UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Progettazione e Sviluppo di uno Strumento per la Gestione Scolastica

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: García Vicente, Guillermo

Tutor: Castiglione, Arcangelo

Tutor: Sánchez Díaz, Juan

Curso 2019 - 2020

Ringraziamenti

Alla mia famiglia, papà, mamma e Mariana
per avermi sostenuto nei momenti più difficili e aver sempre fiducia in me.

Ai miei amici,
che insieme alla mia famiglia sono le persone più importanti della mia vita.

Al prof. Castiglione e Juan Sanchez, miei tutori,
per il suo aiuto nello sviluppo di questo progetto.

Grazie mille.

Resum

El present treball de fi de grau tracta sobre el desenvolupament d'una aplicació d'escriptori destinada a la gestió escolar, de manera que comptarà amb dos rols diferents: professor i alumne. Per a la seva realització farem ús de 3 fases principals, en la primera d'elles analitzarem els requisits de l'aplicació, en la segona ens centrarem en el disseny centrat en l'usuari. Finalment, en la tercera fase, realitzarem la implementació de l'aplicació fent ús de JavaFX i MySQL.

Paraules clau: gestió escolar, professor, alumne, tasques, exàmens, JavaFX, MySQL, fxml.

Sintesi

El presente trabajo de fin de grado trata sobre el desarrollo de una aplicación de escritorio destinada a la gestión escolar, de modo que contará con dos roles diferentes: professore y Studente. Para su realización haremos uso de 3 fases principales, en la primera de ellas analizaremos los requisitos de la aplicación, en la segunda nos centraremos en el diseño centrado en el usuario. Por último, en la tercera fase, realizaremos la implementación de la aplicación haciendo uso de JavaFX y MySQL.

Palabras clave: gestión escolar, professore, Studente, tareas, exámenes, JavaFX, MySQL, fxml.

Abstract

The present work of end of degree deals with the development of a desktop application for school management, so it will have two different roles: teacher and student. For its realization we will make use of 3 main phases, in the first one we will analyze the application requirements, in the second we will focus on the user-centered design. Finally, in the third phase, we will carry out the implementation of the application using JavaFX and MySQL.

Keywords: school management, teacher, student, homework, exams, JavaFX, MySQL, fxml.

Indice

Indice delle figure

1. Introduzione

In primo luogo, si può dire che sarà implementato e progettato un'applicazione desktop, sviluppato con il linguaggio Java. Questa applicazione sarà uno strumento per la gestione scolastica come aula virtuale, dove professori e studenti avranno a disposizione tutte le informazioni necessarie per il corso. L'interfaccia grafica sarà sviluppata in Javafx, mentre per salvare i dati verrà creato un database in Mysql Workbench.

Per la realizzazione di questa applicazione effettueremo diverse indagini presso gli utenti (professori e studenti). Questi saranno analizzati con l'obiettivo di rendere l'applicazione a piacere degli utenti. Dopo aver eseguito i prototipi dell'interfaccia, effettueremo nuovamente indagini per verificare che l'interfaccia non sia difficoltosa per l'utente finale e faremo lo stesso nelle varie fasi dell'interfaccia fino alla versione finale.

1.1. Motivazione

In primo luogo, la nostra generazione sta vivendo nell'era della tecnologia, che avanza a grande velocità, perché è in costante evoluzione. Per questo, voglio fare riferimento al mio passato scolastico, in cui ho avuto a malapena contatto con la tecnologia quando ero alle elementari e superiori. A seguito di ciò, mi venne l'idea di sviluppare uno strumento in cui gli studenti potessero avere a disposizione tutte le informazioni sul corso. Oggi questo fenomeno esiste nelle università, con le aule virtuali, ma è poco diffuso nelle scuole materne e nelle scuole secondarie obbligatorie. L'applicazione di questo strumento nelle scuole fornirebbe sia agli insegnanti che agli studenti un'organizzazione del corso.

Per quanto sopra esposto, eseguirò questa tesi di laurea, con gli obiettivi che saranno descritti di seguito.

1.2. Obiettivi

L' obiettivo di questa tesi è quello di fornire sia agli studenti che agli insegnanti un'organizzazione dell'anno scolastico. A tal fine è necessario che gli insegnanti siano in grado di:

- Gestire i compiti per gli studenti (creare, modificare ed eliminare).
- Gestire la data degli esami.
- Accomodare gli studenti a una valutazione.
- Aggiungere orario del corso.

Gli studenti possono inoltre:

- Vedere le task imposti dagli insegnanti.
- Vedere la data degli esami.
- Vedere le note degli esami.
- Vedere l'orario del corso.

1.3. Struttura della memoria.

Questo documento si compone di sei punti. Il primo illustra brevemente il problema da risolvere e definisce gli obiettivi che l'attuazione deve raggiungere. C'è anche una piccola riflessione sul contatto tecnologico nella mia scuola.

Nel secondo punto verrà illustrato il contesto tecnologico in cui ci troviamo, spiegando tutto il materiale che abbiamo utilizzato e i linguaggi utilizzati nel corso de la tesi. Si parla anche di applicazioni simili.

Il terzo punto definisce i requisiti funzionali utilizzando diversi diagrammi (diagramma delle classi e diagramma dei casi d'uso).

Il quarto punto riguarda la progettazione del l'applicazione e una valutazione della progettazione, poi passa al l'analisi delle esigenze del l'utente, dove saranno inviati questionari agli utenti. Viene inoltre illustrato il progetto della database.

Il quinto punto spiega l'attuazione dell'applicazione. Dopo l'analisi si passa a spiegare come è stata realizzata l'implementazione commentando i diversi strati utilizzati, passando per lo strato logico, lo strato di presentazione e infine lo strato di persistenza.

Infine, si esporrà la conclusione con possibili miglioramenti per il futuro dell'attuazione.

2. Contesto tecnologico

In questo paragrafo mi soffermerò sulla situazione tecnologica in cui ci troviamo per quanto riguarda le applicazioni che già esistono e che svolgono funzioni simili, formuleremo una breve critica nei confronti delle più importanti e presenteremo la proposta di miglioramento.

2.1. Applicazioni esistenti

Attualmente esistono diverse applicazioni per la gestione scolastica, quali:

- Additio App [1]: Funziona come un quaderno per gli insegnanti di pianificare il corso, controllare le presenze e tenere traccia dei loro appunti.
- Apliaula [2]: Si tratta di un'applicazione web per la scuola, che aiuta a mantenere la scuola organizzata, mantenendo la comunicazione tra gli alunni, i genitori degli studenti, gli insegnanti e l'amministrazione della scuola. La versione gratuita è limitata a soli 3 studenti per classe.
- Gestione scolare [3]: È necessario avere Windows. Consente alcuni compiti di gestione accademica (ad esempio, database degli allievi, insegnanti ...) e anche amministrativi (fatturazione, emissione di ricevute e gestione dei crediti ...).
- RM Gestione accademica [4]: Piattaforma multi-utente accessibile da qualsiasi dispositivo mobile che integra l'area di gestione accademico-amministrativa e la comunicazione tra centro e famiglia.
- Algebraix [5]: È uno strumento per integrare l'esperienza scolastica degli insegnanti, studenti e genitori e amministratori.
- School Control [6]: Si tratta di una piattaforma di gestione, comunicazione e transazione di pagamenti scolastici da cui insegnanti, studenti e genitori possono consultare informazioni sulla scuola, voti, avvisi, calendario di eventi, report, compiti, così come seguire le prestazioni accademiche dello studente.

2.2. Critica le applicazioni esistenti

Come abbiamo visto, attualmente esistono diversi tipi di strumenti per la gestione scolastica. Alcune con versione premium, che lo rende a pagamento (alcune anche con un prezzo troppo alto). Altre, come la ‘gestione scolare' non hanno un'interfaccia intuitiva. D'altro canto, abbiamo ‘Algebraix' e ‘School Control' che non hanno una buona valutazione e ottengono bassi punteggi.

2.3. Proposta

Alla luce di quanto precede, abbiamo deciso di sviluppare ‘School App' uno strumento per la gestione scolastica nelle scuole elementari e secondarie obbligatorie, con i seguenti miglioramenti principali:

- Un'interfaccia intuitiva e amichevole.
- Scalabilità

A tal fine utilizzeremo un ambiente di programmazione che sarà illustrato di seguito.

2.3.1. Contesto

School App, che è come chiameremo la nostra applicazione, è stata sviluppata come un'applicazione desktop per l'esecuzione su sistemi operativi come Windows, Ubuntu, Debian e Macos, a condizione che abbiano installato la JVM di Java

Per lo sviluppo dell'applicazione abbiamo scelto di utilizzare l'ambiente di programmazione Netbeans IDE [7], è un ambiente di sviluppo integrato libero, realizzato principalmente per il linguaggio di programmazione Java. Netbeans IDE è un prodotto libero e gratuito senza restrizioni d'uso. È un framework che semplifica lo sviluppo di applicazioni per Java Swing. Il pacchetto Netbeans IDE per Java SE contiene ciò che è necessario per iniziare a sviluppare plugin e applicazioni basate sulla piattaforma Netbeans; non è richiesto un SDK aggiuntivo.

D'altro canto, va detto che esistono anche altri ambienti di sviluppo per Java, tra cui Eclipse [8], che è una piattaforma software composta da un insieme di strumenti di programmazione open-source multipiattaforma per sviluppare quello che il progetto chiama "Applicazioni Cliente Arricchito".

Questa tesi sarà sviluppato in Java [9] e Java FX [10], essendo Javafx il linguaggio utilizzato per l'interfaccia grafica, perché si tratta di un linguaggio destinato a creare applicazioni desktop, per telefoni cellulari o per webs.

Per l'interfaccia grafica utilizzeremo:

- **Scene Builder** [11]: Si tratta di uno strumento di design visivo che consente agli utenti di progettare rapidamente interfacce utente di applicazioni Javafx, senza codifica. Gli utenti possono trascinare e rilasciare componenti dell'IU in un'area di lavoro, modificarne le proprietà, applicare fogli di stile e il codice FXML per il design che si sta creando viene generato automaticamente in background. Il risultato è un file FXML che può essere combinato con un progetto Java collegando l'interfaccia utente alla logica dell'applicazione.

Inoltre, abbiamo utilizzato per la database:

- **MySQL** [12]: è un sistema relazionale di gestione dei database ed è considerato come il più popolare database open source del mondo e uno dei più popolari in generale insieme a Oracle e Microsoft SQL Server, soprattutto per ambienti di sviluppo web.

3. Specifiche dei requisiti

In questa sezione verranno spiegate le funzioni che l'applicazione deve svolgere per soddisfare le esigenze del cliente.

3.1. Diagramma dei casi d'uso

Il diagramma dei casi di utilizzo è inteso a definire quali funzioni fornirà l'applicazione dal punto di vista di diversi atti.

Sono stati definiti due ruoli: insegnante e studente. Da un lato, il ruolo del l'insegnante potrà svolgere le funzioni comuni di entrambi e, dal l'altro, funzioni quali: creare, modificare ed eliminare compiti ed esami, qualificare gli alunni, aggiungere orari. Inoltre, l'allievo avrà solo la possibilità di vedere, visualizzare i compiti imposti dagli insegnanti, vedere le qualifiche, vedere il calendario degli esami, vedere l'orario.

