**WineShop – Federico Risoli Elia Candida**

**Summary**

# Summary 1 Introduction 1 Revision History 2 Original Assignment 2-4 Requirements 5-6 Use Case Documentation 7-14 Architectural Model 15 Detailed Software Model 16-31 Functional Tests 32 System Manual: 33 Installation 33 Configuration 34 User Guide 35-48

# Changelog 49-57

# 

**Introduction**

# WineShop è un’applicazione client-server che permette a clienti e dipendenti di un negozio di vini, di interfacciarsi con un Sistema che permette ai clienti di comprare e ordinare vini e ai dipendenti di gestire gli ordini, clienti e altro, tutto in un unico Sistema.

**Revision History**

| **Version** | **Date** | **Author(s)** | **Description** |
| --- | --- | --- | --- |
| **1** | **Fine dicembre** | **Elia Candida-Federico Risoli** | **Inizio documentazione e progetto** |
| **2** | **Metà gennaio** | **Elia Candida-Federico Risoli** | **Implementate metà delle funzionalità** |
| **3** | **Metà marzo** | **Elia Candida-Federico Risoli** | **Bugfix, fine Progetto e documentazione ultimata** |

**Original Assigment**

L’obiettivo è sviluppare (usando il linguaggio Java) e documentare un sistema software per la vendita online di bottiglie di vino, utilizzando in modo appropriato le tecniche di riferimento della programmazione orientata agli oggetti. Il sistema realizzato sarà installato presso l’azienda che ha finanziato la realizzazione del sistema.

Il sistema interagisce con clienti (persone che vogliono acquistare del vino), impiegati (persone che gestiscono la vendita e la spedizione e ricezione dei prodotti) e un amministratore (persona che controlla il lavoro dei dipendenti e gestisce le attività finanziarie dell’azienda).

I clienti sono identificati da: nome, cognome, codice fiscale, indirizzo email, numero telefonico e indirizzo di consegna. Gli impiegati e l’amministratore sono identificati da: nome, cognome, codice fiscale, indirizzo email, numero telefonico e indirizzo di residenza. Per le consegne dei vini ai clienti e per la fornitura dei vini, l’azienda si serve dei servizi di un corriere e di un fornitore. Queste due persone sono identificate da: nome, cognome, indirizzo email, numero telefonico, e codice fiscale e indirizzo della loro azienda. I vini, sono identificati dai seguenti attributi: nome, produttore, provenienza, anno, note tecniche, e i vitigni da cui derivano. Ogni bottiglia ha un prezzo che dipende dalla qualità del vino e dal suo numero di vendite.

Per poter utilizzare il sistema, un nuovo cliente deve registrarsi sul sistema inserendo nome utente e password e le altre informazioni richieste dal sistema. Un cliente registrato può fare ricerca dei vini per nome e anno di produzione e acquistare bottiglie di vino dopo un accesso autenticato. Ogni volta che un cliente accede nel sistema, il sistema informa il cliente sulle promozioni in corso. Anche gli impiegati e l’amministratore devono accedere al sistema tramite nome utente e password. e possono modificare la loro password. In aggiunta, l’amministratore ha una password iniziale predefinita che può comunque modificare. Inoltre l’amministratore deve registrare (con nome utente e password) gli altri dipendenti, e resettare e cancellare i loro dati di registrazione quando diventa necessario. Gli impiegati e l’amministratore possono fare la ricerca dei clienti per cognome, dei vini per nome e/o anno di produzione, degli ordini di vendita e di acquisto e delle proposte di acquisto per intervallo di date. Inoltre, l’amministratore deve preparare un report mensile sullo stato dell’azienda (introiti, spese, numero bottiglie vendute e disponibili alla vendita, numero di vendite per i diversi vini, valutazione dei dipendenti).

Un cliente può acquistare le bottiglie di vino con confezioni da 1 a 5 bottiglie e da casse da 6 e 12 bottiglie. Ognuna di queste casse contiene solo bottiglie dello stesso tipo di vino. La vendita di casse di vino di 6 e 12 bottiglie è valorizzata da uno sconto del 5 e 10%. La vendita di più di una cassa è valorizzata da un ulteriore sconto del 2 e 3%. Il sistema deve tenere traccia per ogni tipo di vino del numero di bottiglie vendute e di quelle ancora disponibili per la vendita.

