PROGETTO B:
SCHOOLLIB

AUTORI:
Lorenzo Gavazzeni
Gerald Shyti

INDICE

1 - Scelte Algoritmiche

SERVER

- Si è cercato il più possibile di utilizzare il pattern Model View Controller, per separare la logica del codice dalla visualizzazione dell'interfaccia.

- Si è utilizzato l'RMI (Remote Method Invocation) per la comunicazione Client-Server che, rispetto al classico utilizzo dei socket, è uno strumento più ad alto livello e di configurazione più immediata, risulta più semplice da utilizzare ed è più facile scrivere codice che rispetti criteri di qualità.
Inoltre garantisce automaticamente il multiclient

- I metodi per la gestione del database sono stati divisi in oggetti di diverso tipo e funzionalità, questi oggetti vengono generati solo una volta alla prima esecuzione del server e vengono richiamati solo mediante l'oggetto SQLCore che incapsula i metodi corrispondenti agli oggetti di cui si necessita

- I metodi lato server aventi come compito la comunicazione con la base di dati sono tutti "synchronized", questo garantisce un accesso contemporaneo e sicuro al database.

- Il codice di EmailSender2 è il risultato di una modifica del codice dato da EmailSender.

- Alla prima esecuzione del server viene invocato un metodo per il controllo dell'esistenza delle tabelle e per la loro generazione.
Le tabelle vengono generate in modo parametrico, i metodi infatti sfruttano l'oggetto SQLSupporter che mette a disposizioni varaibili costanti e metodi utili in tutti i processi di creazione/inserimento/cancellazione/aggiornamento.

CLIENTS

- All’inizio dell’esecuzione si accede all’interfaccia di Login.

- Viene creato un oggetto Reader/Librarian che stabilisce immediatamente una connessione con il server e salva un oggetto di tipo ServerInterface che gli permetterà di accedere ai metodi esposti del server.

- Per le diverse applicazioni (Reader/Librarian) si è scelto di utilizzare due differenti progetti in quanto l’interfaccia Server utilizzata è diversa e per questioni di sicurezza è importante non esporre metodi di un tipo di Client ad un altro tipo di Client

- Viene introdotto un Package Checker che permettono di effettuare controlli su quelli che sono i paramentri inseriti dagli utenti (di qualasi tipo) per il controllo di validità solamente sulla sintassi di questi

- Vengono poi introdotti metodi mediante interfaccia del server per effettuare verifiche sui paramentri inseriti una volta che questi vengono dichiarati come validi dal Checker

- Viene introdotto un Package Common lato Server e lato Client contente gli oggetti e le loro strutture per effettuare le corrette procedure di Setting e Getting dei dati contenuti negli oggetti.
Le Classi all’interno di questo Package estendono tutte Serializable, questo perchè il Client deve essere in grado di inviare un oggetto nella rete e RMI ha come vincolo che questo sia Serializable, inoltre il Server deve poter contare sul fatto che questi oggetti inviati ed entranti debbano poter essere inseriti correttamente all’interno del DB. Ognuno di questi oggetti ha infatti un serialVersionUID settato su 1L che deve corrispondere sia lato client che lato server.

2 - Strumenti

Sistemi Operativi su cui si è effettuato ideazione, sviluppo e test sono:
Windows 10 e Mac OSX HighSierra

La versione utilizzata per la realizzazione dell'interfaccia grafica e del suo design è NetBeans IDE 8.2

La versione utilizzata per le modifiche, otimizzazione, refactoring, bugfixing e per la release è IntelliJ IDEA 2017.1

Per il database è stato utilizzato PgAdmin 4 PostgresSQL.
Tutte le query per il corretto funzionamento di questo programma sono state sviluppate e testate su questo software, l'utilizzo di un altro software non garantisce il corretto funzionamento del software

Per coordinare il lavoro, gestire le diverse versioni e per tenere traccia di ogni modifica effettuata al progetto durante lo sviluppo è stato utilizzato Git.
Per verificare forniamo il link pubblico al progetto:
https://github.com/garynk/SCHOOLIB-3

Per la compilazione e per la generazione dei .jar è stato utilizzato Apache Ant v1.10.1
Viene utilizzato un file "schoolibbuildfile.xml" uguale per tutte le Applicazioni del progetto.
Vengono differenziati i file .properties a seconda del sistema operativo su cui si sta andando ad effettuare la build o la generazione del .jar file.