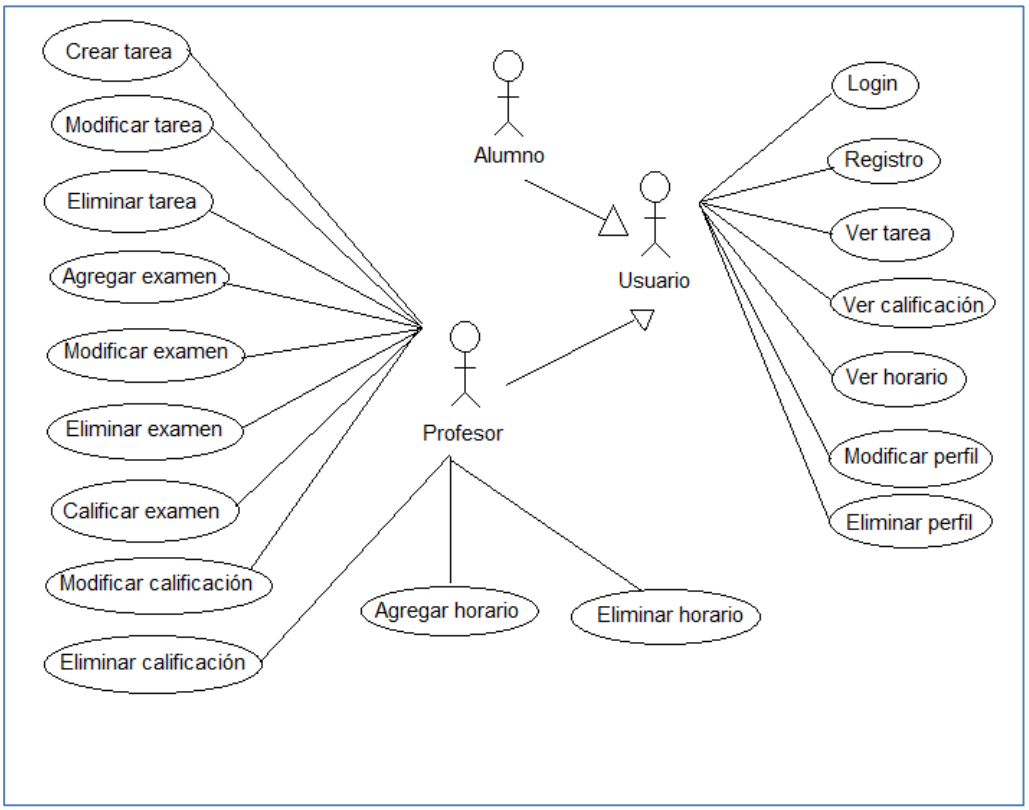

Figura 1.- Diagramma dei casi d'uso

3.2. Descrizione dei casi d'uso

I casi d'uso di cui alla figura 1 saranno illustrati più dettagliatamente qui di seguito. Si mostrerà anche alcuni flussi dell'applicazione e verrà spiegato come si comporta in caso di errori.

I seguenti due casi di utilizzo si riferiscono alla registrazione e all'accesso. Queste due azioni potranno essere realizzate sia dal l'insegnante che dal l'alunno.

Caso d'uso	Registro
Attori	Professore / Studente
Proposito	Registrare un utente che non esiste nel sistema
Sintesi	L'utente deve inserire i propri dati per registrarsi
Precondizioni	Sarà stata scelta l'opzione "Signup" nel menu di avvio
Postcondizioni	L'utente viene registrato

Estensioni asincrone
#1. Se viene inserito un nome utente esistente, viene visualizzato un messaggio di errore.
#2. Se le password non corrispondono, apparirà un messaggio di errore.
#3. Se il formato e-mail non è valido, viene visualizzato un messaggio di errore.
#4. Se la carta d'identità non è corretta, viene visualizzato un messaggio di errore.

Caso d'uso	Effettuare l'accesso
Attori	Professore / Studente
Proposito	Entrare nel sistema
Sintesi	L'utente deve inserire i dati corretti per accedere al sistema
Precondizioni	Scelta dell'opzione "Login" nel menu di avvio
Postcondizioni	L'utente entra nel sistema

Estensioni asincrone
#1. Se viene inserito un nome utente inesistente, viene visualizzato un messaggio di errore.
#2. Se la password non è corretta, viene visualizzato un messaggio di errore.

I seguenti tre casi di utilizzo si riferiscono alla gestione delle attività (creare, modificare e rimuovere).

Caso d'uso	Creare task
Attori	Professore
Proposito	Creare una task
Sintesi	Il professore crea una task, con il nome della materia e i dettagli della task
Precondizionies	Scelta dell'opzione "Creare task" nel menu principale
Postcondizioni	La task viene creata

In questa interazione non c'è alcun possibile caso di errore.

Caso d'uso	Modificare task
Attori	Professore
Proposito	Editare una task
Sintesi	Il professore modifica una task esistente
Precondizionies	Una task è stata scelta dall'elenco delle task
Postcondizioni	La task viene modificata

Estensioni asincrone
#1. Se l'attività da modificare non è creata dal professore che vuole modificarla, apparirà un messaggio di errore.

Caso d'uso	Rimuovere task
Attori	Professore
Proposito	Eliminare una task
Sintesi	Il professore elimina una task esistente
Precondizionies	Una task è stata scelta dall'elenco delle task
Postcondizioni	La task viene eliminata

Estensioni asincrone
#1. Se l'attività da eliminare non è creata dal professore che vuole eliminarla, apparirà un messaggio di errore.

Caso d'uso	Vedere task
Attori	Professore / Studente
Proposito	Vedere l'elenco delle attività
Sintesi	Gli utenti possono vedere un elenco dei compiti imposti dai professori. Nel menu "Task"
Precondizionies	Effettuare l'accesso
Postcondizioni	-

I seguenti tre casi di utilizzo si riferiscono alla gestione degli esami (creare, modificare ed eliminare).

Caso d'uso	Creare essame
Attori	Professore
Proposito	Fissare la data di un esame
Sintesi	Il professore sceglie un giorno del calendario e fissa la data del l'esame
Precondizionies	Sarà stata scelta l'opzione "Appelli" nel menu principale e successivamente si dovrà selezionare un giorno
Postcondizioni	L'esame è imposto dal calendario degli esami

Caso d'uso	Modificare essame
Attori	Professore
Proposito	Editare essame
Sintesi	Il professore seleziona un esame precedente e lo cambia di nome o di data
Precondizionies	L'opzione "Appelli" è stata scelta nel menu principale e successivamente deve essere selezionato un esame del calendario degli esami
Postcondizioni	L'esame è modificato

Caso d'uso	Eliminare essame
Attori	Professore
Proposito	Eliminare un essame
Sintesi	Il professore seleziona un esame messo in precedenza e lo cancella, quindi deve dare il pulsante "Aggiungere"
Precondizionies	L'opzione "Appelli" è stata scelta nel menu principale e successivamente deve essere selezionato un esame del calendario degli esami
Postcondizioni	L'esame è stato eliminato

I seguenti casi d'uso si riferiscono alle note d'esame, in cui gli insegnanti possono qualificare gli alunni, modificare questa nota ed eliminare questa nota in caso di errore. Inoltre, professori e alunni potranno vedere queste note in una tabella, dove per identificare lo studente si utilizzerà il loro ID, e non il loro nome per riservatezza. Avrete anche la possibilità di filtrare per studente, materia, esame o nota.

Caso d'uso	Qualificare esame
Attori	Professore
Proposito	Qualificare l'esame di uno studente
Sintesi	Il professore deve premere il pulsante "Carica studenti", una volta che gli studenti sono nella tabella, il professore deve inserire il nome della materia e il tipo di esame per poi qualificare lo studente
Precondizionies	Sarà stata scelta l'opzione "Libretto" nel menu principale
Postcondizioni	-

Caso d'uso	Modificare nota
Attori	Professore
Proposito	Modificare la nota di uno studente
Sintesi	Il professore deve selezionare "Visualizza note" in questa finestra, il professore visualizza le note degli studenti e può modificare la nota
Precondizionies	Sarà stata scelta l'opzione "Libretto" nel menu principale
Postcondizioni	La nota è modificata

Caso d'uso	Eliminare nota
Attori	Professore
Proposito	Eliminare la nota di uno studente
Sintesi	Il professore deve selezionare "Visualizza note" in questa finestra, il professore visualizza le note degli studenti e può rimuovere la nota
Precondizionies	Sarà stata scelta l'opzione "Libretto" nel menu principale
Postcondizioni	La nota è eliminata

Caso d'uso	Vedere note
Attori	Professore / Studente
Proposito	Vedere le note degli studenti
Sintesi	L'utente visualizza le note degli studenti in una tabella
Precondizionies	Sarà stata scelta l'opzione "Libretto" nel menu principale
Postcondizioni	-

Caso d'uso	Filtrare note
Attori	Professore / Studente
Proposito	Filtrare la nota di uno studente
Sintesi	L'utente può filtrare per studente, esame, materia o nota
Precondizionies	Sarà stata scelta l'opzione "Libretto" nel menu principale.
Postcondizioni	-

I seguenti casi d'uso si riferiscono alla gestione e configurazione degli orari. Professore e studente può vedere tutti gli orari che ci sono nel sistema. Oltre a questo, i professori possono aggiungere un nuovo orario o eliminare uno già esistente.

Caso d'uso	Aggiungere orario
Attori	Professore
Proposito	Aggiungere al sistema un nuovo orario
Sintesi	Il professore aggiunge un nuovo orario al sistema
Precondizionies	Sarà stata scelta l'opzione "Orario" nel menu principale.
Postcondizioni	L'orario è aggiunto nel sistema

Caso d'uso	Eliminare orario
Attori	Professore
Proposito	Eliminare un orario del sistema esistente
Sintesi	Il professore elimina un orario
Precondizionies	Sarà stata scelta l'opzione "Orario" nel menu principale.
Postcondizioni	L'orario viene eliminato nel sistema

Caso d'uso	Vedere orari
Attori	Professore / Studente
Proposito	Vedere gli orari nel sistema
Sintesi	Professori e studenti possono vedere gli orari nel sistema
Precondizionies	Sarà stata scelta l'opzione "Orario" nel menu principale.
Postcondizioni	-

Per concludere, abbiamo i casi d'uso di gestire il profilo. In questa sezione professori e studenti potete modificare i dati del suo profilo o rimuovere il suo account.

Caso d'uso	Modificare profilo
Attori	Professore / Studente
Proposito	Modificare il profilo
Sintesi	Professori e studenti possono modificare i dati del profilo, meno il nome utente
Precondizionies	Sarà stata scelta l'opzione "Profilo" nel menu principale.
Postcondizioni	Il profilo è modificato

Estensioni asincrone
#1. Se le password non corrispondono, non è possibile modificare il profilo e apparirà un messaggio di errore.
#**2.** Se il formato e-mail non è corretto, viene visualizzato un messaggio di errore.
#**3.** Se la carta d'identità non è corretta, viene visualizzato un messaggio di errore.

Caso d'uso	Eliminare profilo
Attori	Professore / Studente
Proposito	Eliminare il profilo
Sintesi	Professori e studenti possono eliminare il proprio account dal sistema
Precondizionies	Sarà stata scelta l'opzione "Profilo" nel menu principale.
Postcondizioni	Il profilo è eliminato

Estensioni asincrone
#1. Se le password non corrispondono, non è possibile rimuovere il profilo e apparirà un messaggio di errore.