Un cliente può visualizzare i vini in vendita e selezionare uno o più vini e, per ogni vino, selezionare il numero di bottiglie (default 1). Selezionate le bottiglie, il cliente può decidere di acquistarle. Il sistema permette questa operazione fornendo la possibilità di pagare tramite un bonifico o una carta di credito. Fatto il pagamento, il cliente riceve un ordine di vendita contenente le informazioni su: il cliente che ha fatto l’acquisto, i vini acquistati (nome vino, numero bottiglie e prezzo), l’indirizzo di consegna e la data di consegna.

Quando non c’è un numero sufficiente di bottiglie di uno o più vini per soddisfare la richiesta del cliente del numero di bottiglie che vorrebbe acquistare: quando le bottiglie richieste saranno disponibili, il sistema notificherà la loro disponibilità all’utente.

Nel caso in cui il numero di bottiglie di uno o più vini non riesce a soddisfare la richiesta del cliente, allora il cliente può compilare una proposta di acquisto contenente: i vini che vuole acquistare (nome vino, numero bottiglie) e l’indirizzo di consegna. Ricevuta la proposta di acquisto, il sistema invia una copia dell’ordine di vendita ad uno degli impiegati. Questo impiegato preparerà un ordine di acquisto e lo invia al fornitore. Ricevuto l’ordine, il fornito prepara il materiale ed effettua la consegna. Ricevuto il materiale dal fornitore, l’impiegato chiede al sistema di generare un ordine di vendita partendo dai dati della proposta di acquisto, dai prezzi dei vini e dalla data di consegna (definita con il corriere) e invia l’ordine di vendita. Ricevuto l’ordine di vendita, il cliente può decidere se acquistarlo o no. Se si, il cliente deve pagare tramite il sistema; se no, il cliente deve annullare l’acquisto. Il sistema e il dipendente che ha seguito la possibile vendita devono completare le attività necessarie in base alla scelta fatta dal cliente.

Il sistema mantiene e gestisce gli ordini di vendita ricevuti dai clienti. In particolare, ogni volta che viene effettuato un acquisto, il sistema aggiorna il numero di vendita e disponibilità dei vini, e invia una copia dell’ordine di vendita ad uno degli impiegati. Questo impiegato deve esaminare l’ordine, preparare il materiale per la spedizione, concordare con il corriere la data di prelievo e consegna del materiale e informare il cliente sulla data di consegna. Dopo la consegna del materiale al corriere, il dipendete deve completare l’ordine di vendita firmandolo digitalmente.

Il sistema deve garantire che per ogni tipo di vino ci sia sempre un numero ragionevole di bottiglie. Per gestire questo vincolo, ad ogni arrivo di un ordine di acquisto il sistema controlla se il numero di bottiglie di alcuni vini è sceso sotto la soglia (il valore della soglia può dipendere dal vino e dai suoi numeri di vendita). Se ciò avviene, il sistema informa uno dei dipendenti sulle quantità disponibili di questi vini. Ricevuta l’informazione, l’impiegato preparerà un ordine di acquisto e lo invia al fornitore. Ricevuto l’ordine, il fornito prepara il materiale ed effettua la consegna. Ricevuto il materiale dal fornitore, l’impiegato deve completare l’ordine di acquisto firmandolo digitalmente, e caricare l’ordine sul sistema. Il sistema elabora l’ordine e aggiorna le quantità di vini.

Il sistema è basato su un’architettura client server. Il server deve supportare l’accesso in parallelo di diversi nodi client. L’interfaccia grafica dei client deve essere implementata con JavaFX. Il database del sistema deve mantenere le informazioni di almeno una decina di vini differenti.

