della proiezione da prenotare al massimo delle possibilità. Tutto ciò sfruttando un'interfaccia intuitiva basandoci sui DesignPatter più famosi e prendendone spunto.

1.3 Definizioni, Acronimi ed Abbreviazioni

- CeriCinema (Chiara, Edilio, Raffaele, Ildo): Nome dell'applicazione web.
- RAD: Requirements Analysis Document.
- Altervista: Servizio online di Hosting.
- MySQL: Applicazione per la gestione del DBMS.
- DBMS: Database Management System.
- Design Patter: regole di successo da seguire per poter raggiungere un determinato obiettivo di design\usabilità.

1.4 Riferimenti

Documento Requirement Analysis Document del progetto CeriCinema.

1.5 Panoramica

Di seguito sono riportate le attività del system design che costituiscono le fondamenta per l'architettura software del sistema.

- Decomposizione del sistema in sottosistemi caratterizzati dai servizi offerti verso gli altri sottosistemi. Tale insieme di servizi è denominata come Interfaccia.
- Mapping Hardware/Software, ovvero la scelta della configurazione hardware del sistema, la comunicazione tra i nodi e l'incapsulamento dei servizi del sottosistema.
- Gestione della persistenza dei dati.
- Politiche di accesso e Sicurezza.
- Controllo del software globale, che permette di guidarci su quali operazione eseguire ed in quale ordine, per garantire il corretto flusso di controllo del sistema.
- Condizioni Boundary, che includono oltre l'avvio e lo shutdown anche la gestione dei fallmenti dovuti all'invecchiamento del sistema, interruzione di corrente o anche errori di progettazione.

2. Architettura del software Proposto

3.1 Panoramica

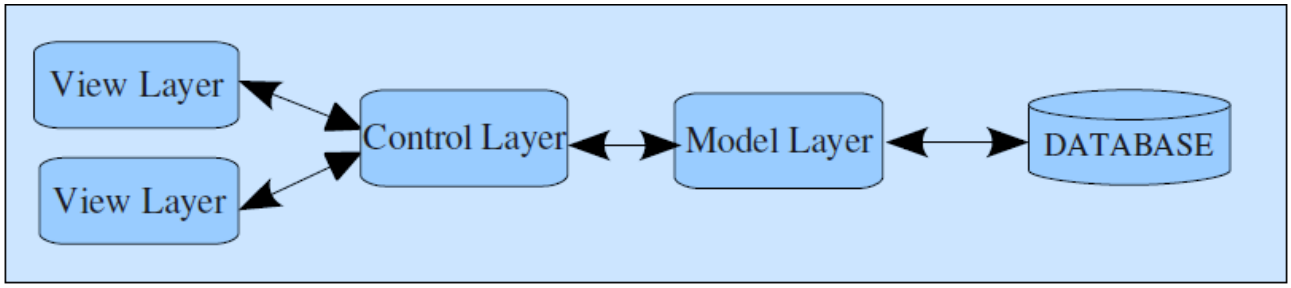
L'architettura del nostro sistema si presenterà suddivisa in tre diversi livelli seguendo lo schema dell'architettura MVC (Model/View/Control). Tale scelta è dovuta alla necessità di un'architettura che permetta una separazione netta tra i componenti software e i componenti che gestiscono i dati stessi.
Gli utenti di Coffee potranno accedere al sistema attraverso un client sul quale risiederanno l'interfaccia utente e la logica applicativa, mentre l'archivio dei dati e la sua gestione risiederanno su un unico server.

I diversi livelli previsti dall'architettura MVC sono:

- **View Layer**: il quale racchiuderà tutti gli oggetti boundary che costituiranno l'interfaccia grafica e attraverso i quali l'utente interagirà con il sistema.

- **Control Layer**: il quale conterrà tutti gli oggetti control che fungeranno da intermediari con le entità del sistema.

- **Model Layer**: nel quale saranno presenti tutti gli oggetti entity.



L'applicazione sarà strutturata su 3 livelli (three-tier):

- Interface layer.
- Application Logic layer.
- Data layer.

