Università degli Studi di Padova

Progetto di Programmazione ad Oggetti

A.A.2022-2023

PlannerVi

Elena Chilese
Matricola: 2008074

1 Abstract

Lo scopo del progetto è la realizzazione di un'applicazione in C++/Qt per la gestione della prenotazione delle aule all'interno del Conservatorio di Vicenza. Viene fornita l'interfaccia in cui gli utenti del conservatorio possono effettuare il login con le proprie credenziali (oppure registrarsi in caso di nuovo utente) e consultare l'occupazione delle varie aule. Le aule in questione possono essere di tre tipi differenti, in base alle esigenze dell'utilizzatore:

1. **Aula Studio**: aula dedicata allo studio individuale o in piccolo consort dello strumento/i in possesso dello studente/i.
   L'aula fornisce leggii e prese per la ricarica di dispositivi elettronici

2. **Aula Strumentale**: aula adatta a lezioni di strumento, quindi con la compresenza di un docente, può essere utilizzata anche come aula studio.
   Si differenzia per la presenza di uno strumento non trasportabile (pianoforte, clavicembalo, fortepiano, organo, ...) oppure perchè dedicata ad uno strumento specifico (classe di liuto, classe di arpa, ...)

3. **Aula Concerto**: aula utilizzata per esercitazioni corali, orchestrali e, nel rispetto del calendario delle Produzioni del Conservatorio, per i concerti e gli eventi pubblici

PlannerVi può essere utilizzato nel concreto in quanto attualmente il Conservatorio di Vicenza è sprovvisto di tale sistema informatico di prenotazione.

Sito web PlannerVi

Attualmente la prenotazione è gestita tramite invio di mail, ma spesso non vengono lette in quanto sono numerose e poco pratiche.

2 Descrizione classi

2.1 Model

aula: classe base astratta che permette la memorizzazione dei dati comuni delle varie aule, come ad esempio il numero dell'aula.

aulaStudio: classe contenente i campi dati specifici di un'aula studio, come il numero di leggii e di prese corrente.

aulaStrumentale: classe contenente i dati specifici di un'aula strumentale, ossia lo strumento specifico presente in quell'aula.

aulaConcerto: classe contenente i dati relativi alla capienza massima di pubblico e alla presenza di un sistema di amplificazione, oltre alle informazioni sugli strumenti non trasportabili lì presenti.

utenti: classe che rapprensenta un utente, quindi contiene nome, cognome, codice fiscale, telefono, ruolo (Docente o Studente), email e password.

prenotazioni: classe rappresentante una prenotazione vera e propria e usata come "collegamento" tra aule e utenti in quanto contiene sia la mail dell'utente che il numero dell'aula da prenotare. Oltre a tali dati contiene la data, l'ora di arrivo e di uscita dall'aula selezionata e la causale dell'occupazione di tale aula (ad esempio: lezione di strumento, studio individuale, ...).

storage: è una classe creata appositamente per la memorizzazione delle aule, delle prenotazioni e degli utenti. Aule e prenotazioni sono memorizzate tramite contenitori, strutture dati appositamente creati tramite nodi, e gli utenti sono invece memorizzati in un vector.

2.2 View

view: classe base per le viste.

login_view: classe per la finestra di login, che permette l'inserimento della mail e della password in caso di utente registrato, altrimenti vi è un link da cui si accede alla finestra di registrazione.

registrazione_view: classe per la finestra di registrazione in cui inserire i propri dati personali, la mail istituzionale (deve finire con "@consvi.it" e la password (deve contenere almeno 6 caratteri, di cui almeno una lettera maiuscola e un numero).

admin_view: classe per la finestra dedicata all'admin, ossia al gestore delle prenotazioni. A tale finestra si accede automaticamente dopo aver effettuato il login con le credenziali dell'admin.

menu_view: classe per la finestra principale di menu che si apre agli utenti dopo aver effettuato il login. In essa sono presenti 3 pulsanti che permettono rispettivamente di: visualizzare il proprio profilo, visualizzare le prenotazioni personali e visualizzare tutte le prenotazioni (serve per consultare l'occupazione delle aule e per trovare le aule dove si svolgono le lezioni di interesse).

profilo_view: classe per la finestra di visualizzazione dei dati personali.

myPren_view: classe per la visualizzazione delle prenotazioni personali, con possibilità di aggiunta e di rimozione delle stesse.

prenotazioni_view: classe per la visualizzazione di tutte le prenotazioni, in ordine di data, di numero dell'aula.