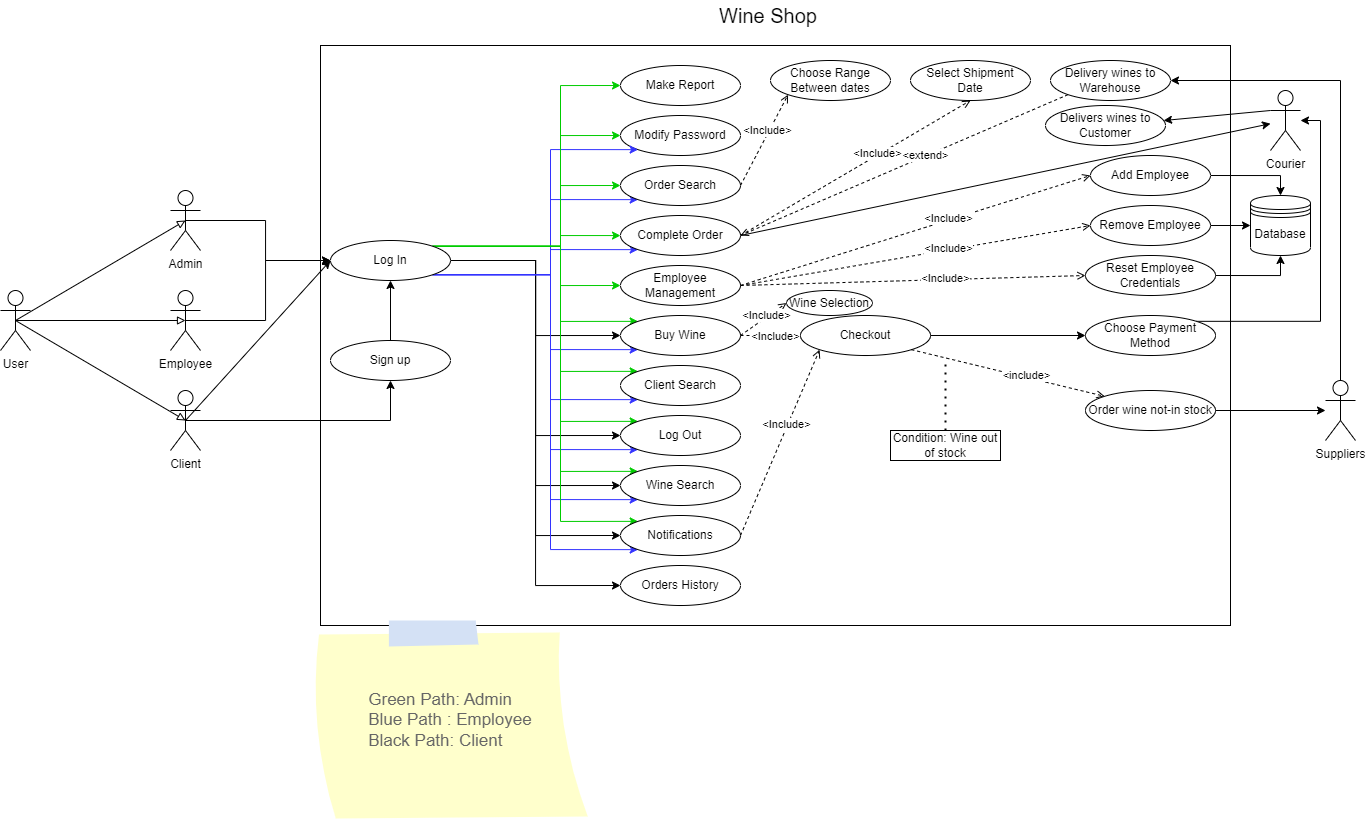
Il numero di impiegati al lavoro deve essere almeno tre e il sistema assegna le attività agli impiegati con una politica “round-robin” ovviamente tenendo conto degli impiegati al momento impegnati. Se l’attività non è terminata entro un tempo prestabilito, il sistema l’assegna ad un altro impiegato. Il sistema mantiene le informazioni sulle attività completate e non completate dai dipendenti.

# **Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Function name** | **Status** | **Priority** |
| F1 | Login | implemented | High |
| F2 | User Registration | implemented | High |
| F3 | Employee Registration | implemented | Medium |
| F4 | Modify Password | implemented | Low |
| F5 | Wine Search | implemented | High |
| F6 | Variable Wine Price | implemented | Medium |
| F7 | Popup Promozioni | implemented | Low |
| F8 | Buy Wine | implemented | High |
| F9 | Logout | implemented | Low |
| F10 | My order | implemented | Low |
| F11 | Notify | implemented | Medium |
| F12 | Complete Order | implemented | High |
| F13 | Client Search | implemented | Medium |
| F14 | Order Search | implemented | Medium |
| F15 | Delete Employee | implemented | Medium |
| F16 | Reset Password Employee | implemented | Medium |
| F17 | Generate Monthly Report | implemented | High |
| F18 | Automatic Discount | implemented | Low |
| F19 | Multiple Payment Methods | implemented | Medium |
| F20 | Automatic Orders | implemented | Medium |
| F21 | Order Wines not in Stock | implemented | High |
| F22 | Automatic Orders to Supplier for easier use | Implemented | Medium |
| F23 | Employee Can mark Order as arrived and change the price if needed | Implemented | High |
| F24 | Clients can Buy Wines that were not in stock if they still wish to do this | Implemented | High |
| F25 | Automatic Orders when Wines are Low | Implemented | High |
| F26 | Client/Server Architecture | Implemented | High |
| F27 | Round Robin Work Assignment | Implemented | High |