3.3. Diagramma di classi

La figura 2 è riportata di seguito. Si tratta del diagramma di classe dell'applicazione.

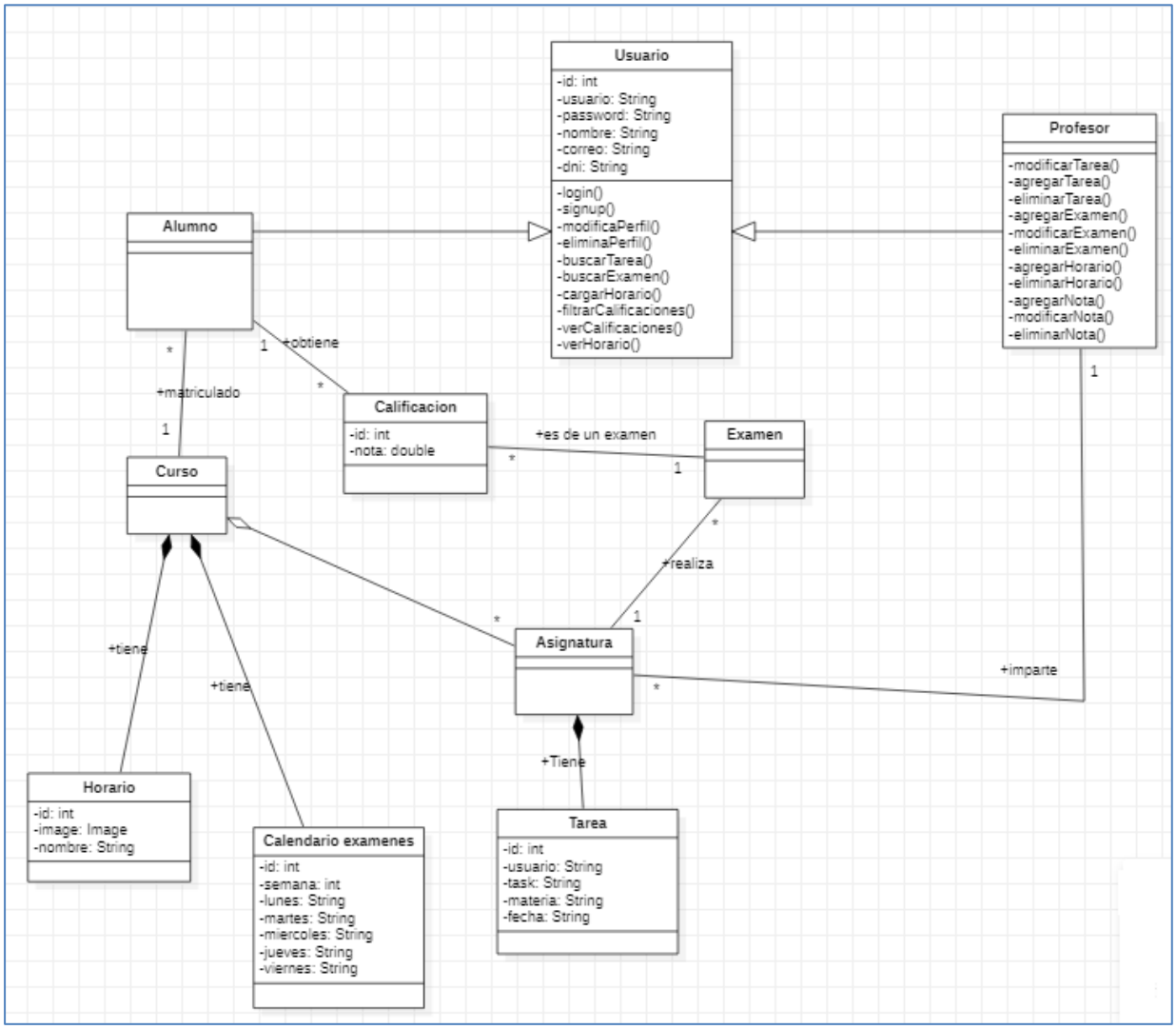

Figura 2.- Diagramma di classi

3.4. Requisiti futuri

In futuro sarebbe interessante implementare le seguenti funzionalità:

- Chat interna tra professori e studenti.
- Forum interno per eventuali domande degli studenti.
- Relazione statistica sulle note.
- Accesso da parte dei genitori degli studenti.

4. Progettazione

In questa sezione vi spiegheremo come abbiamo progettato l'interfaccia utente.

4.1. Progettazione dell'interfaccia utente

In primo luogo, quando si apre l'applicazione viene visualizzato il menu di inizio. Come possiamo vedere nella Figura 3, sul lato sinistro ci sono due pulsanti: Login e Signup. Premendo il primo pulsante abbiamo la possibilità di accedere al sistema inserendo nome utente e password. D'altro canto, premendo il pulsante Signup abbiamo la possibilità di registrarci nel sistema con i nostri dati, come possiamo vedere nella Figura 4.

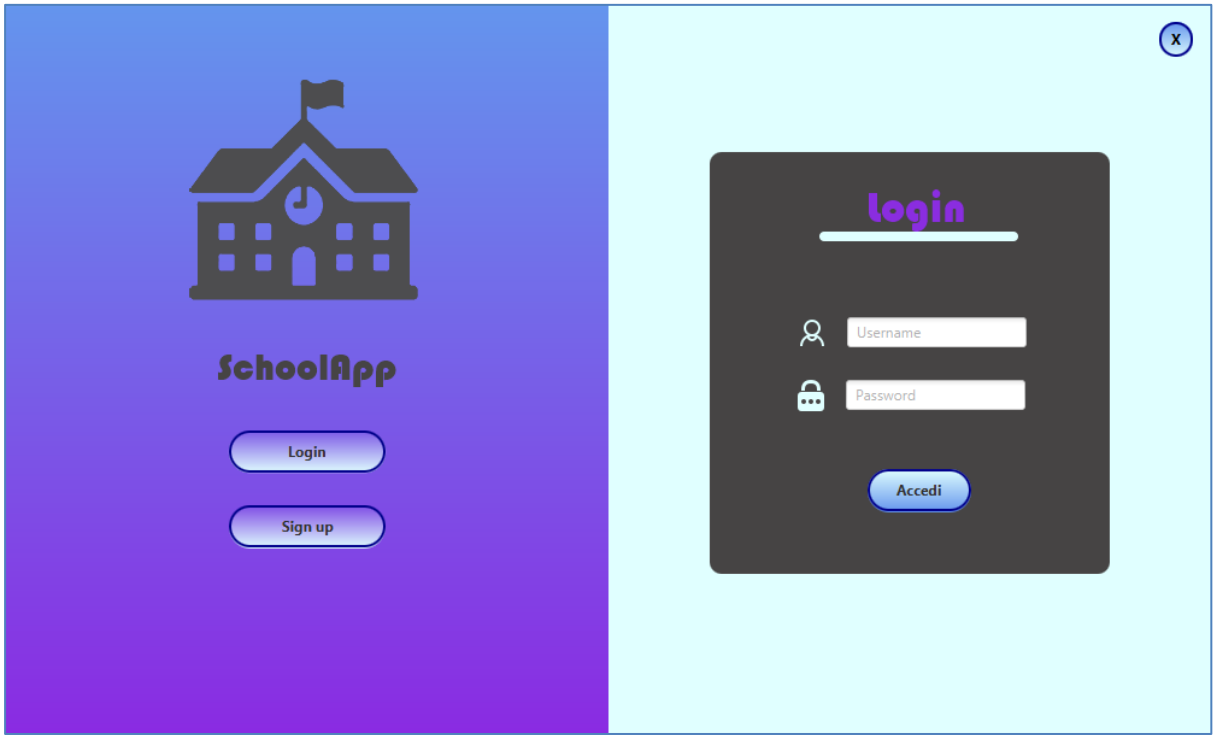

Figura 3. Menu di inizio (Login)

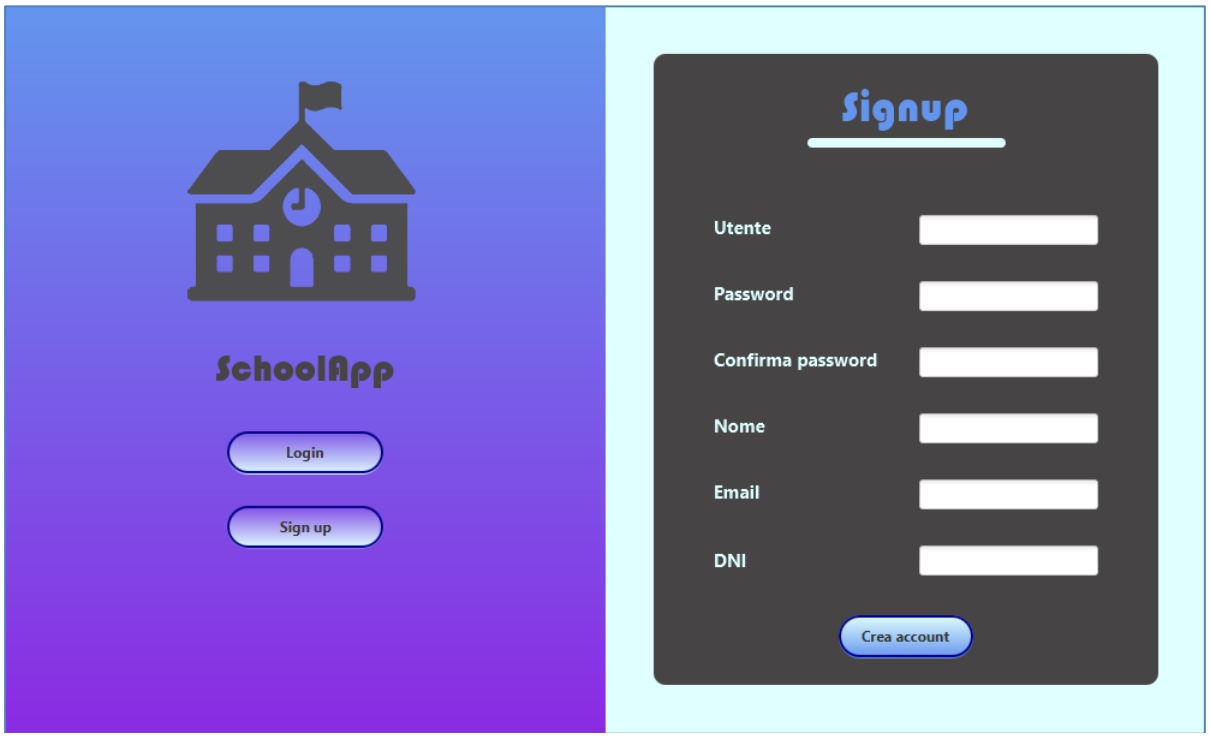

Figura 4.- Menu di inizio (signup)

Una volta che abbiamo fatto un login riuscito, siamo entrati nel menu principale. Come vediamo nella Figura 5 a sinistra abbiamo 6 pulsanti: Task, Appelli, Libretto, Orario, Profilo ed Esci.

- **Pulsante Task**: Nella vista del professore possiamo vedere una tabella con tutte le attività pubblicate dai professori, e 3 pulsanti, creare attività, modificare attività e rimuovere attività. E nella vista dello Studente possiamo vedere solo la tabella con i compiti. Nella figura 5 si vede la vista del professore.