3 - Documentazione

Nella cartella Documents del progetto sono presenti:

- UML:

ClassDiagram:
- serSchoolLib
- appLibrarian
- appReader

Usecase Diagram

- Versioning:

Un file di testo contenente lo storico delle versioni fino ad una release stabile chiamato “LogBook”

- Guida per l’istallazione

Un file di testo con le istruzioni per installare correttamente i vari programmi

- Database

In formato immagine è presente un E/Rdiagram per la struttura del database

- Javadoc

In ogni cartella di progetto è contenuto il Javadoc del corrispondente

4 - Presentazione

Generale - AppReader / AppLibrarian

- Schermata di LogIn

Login Utente: E' possibile inserire i dati di Username e Password e successivamente effettuare il LogIn prendo il tasti "Entra"

In caso di utente non ancora attivato verrà richiesto il codice ottenuto per email in fase di registrazione

Registrazione Nuovo Utente: Premendo il tasto Registrati si accede alla form di registrazione per un nuovo utente.
Sulla email inserita si riceverà un codice per confermare la registrazione

Password Dimenticata: Premendo la scritta "Password dimenticata ?" e inserendo l'ID utente di cui si vuole riottenere la password, si riceverà una mail con la nuova password sulla mail corrispondente a quel dato ID

AppReader

- Schermata Principale

Ci troviamo di fronte alla schermata principale.
La form è composta da un campo di Dati Utente e da una tabella centrale con all'interno l'intero catagolo dei libri consultabile su pagine di 10 righe per pagina.

Effettuare una prenotazione: Selezionando preventivamente un libro

dalla tabella verrà sbloccato il pulsante “Prenota”.
Selezionando un libro dall’elenco e premendo il dato pulsante si potrà effettuare una prenotazione.

Disdici Prenotazione: Selezionando un libro dall’elenco, se questo è stato prenotato dall’utente loggato, si sbloccherà il pulsante “Disdici”, premendo questo pulsante si potrà disdirre la prenotazione

Refresh: Premendo l’icona con la freccina sarà possibile aggiornare la tabella

Cerca: Inserendo nella apposita casella di testo una parte di un titolo, di un autore o di una categoria e premendo il tasto con la lente sarà possibile ricercare un libro secondo il paramentro inserito.

Visulizzazione Prenotazioni/Prestiti: Premendo i rispettivi tasti in alto si potranno visualizzare le prenotazioni o prestiti attivi su quel dato utente

Modifica: Premendo questo tasto di entra nella Form di modifica delle informazioni e/o cancellazione relative all’utente attualmente loggato.

AppLibrarian

- Schermata Principale

Ci troviamo di fronte alla schermata principale.
La form è composta da un campo di Dati Utente e da una tabella centrale con all’interno l’intero catagolo dei libri consultabile su pagine di 10 righe per pagina.

Aggiungere: Premendo sul pulsante “Aggiungi” sarà possibile accedere alla form di compilazione dei campi relativi all’aggiunta di un nuovo libro.

Eliminare: Premendo sul pulsante “Elimina” e selezionando un libro dalla tabella sovrastante, sarà possibile eliminare il libro selezionato

dal catalogo.

Crea Lettore: Inserendo un ID corretto nel campo di testo e premendo il pulsante “Crea” sarà possibile accedere alla form per crare un nuovo utente Reader.

Cancella Lettore: Inserendo un ID corretto nel campo di testo e premendo il pulsante “Cancella” sarà possibile eliminare un Reader con quel dato ID

Crea Prenotazione: Selezionando un libro dalla tabella e inserendo un ID di un reader esistente e premendo sul tasto “Crea” sarà possibile creare una prenotazione

Visualizza Prenotazione: Inserendo nel campo di testo un ID Reader corretto sarà possibile visualizzare sulla tabella solo le prenotazioni di quel dato ID Reader

Disdici Prenotazione: Selezionando un libro dalla tabella e inserendo un ID di un reader esistente e premendo sul tasto “Elimina” sarà possibile eliminare una prenotazione se questa esiste.