# 

# **Use Cases Documentation**

****

**Use cases diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 1 | | |
| Use Case Name: | Login | | |
| Created By: | Elia Candida | Last Updated By: | Federico Risoli |
| Date Created: | 18/12/22 | Date Last Updated: | 27/02/23 |

|  |  |
| --- | --- |
| Description: | The User can perform the Login into the system using the Username and the Password |
| Primary Actor: | User |
| Secondary actors: | Database |
| Pre-conditions: | The user was already Sign Up into the system by himself or an admin |
| Post-conditions: | The Username and Password must be the same of the ones into the Database |
| Trigger: | The User opens the application |
| Priority: | High |
| Frequency of Use: | High |
| Normal Course of Events: | The User inserts Username and the Password then clicks on Login |
| Alternative Courses: | The User inserts Username and the Password then clicks on Login but receives an error saying incorrect user/password and must try again |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 2 | | |
| Use Case Name: | Signup | | |
| Created By: | Elia Candida | Last Updated By: | Federico Risoli |
| Date Created: | 18/12/22 | Date Last Updated: | 20/12/22 |

|  |  |
| --- | --- |
| Description: | The User can Sign up into the System |
| Primary Actor: | Clients |
| Secondary actors: | Database |
| Pre-conditions: | The Client was not already Sign Up into the system |
| Post-conditions: | The text inserted is not Blank |
| Trigger: | The User Clicks on “Registrati” |
| Priority: | High |
| Frequency of Use: | Low |
| Normal Course of Events: | The Client writes in all the fields and then sends the form. |
| Alternative Courses: | The User inserts non-valid text in the fields and so it gets an error and must correct the non-valid text |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 3 | | |
| Use Case Name: | Make Report | | |
| Created By: | Elia Candida | Last Updated By: | Elia Candida |
| Date Created: | 19/02/23 | Date Last Updated: | 03/03/23 |

|  |  |
| --- | --- |
| Description: | The Admin generates a report detailing various things about the System |
| Primary Actor: | Admin |
| Secondary actors: | Database |
| Pre-conditions: | The Admin has logged-in |
| Post-conditions: | None |
| Trigger: | The Admin Clicks the button “Genera Report Mensile” |
| Priority: | High |
| Frequency of Use: | Low |
| Normal Course of Events: | The Admin Clicks the button “Genera Report Mensile” |
| Special Requirements: | The App must have the ability to write in its own folder |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 4 | | |
| Use Case Name: | Modify Password | | |
| Created By: | Federico Risoli | Last Updated By: | Federico Risoli |
| Date Created: | 21/12/22 | Date Last Updated: | 29/12/22 |

|  |  |
| --- | --- |
| Description: | The Admin can change the password of everyone, the Employee can change his own password |
| Primary Actor: | Admin, Employee |
| Secondary actors: | Database |
| Pre-conditions: | The user was already Sign Up into the system by The Admin or was the Admin and they did the login successfully |
| Trigger: | The User clicks on “ModifyPSW” |
| Priority: | Low |
| Frequency of Use: | Low |
| Normal Course of Events: | The User fills and sends the form |
| Alternative Courses: | The User makes an error and must recompile or the user press “indietro” |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 5 | | |
| Use Case Name: | Order Search | | |
| Created By: | Federico Risoli | Last Updated By: | Elia Candida |
| Date Created: | 10/01/23 | Date Last Updated: | 13/03/23 |

|  |  |
| --- | --- |
| Description: | The User can Search trough the various Orders the one that they want to see |
| Primary Actor: | Admin, Employee |
| Secondary actors: | Database |
| Pre-conditions: | The User is logged in |
| Post-conditions: | Dates must be in a valid format and the time interval must be valid |
| Trigger: | The User goes in the “Ordini” tab |
| Priority: | Medium |
| Frequency of Use: | Medium |
| Normal Course of Events: | The User goes in the “Ordini” tab and can search by date interval the order they want |
| Alternative Courses: | The User goes in the “Ordini” tab and search trough all the orders the one they want by ordering how they prefer using the table.  The User goes in the “Ordini” tab and inserts an invalid date and must do it again in order to get the right order they are searching |
| Includes: | Choose Range Between Dates |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 6 | | |
| Use Case Name: | Complete Order | | |
| Created By: | Federico Risoli | Last Updated By: | Elia Candida |
| Date Created: | 10/01/23 | Date Last Updated: | 13/03/23 |