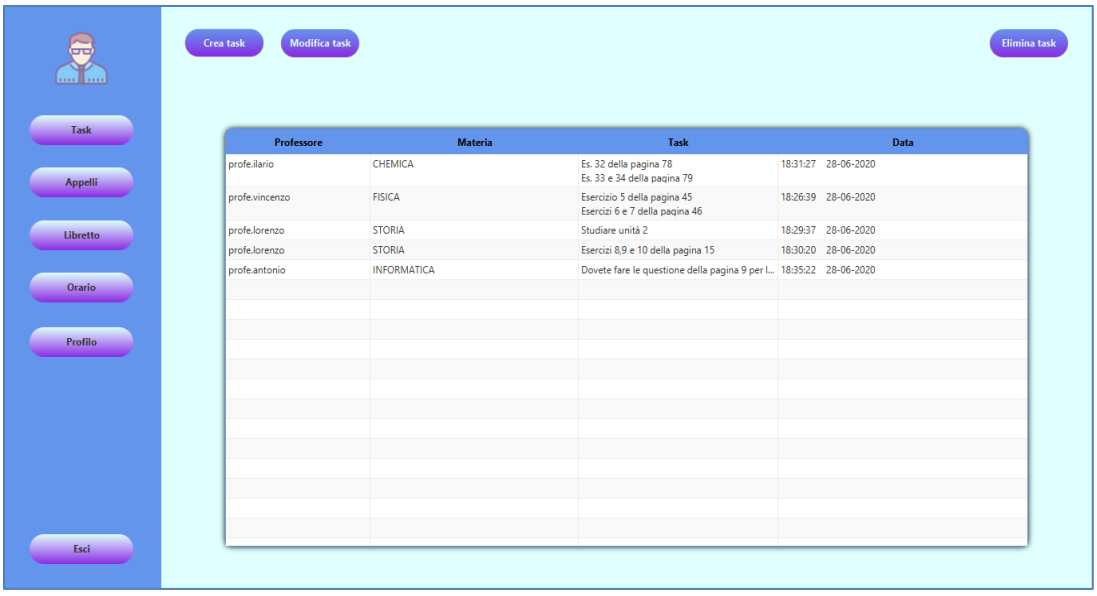

Figura 5.- Task

- **Pulsante Appelli:** Come vediamo nella figura 6, c'è un calendario di esami e i mesi dell'anno, l'unica differenza tra studenti e professori qui è che gli studenti non possono aggiungere un esame, possono solo visualizzare la tabella

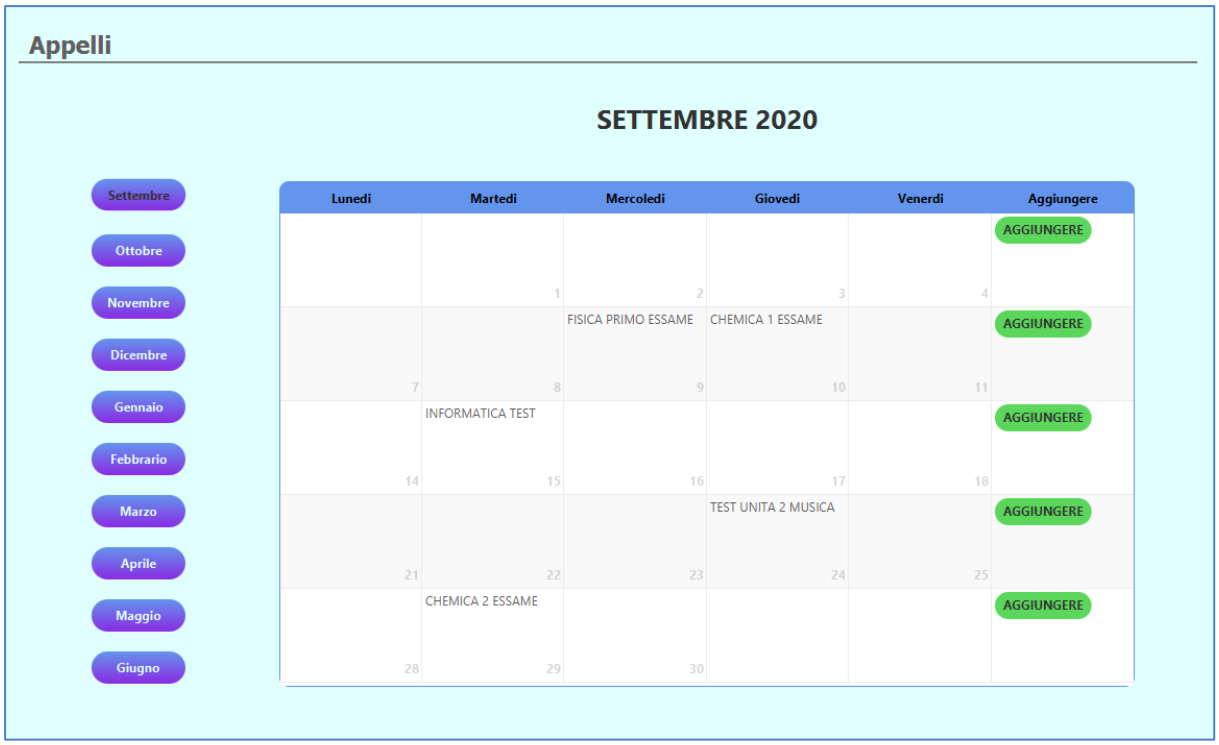

Figura 6.- Appelli

- **Pulsante Libretto:** I professori hanno due opzioni, qualificare gli studenti inserendo il nome della materia, il nome dell'esame e la nota, o vedere le note. Da parte loro gli studenti possono vedere solo le note. Figura 7 e Figura 8.

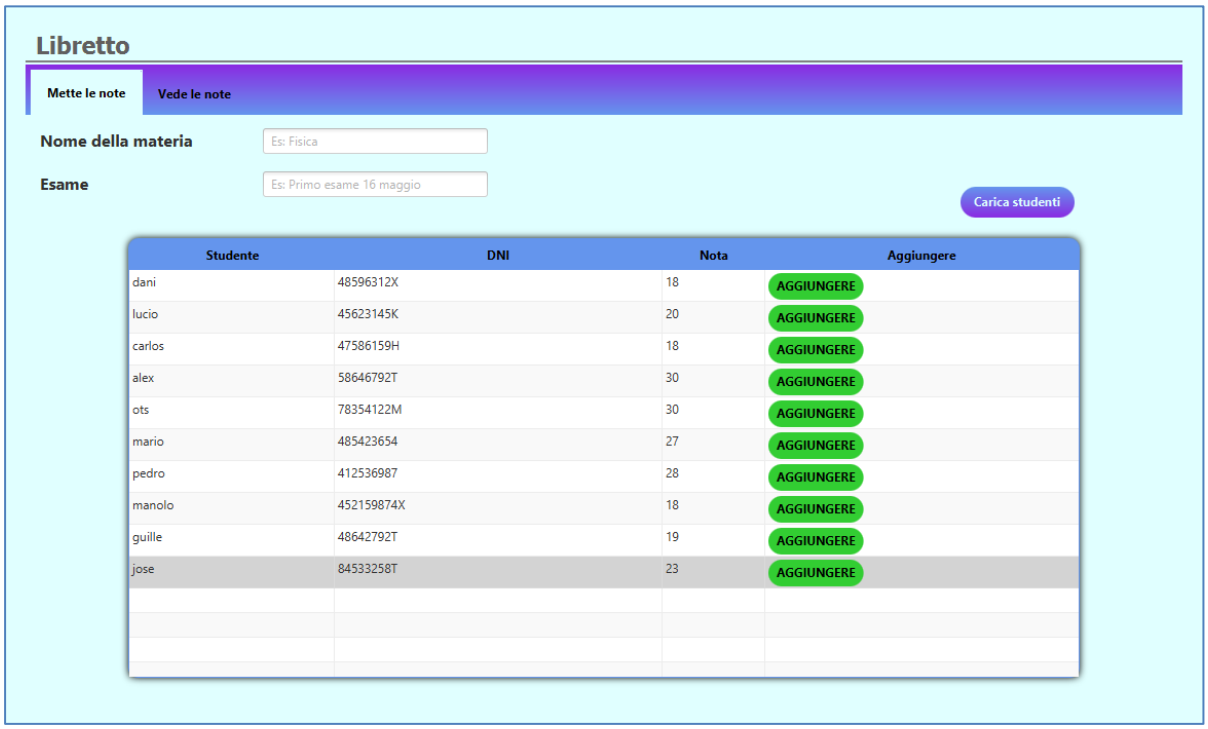

Figura 7.- Libretto 1

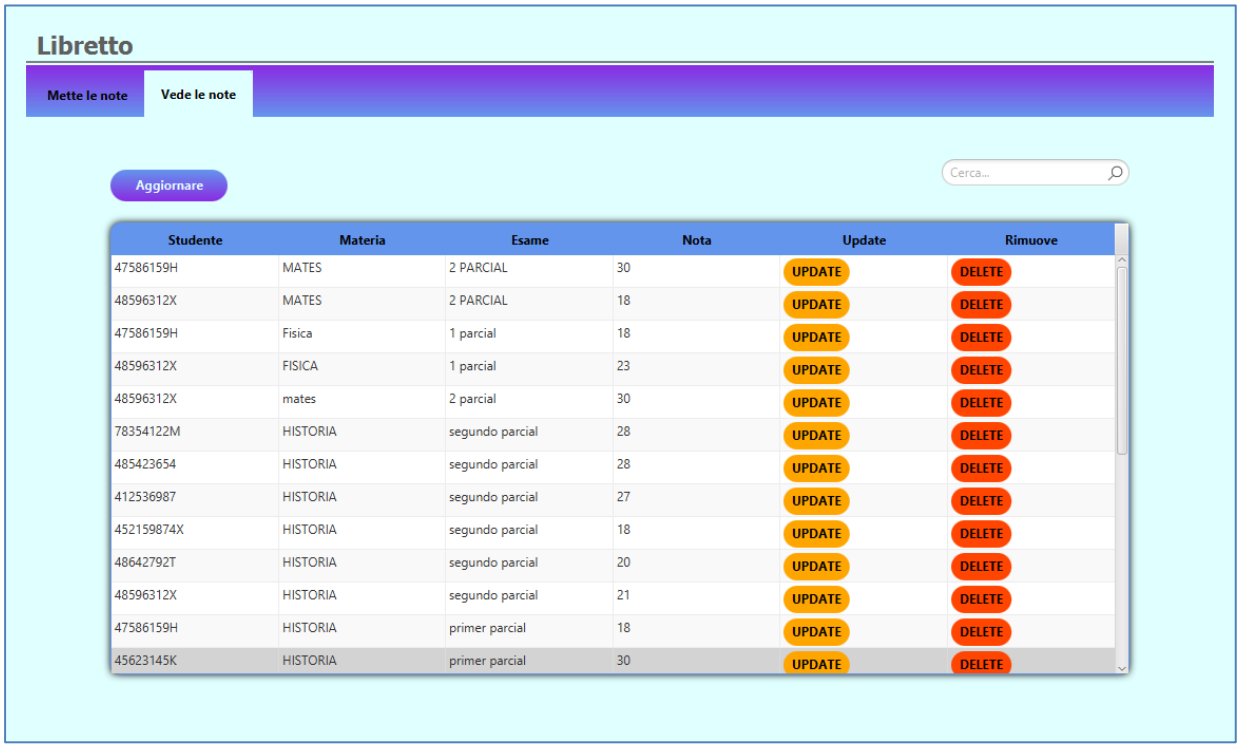

Figura 8.- Libretto 2

- **Pulsante Orario**: I professori possono gestire gli orari aggiungendo nuovi o cancellando quelli esistenti. D'altra parte, sia professori che studenti possono vedere questi orari. Figura 9 e Figura 10.