2.3 Controller

Controller: controller base per la gestione di tutti i controller specifici.

LoginController: controller per la login_view, gestisce i segnali ed effettua i controlli relativi al login (esistenza effettiva della mail nel database e successiva verifica della correttezza della password).

RegController: controller per la registrazione_view che controlla la correttezza dei dati inseriti (nello specifico inserisce limiti riguardanti il numero di telefono che contiene interi e controllo password accettabile).

AdminController: controller che gestisce l' admin_view collegando i segnali ai relativi slot.

MenuController: controller che gestisce il collegamento dei segnali emessi dai 3 pulsanti agli slot per l'apertura delle relative view.

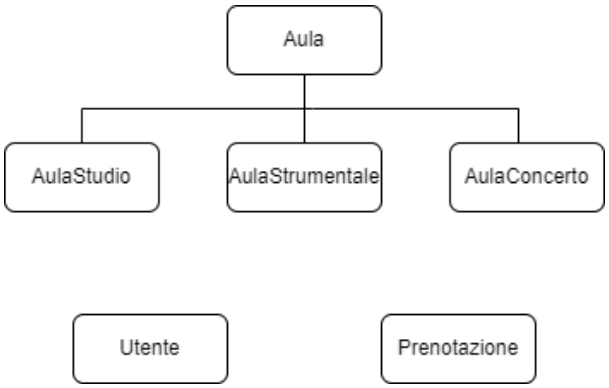
ProfiloController: controller per la profilo_view.

MyPrenController: controller per la myPren_view che gestisce anche la modifica del Model/storage per l'inserimento e la rimozione delle prenotazioni.

PrenController: controller per la prenotazioni_view che gestisce anche la modifica del Model/storage per l'inserimento delle prenotazioni.

salvataggio_dati: controller per la gestione del file.json relativo alla persistenza dei dati.

3 Rappresentazione grafica



4 Descrizione del codice polimorfo

- **aula**: classe astratta implementata nel Model.

- **view**: classe astratta che contiene il metodo per la chiusura della view, il settaggio del titolo e della dimensione della finestra oltre che ai metodi per la visualizzazione di messaggi ed errori.

- **controller**: classe astratta che contiene il distruttore virtuale dei controller e i metodi virtuali getView() e getModel().

- **model**: classe astratta che contiene il distruttore virtuale di model.

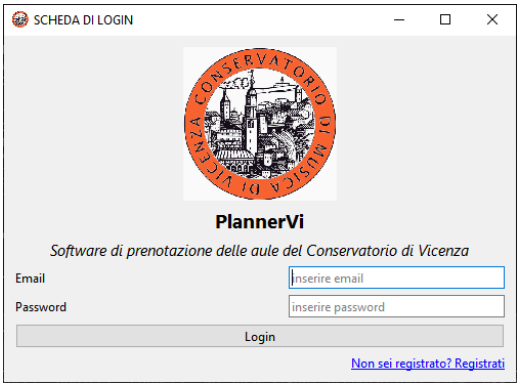
5 Ore richieste

- Analisi preliminare del problema - 2 ore
- Progettazione modello e GUI - 3 ore
- Codifica modello – 10 ore
- Codifica GUI (Model, View, Controller) in Qt - 25 ore
- Debugging – 10 ore
- Testing – 3 ore
- Apprendimento Latex e scrittura relazione – 3 ore

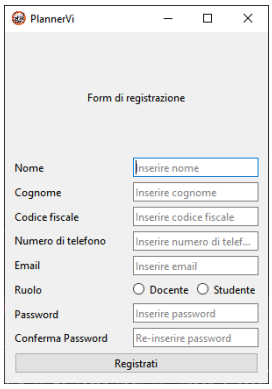
Le ore totali sono 56, ossia 6 in più delle 50 richieste nelle specifiche del progetto, soprattutto a causa dello studio della libreria Qt.