|  |  |
| --- | --- |
| Description: | The Employee and the Admin can Complete the Orders with Price and date of Delivery |
| Primary Actor: | Employee, Admin |
| Secondary actors: | Server, Database, Supplier, Courier |
| Post-conditions: | The Date and Price selected are valid and not blank |
| Trigger: | The Employee or the Admin makes the Login |
| Priority: | High |
| Frequency of Use: | High |
| Normal Course of Events: | The User Contacts the Supplier and the Courier and establish a price and a delivery date with them, then compiles the form and press “Completa Ordine”, in the end the courier will deliver the wines |
| Alternative Courses: | The User press “Completa Dopo” to skip this order for now and gets a new one.  The User takes to much time to complete the order and receives a new one to complete from the server.  The User completes the form incorrectly and must do it again. |
| Includes: | Select Shipment Date |
| Extends: | Delivery Wines to warehouse |
| Special Requirements: | The Wines must be arrived to us otherwise the User will press “Completa Dopo” |
| Notes and Issues: | The Database and server must have enough RAM and be able to support all the connections otherwise the server will stop working and assigning the new works |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 7 | | |
| Use Case Name: | Employee Management | | |
| Created By: | Federico Risoli | Last Updated By: | Federico Risoli |
| Date Created: | 18/12/22 | Date Last Updated: | 31/12/22 |

|  |  |
| --- | --- |
| Description: | The Admin can Add, Remove and Reset Employees |
| Primary Actor: | Admin |
| Secondary actors: | Database, Employee |
| Pre-conditions: | Admin logged in |
| Post-conditions: | The Forms compiled must be valid and not blank |
| Trigger: | The Admin clicks on “Gestisci Dipendenti” |
| Priority: | High |
| Frequency of Use: | Low |
| Normal Course of Events: | The Admin clicks on “Gestisci Dipendenti”, selects “Registra”, “Cancella” or “Resetta”, fills the form and press the button to complete the action |
| Alternative Courses: | The Admin changes his mind and press “Indietro” to cancel the process. The Admin makes an error compiling the form and must do it again |
| Includes: | Add Employee, Remove Employee, Reset Employee Credentials |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 8 | | |
| Use Case Name: | Buy Wine | | |
| Created By: | Elia Candida | Last Updated By: | Federico Risoli |
| Date Created: | 26/12/22 | Date Last Updated: | 27/02/23 |

|  |  |
| --- | --- |
| Description: | All the Users can choose a wine to buy |
| Primary Actor: | Client |
| Secondary actors: | Database, Server, Employee, Admin |
| Pre-conditions: | User performs the login |
| Post-conditions: | Selected at least one wine. |
| Trigger: | The Client makes the login or the Employee (or Admin) goes into the “Vini” tab after the login |
| Priority: | High |
| Frequency of Use: | High |
| Normal Course of Events: | The User chooses some wines to buy and the quantity, clicks “compra”, chooses a method of payment, clicks “Procedi al Pagamento”, pays and get his wines delivered |
| Alternative Courses: | The User inserts a quantity of wines that is greater than the one in stock, the User then can decide to order them and will pay when they arrive. The User makes an error in the payment and must do it again to buy the Wines.  The User changes his mind and press “Indietro” after selecting the wines |
| Includes: | Wines Selection, Checkout, Choose Payment Method, Order Wines not-in Stock |
| Notes and Issues: | The Order in any case is send directly to the Supplier if it is not in stock to have an easier User Experience |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 9 | | |
| Use Case Name: | Client Search | | |
| Created By: | Federico Risoli | Last Updated By: | Federico Risoli |
| Date Created: | 03/01/23 | Date Last Updated: | 09/03/23 |

|  |  |
| --- | --- |
| Description: | The Employee and the Admin can search for a Client |
| Primary Actor: | Employee, Admin |
| Secondary actors: | Database |
| Pre-conditions: | The User performs the Login |
| Post-conditions: | The form must not be blank |
| Trigger: | The User goes into the tab “Clienti” |
| Priority: | Medium |
| Frequency of Use: | Low |
| Normal Course of Events: | The User goes into the tab “Clienti”, put a Surname and get the desired Client |
| Alternative Courses: | The User inserts a wrong or invalid input and must do it again to get what he wants |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 10 | | |
| Use Case Name: | Log Out | | |
| Created By: | Elia Candida | Last Updated By: | Elia Candida |
| Date Created: | 30/01/23 | Date Last Updated: | 09/03/23 |