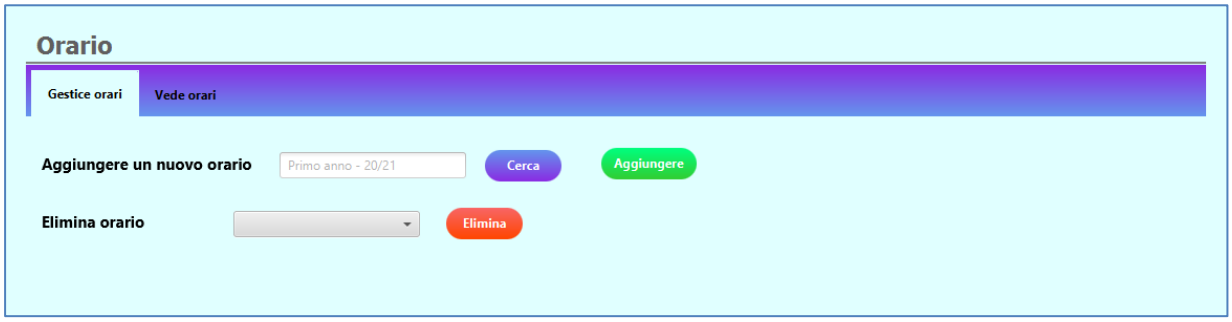

Figura 9.- Orario 1

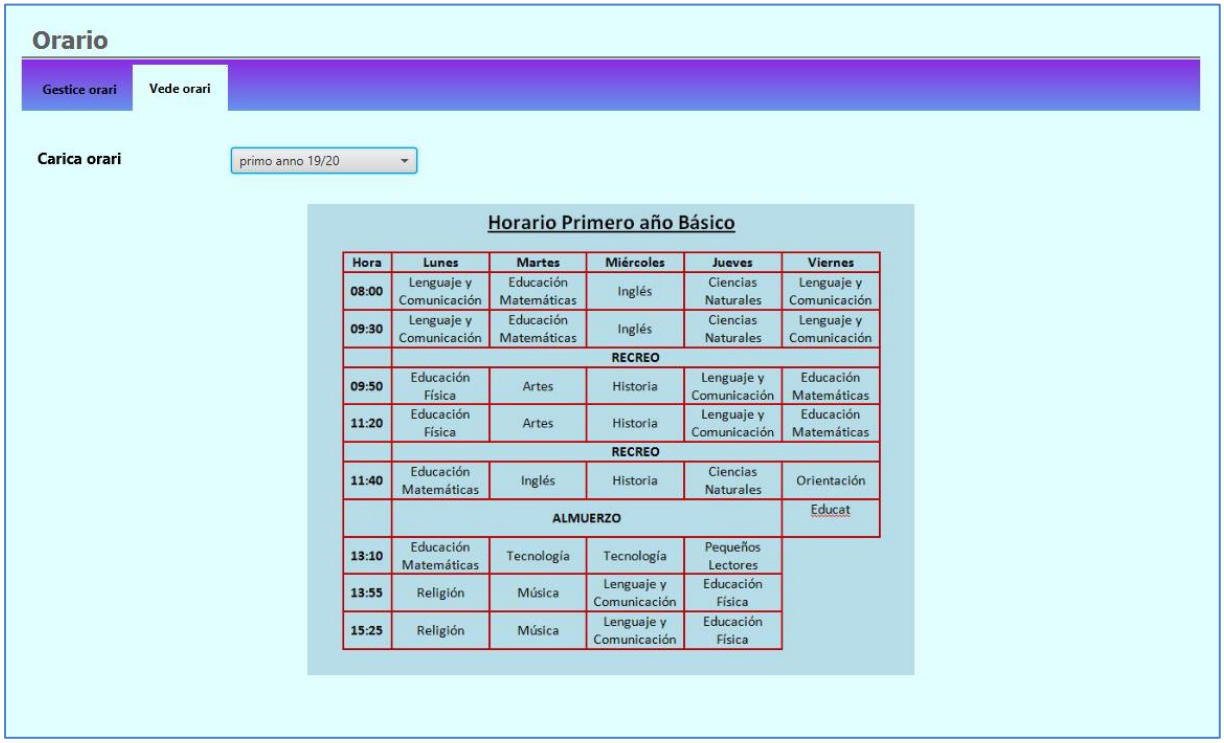

Figura 10.- Orario 2

- **Pulsante Profilo**: Questa opzione è uguale sia per studenti che per professori. Puoi modificare i dati del tuo profilo. Figura 11.

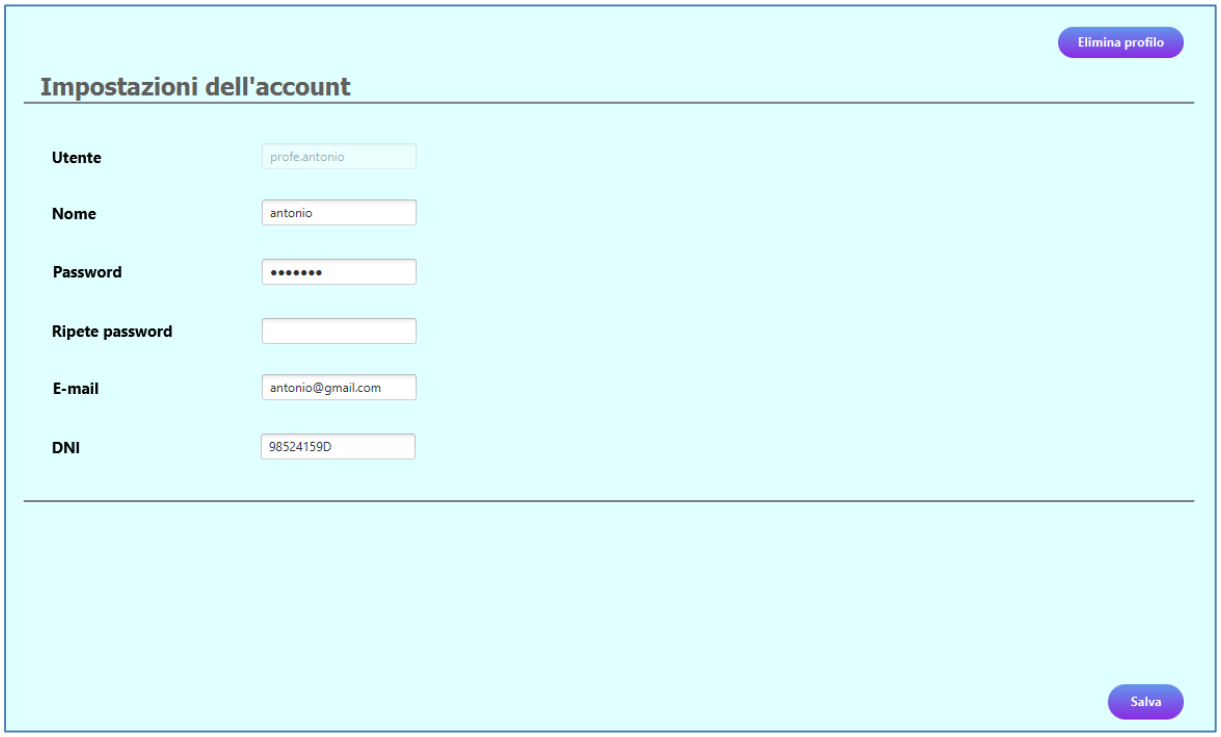

Figura 11.- Profilo

- **Pulsante Esci**: Come indicato dal nome, questo pulsante viene utilizzato per chiudere l'applicazione e uscire dal sistema. Quando viene premuto il pulsante viene visualizzato il messaggio di conferma nella figura 12.

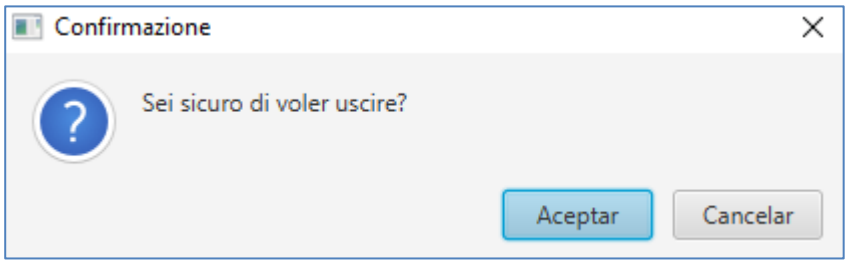

Figura 12.- Esci

4.2. Analisi dell'interfaccia

Dopo la messa a punto di questo prototipo, lo valuteremo mediante test con utenti finali che siano studenti e professori e che successivamente lo utilizzeranno. Per questo sceglieremo 1 professore e 1 studente e faremo un questionario a ciascuno di loro.

Questa valutazione mira a individuare possibili errori nell'interfaccia che influenzano l'usabilità dell'applicazione e quindi a risolverli.

4.2.1. Questionario

Successivamente presenteremo il questionario ai nostri utilizzatori intervistati e per loro presenteremo il seguente questionario:

- L'interfaccia è semplice e intuitiva?
- Manca qualche funzionalità? Cosa aggiungerei?
- In qualità di professore, desidera che questo strumento sia applicato nelle scuole?
- Come studente, volete che questo strumento venga implementato nelle scuole?

Quindi abbiamo fatto questo questionario al nostro utente principale e il risultato dell'intervista è stato il seguente:

Domanda: L'intefaccia è semplice e intuitiva?

Risposta: Sì, si tratta di una progettazione semplice e amichevole che potrebbe essere utilizzato sia nelle scuole che nelle scuole.

Domanda: Manca qualche funzionalità? Cosa aggiungerei?

Risposta: Aggiungerei qualche tipo di corriere tra professori e studenti, per poter avere un contatto diretto tra loro in caso di dubbio. D'altra parte, mi manca qualche tabella statistica con le note dello studente, o le note generali di tutti gli studenti. Mi manca anche la possibilità di caricare documenti come appunti, o esami risolti.

Domanda: In qualità di professore, desidera che questo strumento sia applicato nelle scuole?

Risposta: Sì, a noi professori servirebbe questo tipo di sistemi nelle scuole, perché aiuterebbe a mantenere un'organizzazione del corso.

Domanda: Come studente, volete che questo strumento venga implementato nelle scuole?

Risposta: Sì, mi sentirei molto a mio agio ad avere questo strumento nella mia scuola, visto che potrei avere un'organizzazione del corso. Avere a portata di mano tutti i compiti, esami, note, l'orario in un semplice strumento.

4.2.2. Conclusioni al termine dei questionari

Analizzando le risposte dei nostri intervistati, siamo giunti alla conclusione che l'applicazione dello strumento nelle scuole e negli istituti avrà un effetto positivo. D'altra parte, sarebbe necessario aggiungere alcune funzionalità extra come una sezione dove caricare appunti o esami risoluti o un sistema di messaggistica tra professori e studenti per risolvere dubbi o approccio di loro.