[FOTO DEI LAYER]

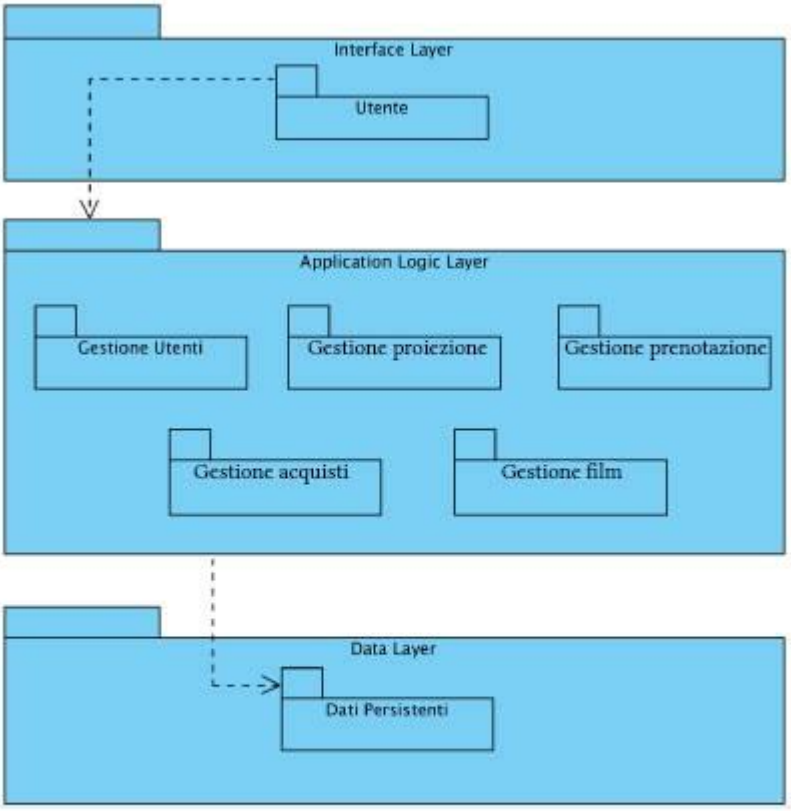
Interface Layer: corrisponde all'interfaccia che funge da tramite fra l'utente ed il sistema.

Application Logic Layer: si occupa di elaborare ed inviare dati al client. Suo compito è di interrogare il database, tramite lo Storage Layer, per accedere ai dati persistenti.

Data layer: ha il compito di memorizzare i dati sensibili del sistema, utilizzando un DBMS. Esso riceve le varie richieste dall'Application Logic layer inoltrandole al DBMS.