|  |  |
| --- | --- |
| Description: | The User can perform the Logout without losing its data |
| Primary Actor: | All the Users |
| Secondary actors: | Server |
| Pre-conditions: | The User performed the Login |
| Trigger: | The User clicks on “logout” |
| Priority: | High |
| Frequency of Use: | High |
| Normal Course of Events: | The User clicks on “Logout” and its data are saved, the server takes note if the user was doing something important and make a secure logout |
| Alternative Courses: | The User press ALT+F4 to logout, in any case the Server will handle that |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 11 | | |
| Use Case Name: | Wine Search | | |
| Created By: | Elia Candida | Last Updated By: | Federico Risoli |
| Date Created: | 23/12/22 | Date Last Updated: | 27/01/23 |

|  |  |
| --- | --- |
| Description: | The Users can search for a specific wine by date and/or name |
| Primary Actor: | All the Users |
| Secondary actors: | Database |
| Pre-conditions: | The User must have logged in |
| Post-conditions: | The Wine exists in the Database |
| Trigger: | The User clicks on “seach” |
| Priority: | Medium |
| Frequency of Use: | High |
| Normal Course of Events: | The User inserts a name of a wines and/or selects a date and then clicks “Search” |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 12 | | |
| Use Case Name: | Check Notifications | | |
| Created By: | Federico Risoli | Last Updated By: | Elia Candida |
| Date Created: | 20/02/23 | Date Last Updated: | 03/03/23 |

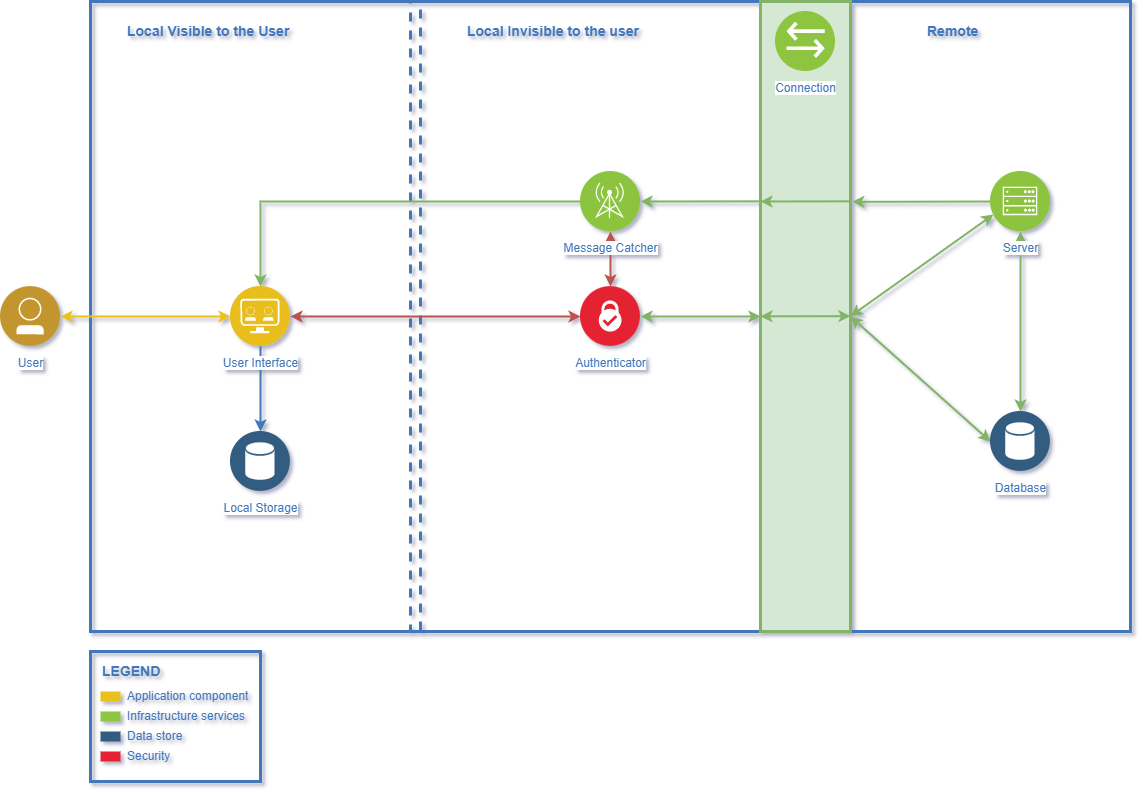
|  |  |
| --- | --- |
| Description: | The User see if there are some notification for him and can take action accordingly |
| Primary Actor: | All the Users |
| Secondary actors: | Database |
| Pre-conditions: | The User logged in |
| Trigger: | The User Clicks on “Notifiche” |
| Priority: | Medium |
| Frequency of Use: | Medium |
| Normal Course of Events: | The User Clicks on “Notifiche” and if there are things to do can choose a payment method to get his wine or reject it or do nothing |
| Includes: | Checkout, Choose payment method |
| Special Requirements: | This will appear only if the Employee has marked the order as ready |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 13 | | |
| Use Case Name: | Orders History | | |
| Created By: | Federico Risoli | Last Updated By: | Federico Risoli |
| Date Created: | 23/01/23 | Date Last Updated: | 27/02/23 |