5. Implementazione

In questa sezione spiegheremo come abbiamo realizzato l'intero processo di implementazione e spiegazione del codice Java utilizzato nell'applicazione.

5.1. Architettura

Per questa applicazione utilizzeremo l'architettura in tre strati: strato di presentazione, strato logico e strato di persistenza.

Lo strato di presentazione è quello che vede l'utente che utilizza l'applicazione e quindi nel nostro caso sarà il pacchetto "vista" del nostro progetto, come possiamo vedere nella Figura 13.

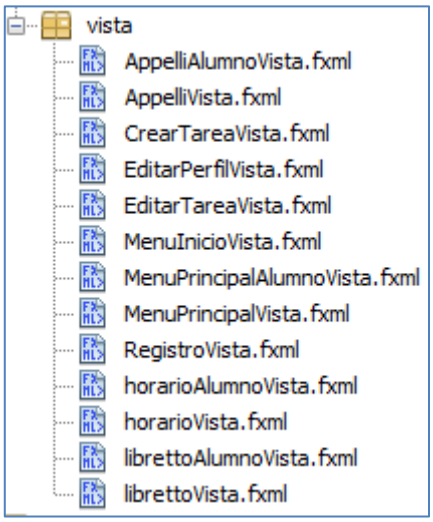

Figura 13.- Pacchetto "vista"

Per quanto riguarda lo strato logico è quello che contiene tutte le classi che implementano tutte le richieste che l'utente farà quando utilizza l'applicazione e sarà anche il livello che comunica lo strato presentazione con lo strato di persistenza, che nel nostro caso lo strato logico è completato dalla cartella del progetto "driver", come vediamo in Figura 14.

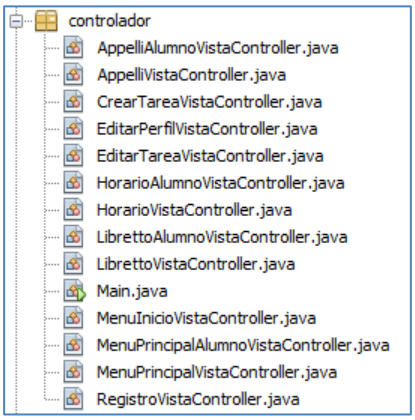

Figura 14.- Pacchetto "controlador"

Per quanto riguarda lo strato di persistenza possiamo dire che è lo strato in cui si trova il collegamento al database dell'applicazione e le informazioni relative agli orari, il calendario degli esami, le note e il profilo. Figura 15.

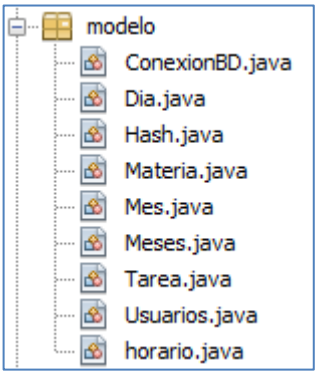

Figura 15.- Pacchetto "modelo"

5.2. Strati implementati

In questa sezione vi spiegheremo come abbiamo implementato ciascuno dei diversi livelli usati.

5.2.1. Strato logico

Poi spiegheremo lo strato logico che abbiamo utilizzato nella realizzazione di questo progetto.

Accesso e registrazione

In primo luogo, nella sezione di accesso, si verifica se l'utente è uno studente o un professore, una volta fatto questo si verifica se l'utente esiste. Se tutto questo è valido, si controlla che la password sia corretta per poi accedere, come vediamo in Figura 16.

```
if (txtUsuario.getText().startsWith("profe")) {
    String sql = "SELECT id, usuario, password, nombre FROM profesores WHERE usuario = ?";
    try {
        ps = con.prepareStatement(sql);
        ps.setString(1, textoUsuario);
        rs = ps.executeQuery();

        if (rs.next()) {

            if (txtPassword.getText().equals(rs.getString(3))) {...
```

Figura 16.- Controllo di accesso

Per la registrazione, vengono effettuati diversi controlli, che l'utente (professore o studente) non esista più nel sistema, si verifica che il campo password e quello di ripetere password siano uguali, Si verifica anche che il formato di posta elettronica sia valido e infine si verifica che la carta d'identità sia corretta, tutto questo lo vediamo nella Figura 17 Si verifica anche che non ci siano campi vuoti.

```
if (txtUsuario.getText().equals("") || pass.equals("") || txtNombre.getText().equals("") || txtCorreo.getText().equals("")) {
    JOptionPane.showMessageDialog(null, "Ci sono campi vuoti, devi compilare tutti i campi");
} else {
    if (pass.equals(passCon)) {
        if (existeUsuario(txtUsuario.getText()) == 0) {
            if (esEmail(txtCorreo.getText())) {
                if(txtUsuario.getText().startsWith("profe")){ ···
```

Figura 17.- Controllo registrazione

Task

Una volta inserito nel sistema, la schermata principale inizia direttamente nella sezione Task, qui è dove il professore può creare, modificare e rimuovere le attività, e queste attività possono essere visualizzate nella tabella delle attività. Per fare questo facciamo uso di due classi, Crea tareavistacontroller e Editartareavistacontroller. Come esempio vedremo come si crea un compito. Quando il professore vuole creare un compito deve riempire i campi "Materia" e "Descrizione della task", che vediamo nella Figura 18.

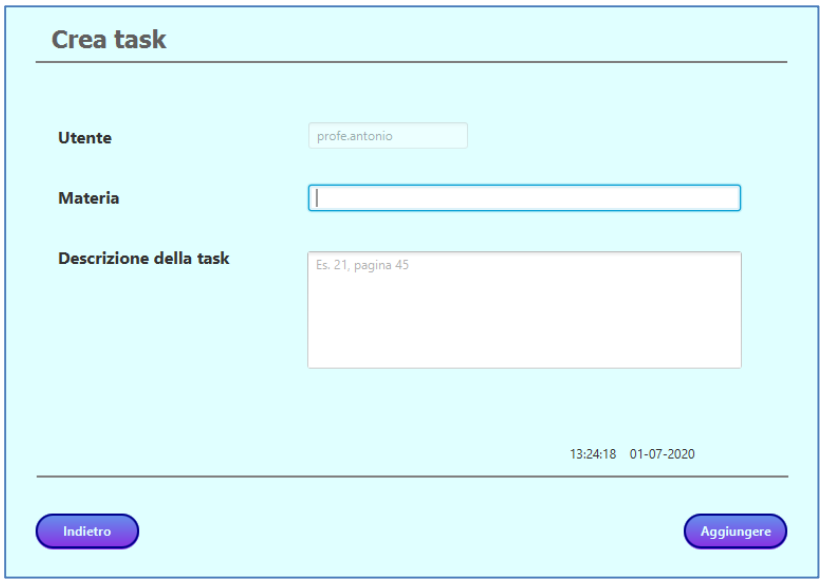

Figura 18.- Crear task

Una volta compilati, questi dati sono aggiunti alla database come vediamo nel codice di Figura 19. Se c'è stato un errore durante il salvataggio nel database, salta un errore con il testo "Errore durante il salvataggio della task in Database".

```
String sql = "INSERT INTO tablatareas(usuario, materia, task, date) VALUES(?,?,?,?)";
try {
    ps = con.prepareStatement(sql);
    ps.setString(1, usuario);
    ps.setString(2, materia);
    ps.setString(3, task);
    ps.setString(4, formato.format(date));
    ps.executeUpdate();
    JOptionPane.showMessageDialog(null, "Task creata correttamente");
} catch (SQLException ex) {
    Logger.getLogger(CrearTareaVistaController.class.getName()).log(Level.SEVERE, null, ex);
    JOptionPane.showMessageDialog(null, "Errore durante il salvataggio della task in DDBB");
}
```

Figura 19.- Creare task (codice)

In questa opzione gli studenti possono anche filtrare le task, con un metodo chiamato searchRecord(), in questo modo gli studenti possono filtrare per materia e per professore. Figura 20.

```
private void searchRecord() {
    FilteredList<Tarea> filterData = new FilteredList<>(tareas, p->true);

        tf_search.textProperty().addListener((observable, oldvalue, newvalue) -> {
            filterData.setPredicate( usr -> {
                if (newvalue == null || newvalue.isEmpty()) {
                    return true;
                }
                String lowerCaseFilter = newvalue.toLowerCase();

                if(usr.getUsuario().toLowerCase().contains(lowerCaseFilter)){
                    return true;
                }
                else if(usr.getMateria().toLowerCase().contains(lowerCaseFilter)){
                    return true;
                }else if(usr.getTask().toLowerCase().contains(lowerCaseFilter)){
                    return true;
                }
                return false;
            });
            SortedList<Tarea> sortedList = new SortedList<>(filterData);
            sortedList.comparatorProperty().bind(tb_tablaTareas.comparatorProperty());
            tb_tablaTareas.setItems(sortedList);

        });

}
```

Figura 20.- Metodo filtrare le task

Appelli

Per salvare un esame, il professore deve selezionare un giorno del calendario, mettere il nome dell'esame e poi premere il pulsante "Aggiungere", questo si traduce nel codice che vediamo in Figura 21.

```
String sql = "UPDATE sept SET lunes=?, martes=?, miercoles=?, jueves=?, viernes=? WHERE semana=?";
try {
    ps = con.prepareStatement(sql);
    ps.setString(1, lun);
    ps.setString(2, mar);
    ps.setString(3, mie);
    ps.setString(4, jue);
    ps.setString(5, vie);
    ps.setInt(6, sem);
    ps.executeUpdate();
    JOptionPane.showMessageDialog(null, "Data messa correttamente");
} catch (SQLException ex) {
    Logger.getLogger(CrearTareaVistaController.class.getName()).log(Level.SEVERE, null, ex);
    JOptionPane.showMessageDialog(null, "Errore durante il salvataggio della task in DDBB");
}
```

Figura 21.- Aggiungere essame

Libretto

Qui il professore ha due opzioni, o qualificare un esame o vedere i voti degli esami. Per qualificare un esame è molto simile ad aggiungere un esame sul calendario, quindi vedremo il codice per vedere la tabella dei voti. In questa tabella, il professore può modificare o rimuovere una nota, come abbiamo visto in Figura 4.6. Per far apparire questa tabella nel sistema con i dati caricati del Database, usiamo il metodo cargaTablaVer(), come vediamo nella Figura 22.

```
ObservableList<Usuarios> listaAlumnos = FXCollections.observableArrayList();
private void cargaTablaVer(){
    try {
        rs = con.createStatement().executeQuery("SELECT * FROM tablanotas");
        while(rs.next()){
            listaAlumnos.add(new Usuarios(
                    rs.getInt("id"),
                    rs.getString("alumno"),
                    rs.getString("DNI"),
                    rs.getString("asignatura"),
                    rs.getString("examen"),
                    rs.getString("nota")
            ));
        }
    } catch (SQLException ex) {
        Logger.getLogger(MenuPrincipalVistaController.class.getName()).log(Level.SEVERE, null, ex);
    }
    this.col_idVer.setCellValueFactory(new PropertyValueFactory("id"));
    this.col_alumnoVer.setCellValueFactory(new PropertyValueFactory("DNI"));
    this.col_asignaturaVer.setCellValueFactory(new PropertyValueFactory("asignatura"));
    this.col_examenVer.setCellValueFactory(new PropertyValueFactory("examen"));
    this.col_notaVer.setCellValueFactory(new PropertyValueFactory("nota"));
    this.tb_notas.setItems(listaAlumnos);

    col_idVer.setSortType(TableColumn.SortType.DESCENDING);
    tb_notas.getSortOrder().add(col_idVer);
    tb_notas.sort();
}
```