3.2 Decomposizione del sistema

Di seguito riportiamo un diagramma generale e la descrizione di ogni modulo

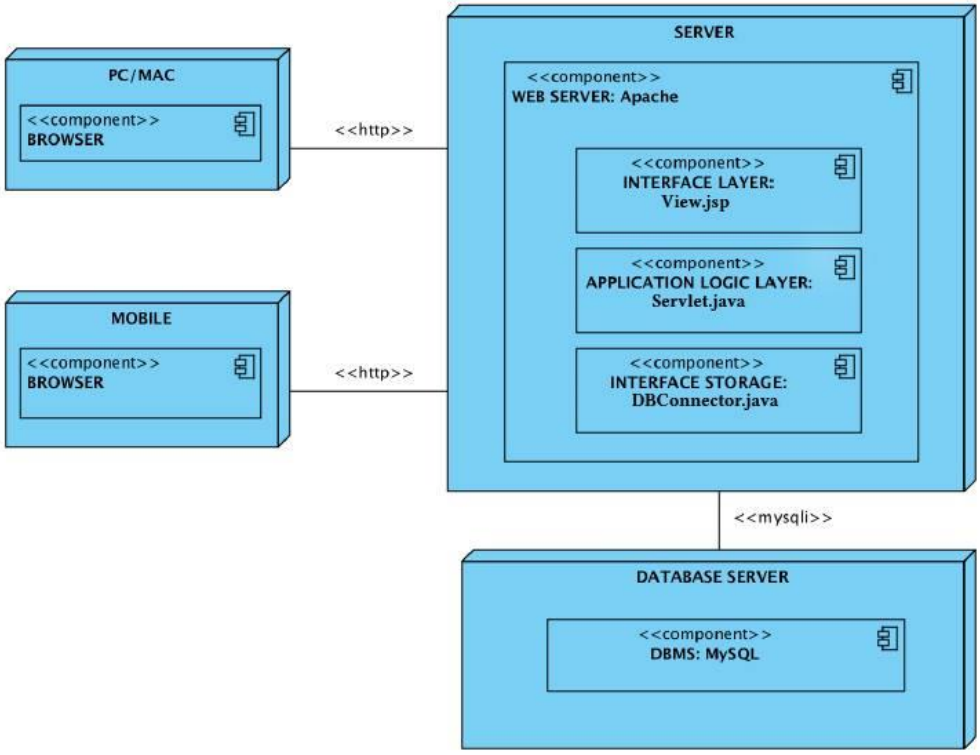


Interface layer	
Utente - Utente Loggato	Modulo che si occupa di gestire le funzionalità dell'utente loggato, ovvero la possibilità di gestire il carrello personale e di prenotare biglietti per le proiezioni.
Utente - Utente Non Registrato/Non loggato	Modulo che si occupa di gestire le funzionalità base di un utente che o non è registrato o che non ha effettuato il login, come mostrare le proiezioni presenti nel database non dando però la possibilità di comprarle.
Utente – Admin	Modulo che si occupa di gestire le funzionalità dell'amministratore, come la gestione delle proiezioni, delle sale, dei prezzi.

Application Logic layer	
Gestione Utenti	Modulo che permette la registrazione di un nuovo utente al sistema, gestisce l'accesso, garantendo l'adeguata assegnazione dei privilegi dovuti e le relative funzionalità.
Gestione Proiezione	Modulo che permette di visualizzare le proiezioni presenti nel database insieme agli ulteriori dati per quest'ultima.
Gestione Prenotazione	Modulo che permette di visualizzare le prenotazioni ed effettuarle.
Gestione Acquisti	Modulo che permette all'utente di visualizzare lo storico dei suoi acquisti.
Gestione Film	Modulo che permette la visualizzazione dei film.

Data layer	
Dati persistenti	Modulo che si occupa di memorizzare dati in memoria, in modo da poter essere prelevati e modificati in modo corretto.

3.3 Hardware/Software Mapping



Web Server

Il server utilizzato sarà Tomcat.

Interface layer

L'utente utilizza il sistema mediante un Browser installato all'interno del suo calcolatore (ad es. Opera, Firefox, Chrome).

Application Logic Layer

Il sistema, e quindi le funzionalità, sono implementate tramite Servlet in java.

Storage Layer

Rappresenta il collegamento con il server da parte del sistema e si occupa di tutte le richieste di accesso e modifiche sui dati permanenti presenti nel database.

Database Server

Il DBMS usato è MySQL il quale presenta molte API che permettono l'interazione tra il sistema ed il database.

3.4 Gestione dei dati persistenti

Il sistema CeriCinema presenta i seguenti oggetti da memorizzare in modo persistente:

- Utente Registrato
- Proiezione
- Prenotazione
- Film

Per la memorizzazione dei dati è stato scelto un database relazione in modo tale da memorizz are i dati in tabelle che seguono uno schema definito. L'organizzazione dei dati attraverso questo modello permette una semplice realizzazione di query complesse.

3.5 Controllo degli accessi e sicurezza

3.5.1 Controllo degli accessi

Il sistema CeriCinema prevede, in generale, le tre figure di utenti: Amministratore, Utente Non Registrato ed Utente Registrato. Ogni utente, ad eccezione dell'utente non registrato, accede tramite procedura di login, a sottosistemi diversi, i quali permettono di usufruire delle funzionalità specifiche ad essi associate.

3.5.2 Sicurezza

Il sistema è protetto dagli accessi non autorizzati attraverso un sistema di login/password. Ogni utente registrato al sistema è proprio di una login univoca ed una password di minimo 6 caratteri e massimo 10. Un utente per poter avere accesso al sistema deve compilare due campi: login e password. Il sistema qundi analizza i campi login e password e controlla l'esistenza di una tupla con gli specifici dati inviati all'interno del database. Nel caso in cui la tupla ricercata non fosse presente nel database viene generato un messaggio di errore e riproposto l'inserimento dei dati.

Il modello login/password non fornisce sicuramente un eleveto livello di sicurezza, ma considerando che gli accessi possono essere effettuati solo sulla rete locale, il livello di sicurezza viene considerato sufficiente per il conteso dell'applicazione.

3.6 Gestione del controllo globale

Il flusso di controllo di questo sistema software non prevede alcuna seq uenza prestabilita, fatta eccezione per l'acquisto di una prenotazione o la registrazione. Sarà l'interazione con l'utente a determinare il flusso

di controllo, le classi Java funzioneranno da ricevitori di eventi per rispondere alle attività dei client. Saranno le servlet di sistema che elaboreranno le richieste prendendo informazioni dal database fino ad inviarle all'utente che grazie ad un'interfaccia grafica saranno mostrate.