Attiva Prestito: Selezionando un libro dalla tabella e inserendo un ID di un reader per cui esiste una prenotazione associata al libro selezionato, sarà possibile iniziare un nuovo prestito

Disattiva Prestito: Inserendo nel campo di testo un ID Reader corretto sarà possibile disattivare un prestito èrecedentemente attivato

Disdici Prestito: Selezionando un libro dalla tabella e inserendo un ID di un reader esistente e premendo sul tasto “Disdici” sarà possibile disdirre un prestito registrandolo tra i prestiti storici

Modifica: Premendo questo tasto di entra nella Form di modifica delle informazioni e/o cancellazione relative all’utente attualmente loggato.

Statistiche: Premendo il tasto statistiche si accede alla form delle statistiche.

La Form è composta da pulsanti e una tabella

Statistiche-Prestiti: inserendo un ReaderID corretto e premendo sul tasto Prestiti / Storico è possibile vedere rispettivamente i Prestiti / Storico Prestiti di quel dato utente ReaderID

Nella Sezione Statistiche-Generali
E' possibile visualizzare i Prestiti/Prenotazione/Prestiti Storici nella tabella a fianco

Selezionando una delle opzioni tra "Assoluta" "perCategoria" "perInquadramento" e premendo sul pulsante Mostra, è possibile visualizzare l'elenco dei libri ordinati secondo il parametro selezionato

serSchoollib

Ci troviamo di fronte alla schermata principale.

Nella sezione Email si dovrà inserire una mail valida (testato sul dominio studenti.uninsubria.it)

Nella sezione Password si dovrà inserire una password associata alla mail nel dominio corrispondente

Premendo il tasto OK si confermano le impostazioni e il server inizierà la sua attività.

5 - Fasi di Lavoro

FASE 1 – analisi, stesura

Analisi dei Requisiti
Suddivisione del lavoro tra gli elementi del gruppo

FASE 2 – Stesura diagrammi UML

Prima stesura per macropunti dei digrammi UML di Classe per Server e Clients
    Utilizzato: Creatly.app
Prima stesura del digramma UML Usecase
    Utilizzato: Visual Paradigm
Prima stesura per macropunti del digramma E/R
    Utilizzato: draw.io

FASE 3 – Sviluppo Server

Creazione della parte relativa all'RMI
Creazione primi metodi di comunicazione Server/Client

Testing della connessione S/C

Creazione classi SQL per la gestione del database
Stesura e Testing su Postgres delle Query usate nei metodi
Implementazione Query nei metodi per gestione del DB lato server

Implementazione dell'EmailSender
Testing EmailSender

Creazione interfaccia grafice Server (NetBeans)

FASE 4 - Strutturazione Oggetti Generici

Partendo dalla struttura delle Tabelle e dai loro parametri

Creazione oggetti “Common” e “Checker”
Creazione Classi per Utente (Librarian e Reader), Libro
Creazione Classi per Checker(UserChecker, BookChecker)

FASE 5 – Sviluppo AppLibrarian

Creazione interfaccia grafica LogIn (NetBeans)
Creazione interfaccia grafica appLibrarian principale (NetBeans)
Creazione interfaccia grafica appLibrarian-Registrazione (NetBeans)

Creazione metodi Core per stabilire e mantenere la connessione
Creazione metodi Core per interfacciarsi al DB e ottenere risultati dal server

Implementazione definitiva dei componenti grafici con relative funzionalità

Testing delle funzionalità

FASE 6 – Sviluppo AppReader

Creazione interfaccia grafica LogIn (NetBeans)
Creazione interfaccia grafica appReader principale (NetBeans)
Creazione interfaccia grafica appReaeder-Registrazione (NetBeans)

Creazione metodi Core per stabilire e mantenere la connessione
Creazione metodi Core per interfacciarsi al DB e ottenere risultati dal server

Implementazione definitiva dei componenti grafici con relative funzionalità

Testing delle funzionalità

FASE 7 – Releasing

Creazione dei file di build su Apache/Ant
Stesura dei file di build Apache/Ant
Testing dei file di build Apache/Ant

FASE 7 – Ottimizzazione

Testing globale del progetto e debugging
Ottimizzazione interfaccie grafiche e funzionalità
BugFix
Rivisitazione completa del progetto

FASE 8 – Stesura Documentazione

Stesura guida di installazione
Stesura Javadoc per ogni Entità
Stesura documentazione generale