6 Manuale GUI

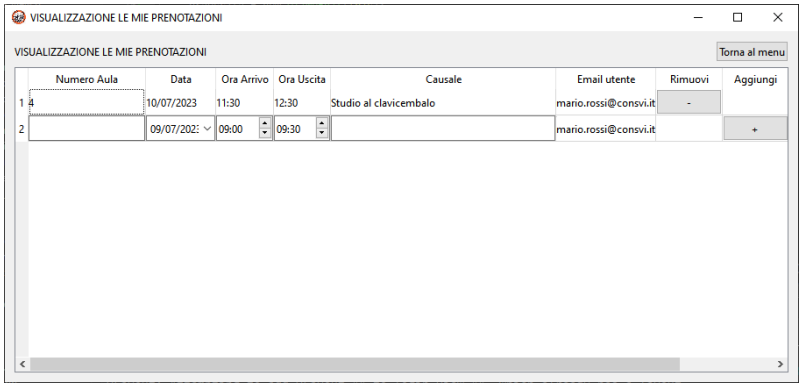
All'avvio dell'applicazione, si deve immediatamente scegliere il file.json che l'applicazione poi caricherà.
Successivamente si visualizza la seguente pagina di login.

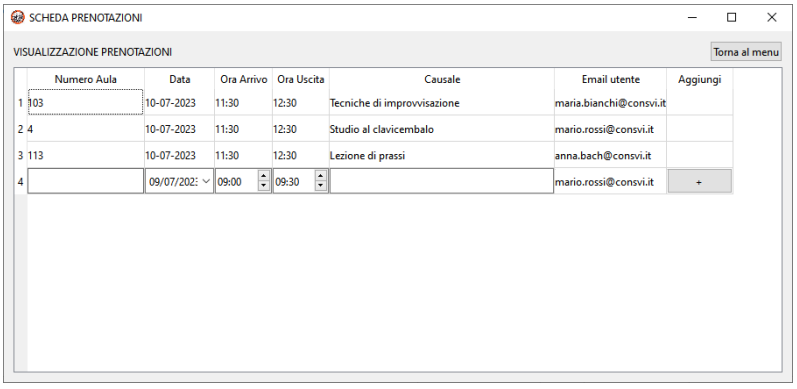


Se non si è registrati vi è un link apposito che porta alla pagina di registrazione del nuovo utente. La registrazione deve avvenire tramite la mail istituzionale fornita al momento dell'iscrizione per gli studenti, al momento della presa di servizio per i docenti.



Dopodichè si aprirà la finestra di menu da cui si può accedere alle finestre di: visualizzazione del profilo utente, visualizzazione delle prenotazioni dell'utente e visualizzazione di tutte le prenotazioni(in ordine di data crescente e aula).


VISUALIZZAZIONE LE MIE PRENOTAZIONI
VISUALIZZAZIONE LE MIE PRENOTAZIONI
Torna al menu
Numero Aula
Data
Ora Arrivo
Ora Uscita
Causale
Email utente
Rimuovi
Aggiungi
1 4 10/07/2023 11:30 12:30 Studio al clavicembalo mario.rossi@consvi.it -
2 09/07/202: 09:00 09:30 mario.rossi@consvi.it +


SCHEDA PRENOTAZIONI
VISUALIZZAZIONE PRENOTAZIONI
Torna al menu
Numero Aula
Data
Ora Arrivo
Ora Uscita
Causale
Email utente
Aggiungi
1 103 10-07-2023 11:30 12:30 Tecniche di improvvisazione maria.bianchi@consvi.it
2 4 10-07-2023 11:30 12:30 Studio al clavicembalo mario.rossi@consvi.it
3 113 10-07-2023 11:30 12:30 Lezione di prassi anna.bach@consvi.it
4 09/07/202: 09:00 09:30 mario.rossi@consvi.it +

7 Note

- **Ambiente di sviluppo:**
  - **Sistema operativo**: Windows 10 Home
  - **Compilatore**: GCC 9.2.0
  - **Versione libreria Qt**: 6.5.1
- **Materiale consegnato:**
  - file *.h* e *.cpp* per la compilazione del progetto;
  - file *PlannerVi.pro* utilizzato per la creazione del makefile;
  - file *resources.qrc* che serve per accedere alla cartella /Immagini;
  - *file.json* necessario per la persistenza dei dati.
- **Credenziali per il test:**
  - **Admin**
    - *email*: admin@consvi.it
    - *password*: PlannerVi2023
  - **Studente**
    - *email*: mario.rossi@consvi.it
    - *password*: Mario2002
  - **Docente**
    - *email*: anna.bach@consvi.it
    - *password*: Anna1981
- **Risorse utilizzate:**
  È stato utilizzato il logo ufficiale del Conservatorio A. Pedrollo di Vicenza.