|  |  |
| --- | --- |
| Description: | The User can check his Order History |
| Primary Actor: | All the Users |
| Secondary actors: | Database |
| Pre-conditions: | The User is logged in |
| Trigger: | The Client clicks on “I miei ordini” or the Employee/Admin search through all the Wines Historyes |
| Priority: | Medium |
| Frequency of Use: | Low |
| Normal Course of Events: | The Client clicks on “I miei ordini” and sees his orders |
| Alternative Courses: | Employee/Admin clicks on the “Ordini” tab and search through all the Wines Historyes |

# **Architectural Model**

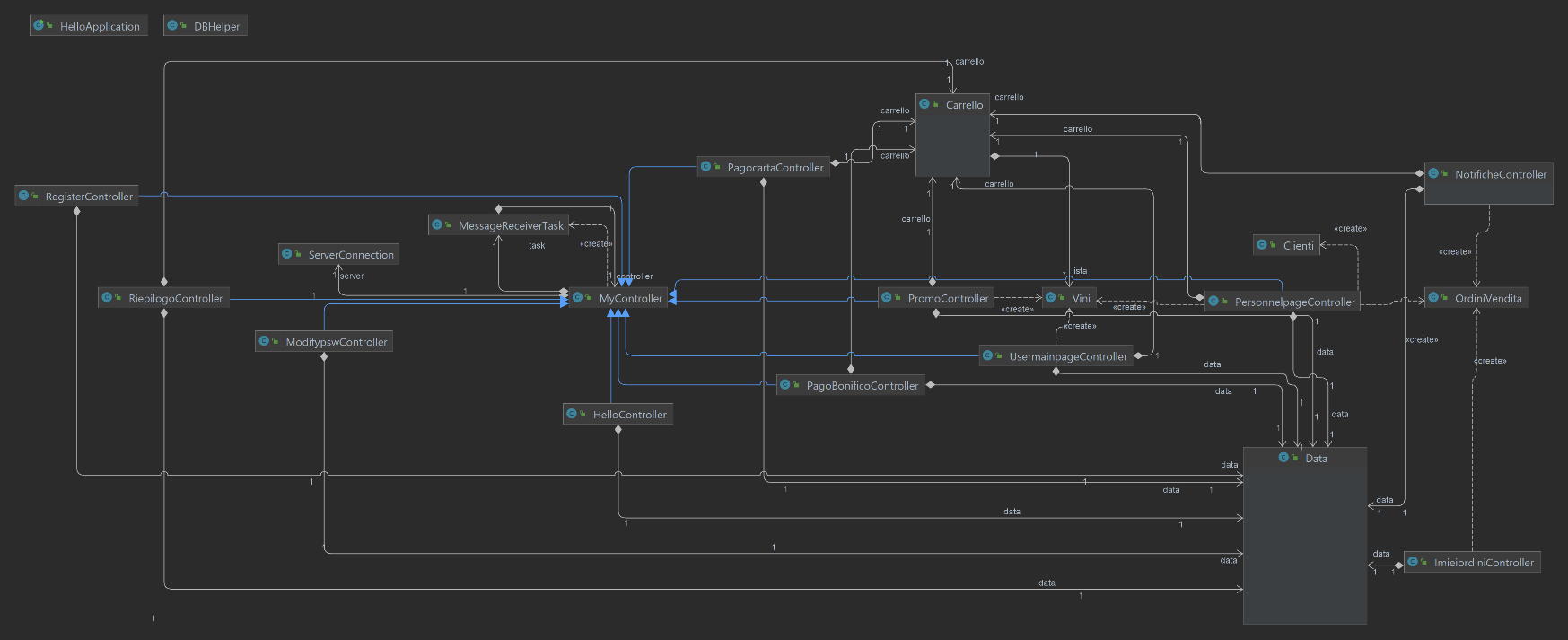
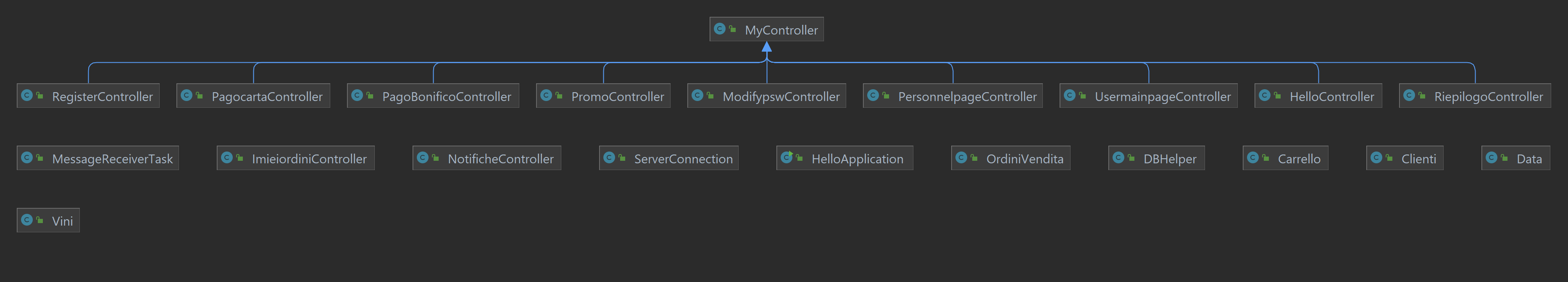
**Architecture diagram**