Figura 22.- Tabella note

Orario

Nell'opzione Orario, sia professori che studenti possono vedere tutti gli orari presenti nel sistema tramite un combobox. Per questo quello che bisogna fare è, caricare gli orari che ci sono nel database al combobox con il metodo comboHorario() e poi con il metodo comboHorarioAction() scegliamo uno specifico per mostrarlo sullo schermo come vediamo nella Figura 23.

```
private void comboHorario() {
    String sql = "SELECT textoHorario FROM horario";
    try {
        ps = con.prepareStatement(sql);
        rs = ps.executeQuery();
        while (rs.next()) {
            options.add(rs.getString("textoHorario"));
            cbHorario.setItems(options);
        }
    } catch (SQLException ex) {
        Logger.getLogger(HorarioVistaController.class.getName()).log(Level.SEVERE, null, ex);
    }
}

private void comboHorarioAction(ActionEvent event) throws FileNotFoundException, IOException {
    String sql = "SELECT imagen FROM horario WHERE textoHorario = ?";
    try {
        ps = con.prepareStatement(sql);
        ps.setString(1, (String) cbHorario.getSelectionModel().getSelectedItem());
        rs = ps.executeQuery();

        while (rs.next()) {
            Blob blob = rs.getBlob(1);
            InputStream is = blob.getBinaryStream();
            Image image = new Image(is);
            imageView2.setImage(image);
        }
    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null, "Impossibile caricare l'immagine");
        Logger.getLogger(HorarioVistaController.class.getName()).log(Level.SEVERE, null, ex);
    }
```

Figura 23.- Combobox Orario

Profilo

Infine, nell'opzione del profilo, sia professori che studenti possono modificare tutti i dati tranne il nome utente. Per cambiare qualsiasi dato o rimuovere l'account è necessario confermare la password, se le password non corrispondono verrà fuori un messaggio di errore. Per cambiare la posta elettronica è necessario avere un formato valido come per cambiare la carta d'identità. Nella Figura 24, possiamo vedere una parte del metodo guardar(), in questo metodo si verifica che le password coincidano e che il email abbia un formato valido con un metodo esterno esEmail(), che lo vediamo nella Figura 25.

```
if (pass.equals(passCon)) {
    if (esEmail(txtCorreoMod.getText())) {

        try {
            String passw = txtPasswordMod.getText();
            String name = txtNombreMod.getText();
            String email = txtCorreoMod.getText();
            String dni = txtDNIMod.getText();

            String sql = "UPDATE alumnos SET password='" + passw + "', nombre='" + name + "', correo='" +
                    email + "', DNI='" + dni + "' WHERE usuario='" + textoUsuario + "'";
            ps = con.prepareStatement(sql);

            ps.executeUpdate();
            ps.close();
            JOptionPane.showMessageDialog(null, "Modificato correttamente");
        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(null, "Errore al modificare");
            Logger.getLogger(EditarPerfilVistaController.class.getName()).log(Level.SEVERE, null, ex);
        }

    } else {
        JOptionPane.showMessageDialog(null, "Formato di e-mail invalido");
    }

}else{
    JOptionPane.showMessageDialog(null, "Le password non corrispondono");
}
```

Figura 24.- Metodo guardar() profilo modificato

```
public static boolean esEmail(String correo){
    Pattern pattern = Pattern.compile("^([0-9a-zA-Z]([_.w]*[0-9a-zA-Z])*@([0-9a-zA-Z][-w]*[0-9a-zA-Z].)+([a-zA-Z]{2,9}.)+[a-zA-Z]{2,3})$");
    Matcher mather = pattern.matcher(correo);
    return mather.find();
}
```

Figura 25.- Metodo esEmail()

5.2.2. Strato di persistenza

In questa sezione vi spiegheremo come abbiamo realizzato lo strato di persistenza, per il quale abbiamo fatto uso di Mysql Workbench.

- **Progettazione**

Al fine di realizzare il database, abbiamo realizzato il diagramma di classe UML, che rappresenta le tabelle utilizzate per la banca dati.

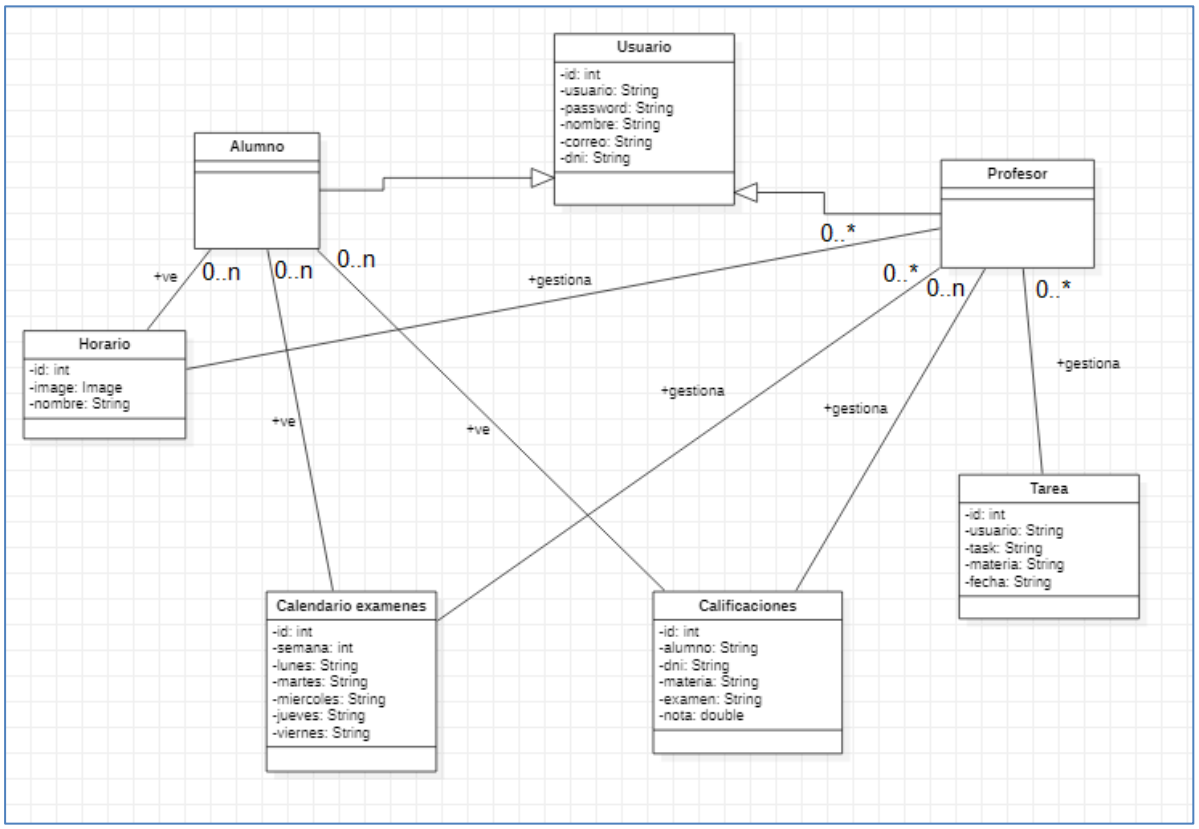

Figura 26.- Diagramma di classi: database

- **Implementazione**

Per integrare il database al progetto, prima dobbiamo importare la libreria "mysql-connector-java-5.1.48" a Netbeans IDE, poi abbiamo creato una classe chiamata ConexionBD.java, che contiene i metodi per connettersi al database chiamato "bdd" di Mysql Workbench come possiamo vedere nella Figura 27.

```
Connection con;
private final String user = "guille";
private final String password = "Guille20.";
private final String url = "jdbc:mysql://151.75.60.214:3306/bdd";

public ConexionBD(){
    con = null;
    try{
        Class.forName("com.mysql.jdbc.Driver");
        con = DriverManager.getConnection(url, user, password);

    }catch(ClassNotFoundException | SQLException e){
        System.err.println("Error: " + e);
    }
}

public Connection getConexion(){
    return con;
}
```

Figura 27.- Classe connessa al database

- **Esempio concreto di database**

Come esempio di salvataggio dei dati nel database, utilizzeremo la creazione di una task da parte del professore. Come vediamo nella Figura 28, inseriamo i record utente, materia, task data nella tabella "tablatareas" del database "bdd". I record materia e task sono caricati dal testo che introduce il professore. Da parte sua il registro utente viene precaricato già come l'utente corrente.

```
String usuario = this.txtNombreUsuario.getText();
String materia = this.txtNombreMateria.getText();
String task = this.txtDescripcionTask.getText();
String fecha = this.labelFecha.getText();

String sql = "INSERT INTO tablatareas(usuario, materia, task, date) VALUES(?,?,?,?)";
try {
    ps = con.prepareStatement(sql);
    ps.setString(1, usuario);
    ps.setString(2, materia);
    ps.setString(3, task);
    ps.setString(4, formato.format(date));
    ps.executeUpdate();
    JOptionPane.showMessageDialog(null, "Task creata correttamente");
} catch (SQLException ex) {
    Logger.getLogger(CrearTareaVistaController.class.getName()).log(Level.SEVERE, null, ex);
    JOptionPane.showMessageDialog(null, "Errore durante il salvataggio della task in DDBB");
}
Stage stage = (Stage) this.btnVolver.getScene().getWindow();
stage.close();
```

Figura 28.- Salvare task in database

5.2.3. Strato di presentazione

Poi spiegheremo come abbiamo realizzato l'interfaccia grafica utilizzata in questa applicazione.

Per l'implementazione abbiamo fatto uso di Javafx e del programma Scenebuilder per la progettazione dell'applicazione. Usando Javafx, l'interfaccia è implementata su un file .xml, e il design è fatto in un file .css, che nel nostro caso lasceremo il file .css che viene di default con Javafx.

Avvisi

Con l'obiettivo di migliorare l'esperienza dell'utente abbiamo implementato avvisi che salteranno, come, ad esempio, quando si esce dall'applicazione che viene visualizzato un messaggio di conferma a questa richiesta (Figura 29).

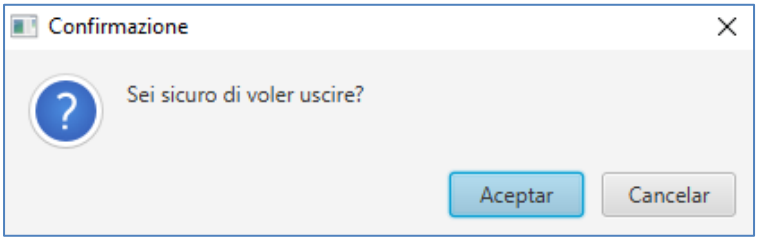

Figura 29.- Confirma uscire

Nel menu del professore vengono inoltre visualizzate avvertenze per la modifica di una task o di un esame, che deve prima selezionare una task o un esame (Figura 30).

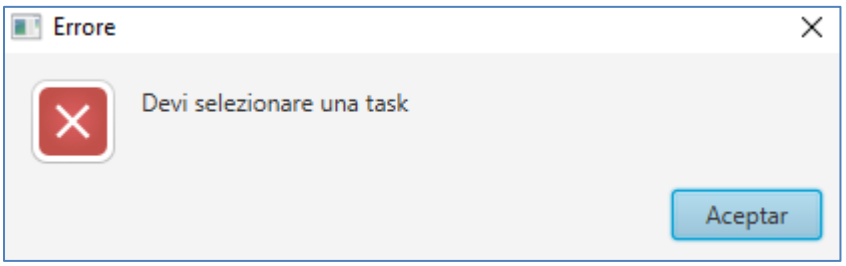

Figura 30.- Seleziona task

D'altra parte, per modificare il profilo è necessario confermare la password, se non viene confermato salta un messaggio di avviso, come se il formato e-mail non è valido (Figura 31).

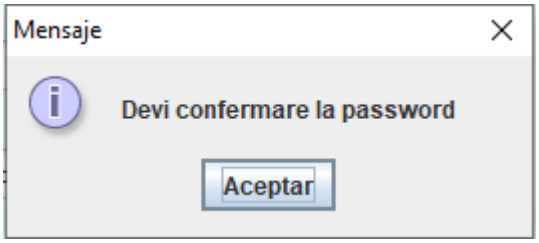

Figura 31.- Confirma password

5.3. Strumenti per lo sviluppo

In questa sezione verrà spiegato il software utilizzato per lo sviluppo dell'applicazione, nonché i motivi per cui è stato deciso di utilizzarlo.

- **NetBeans IDE**

Netbeans è uno degli IDE più famosi per programmare in Java, creato da Sun Microsystems nel 2000. È un ambiente di sviluppo che ci permette di programmare sia in programmazione Java, sia tramite plugins e moduli, ci permette di programmare altri linguaggi come C, C++, PHP, ecc.

Su Netbeans si può anche arrivare a programmare su Android utilizzando i corrispondenti plugin Nbandroid, Graddle Support, Android e Nbandroid Extensions che più avanti spiegheremo come installare sul nostro IDE.

Caratteristiche:

- Gestione dell'interfaccia utente (Menu e barre degli strumenti del linguaggio in cui è programmato, caratteri, ecc.).
- Gestione delle impostazioni utente (Aggiungi l'autore alle classi, impostazioni dei commenti, ecc.).
- Gestione di archiviazione (Salva o carica dati).
- Gestione finestra (Organizza l'IDE a piacimento del programmatore).
- Procedura guidata (Supporto Javadoc e altri precedentemente scaricati dai Siti).
- Libreria visiva di Netbeans
- Strumenti di sviluppo integrato

- **Scene Builder**

L'applicazione Scene Builder permette di progettare, tramite un'interfaccia grafica, le strutture delle finestre delle applicazioni che vogliamo sviluppare utilizzando Javafx. In questo articolo potrai conoscere le basi per iniziare ad usare questo strumento in modo integrato con l'ambiente di sviluppo Netbeans.

Per questo dobbiamo configurare Scene Builder su Netbeans seguendo le seguenti procedure:

1. Apri le Preferenze di Eclipse (menu Window | Preferences e naviga fino a Java | Installed Jres.

2. Se non hai il jre1.8 nella lista di Jres, allora premi Add… , seleziona Standard VM e seleziona la Directory di installazione (JRE Home directory) del tuo JDK 8.

3. Elimina altri Jres o Jdks in modo tale che JDK 8 diventi l'opzione di default.

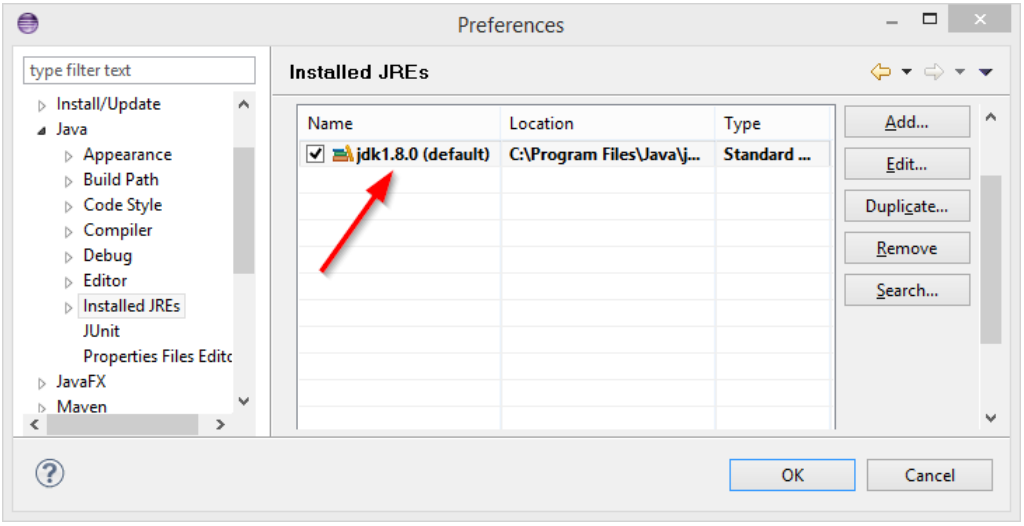

Figura 32.- Paso 3 configurazione Scene builder

4. Naviga in Java | Compiler. Imposta il livello di adempimento del compilatore a 1.8 (Livello di conformità del compilatore).

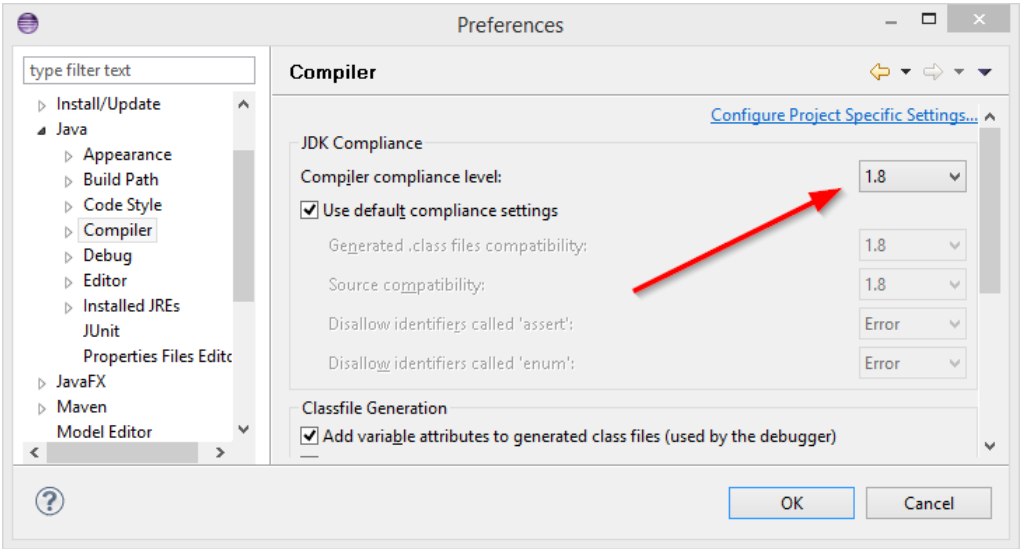

Figura 33.- Paso 4 configurazione Scene builder

5. Naviga fino a Java | Javafx. Specifica il percorso per l'eseguibile di Scene Builder.

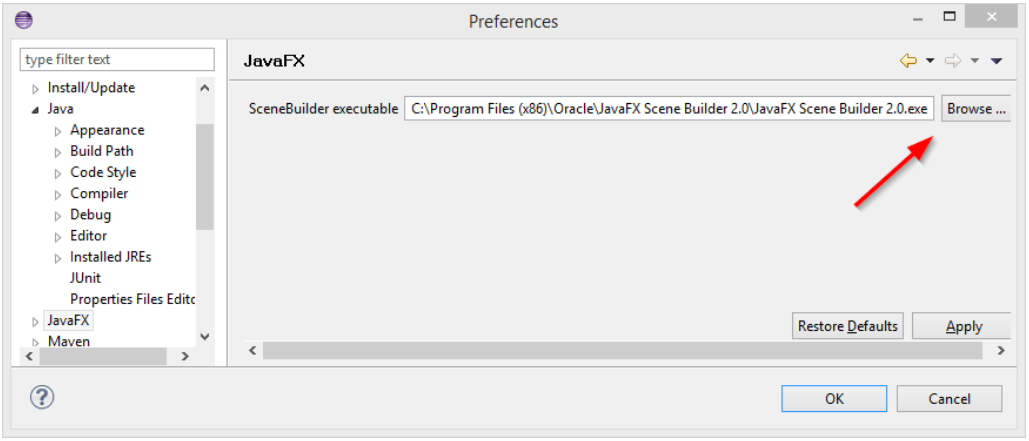

Figura 34.- Paso 5 configurazione Scene Builder

6. Conclusione

In questo paragrafo spiegheremo quali sono le conclusioni cui siamo giunti dopo la realizzazione di questo lavoro di fine grado, così come i miglioramenti che potremmo applicare su di esso in futuro.

6.1. Conclusioni

L'obiettivo principale di questo progetto era quello di progettare e attuare un'applicazione per la gestione scolastica, che fornisse sia agli insegnanti che agli studenti un controllo sul loro corso, nonché un'organizzazione di orari, note, compiti, ecc. Dopo aver completato lo sviluppo, si può affermare che tutti gli obiettivi sono stati raggiunti.

D'altra parte, per quanto riguarda l'implementazione, all'inizio del progetto era più costoso perché aveva bisogno di rinfrescare alcuni concetti di Javafx e Mysql, così come imparare nuove idee di programmazione.

Infine, si parlerà dell'apprendimento acquisito con la realizzazione di questo progetto. Sono state acquisite competenze organizzative per lo sviluppo di progetti destinati alla gestione dei dati e, naturalmente, le conoscenze in materia di programmazione sono state notevolmente ampliate tra altre discipline informatiche.

Da quanto sopra esposto possiamo dire che l'applicazione realizzata è riuscita a soddisfare tutti i requisiti esposti in questa memoria, e potrebbe essere implementata nelle scuole primarie e secondarie.

6.2. Miglioramenti

Qui di seguito spiegheremo eventuali miglioramenti da apportare all'applicazione per il futuro:

- Fare un profilo di padre/madre che sia legato al figlio in modo da essere al corrente di tutto il corso, sapendo così, i suoi appunti, orari, salti a lezione, compiti da compiere, ecc.
- Eseguire tabelle statistiche con i dati del corso.
- Rendere l'applicazione compatibile con più piattaforme come: iOS, Android e MacOS.

7. Bibliografia

[1] «Additio App»: https://www.additioapp.com/es

[2] «ApliAula»: http://www.apliaula.com/

[3] «Gestion Escolar»: https://gestion-escolar.softonic.com/

[4] «RM Gestion Academica»: https://www.rminformaticos.com/rm-escuelas/

[5] «Algebraix»: https://www.algebraix.com/

[6] «School Control»: http://www.schoolcontrol.com/

[7] «NetBeans IDE»: https://netbeans.org/

[8] «Eclipse»: https://www.eclipse.org/

[9] «Java,»: https://www.java.com/es/

[10] «JavaFX»: http://www.oracle.com/technetwork/es/java/javafx/downloads/

[11] «Scene Builder»: https://www.oracle.com/java/technologies/javase/javafxscenebuilder-info.html

[12] «MySQL»: https://www.mysql.com/