****

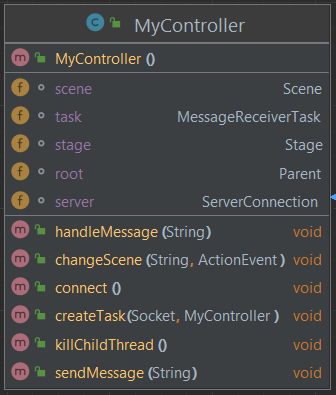
# **Detailed Software Model**

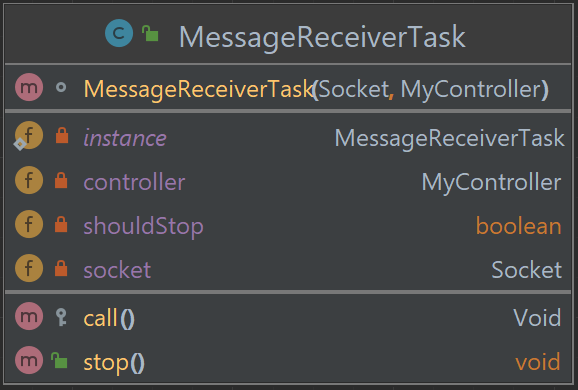
**Class diagram(s)**

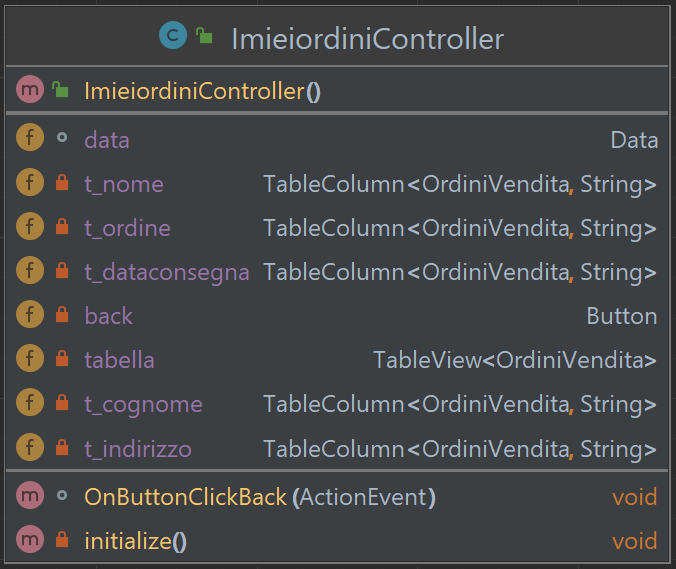
The Full Class Diagram of the main application would be preatty hard to read and so it was decided to split it into multiple views each one hilighting something different.

* Class Diagram Without Methods, Constructors and variables (those can be seen later):
* Hilight of only the Inheritances:
* Main app Diagram with only Inheritances, Methods, Variables, Constructors :
* Zoom on the single classes:

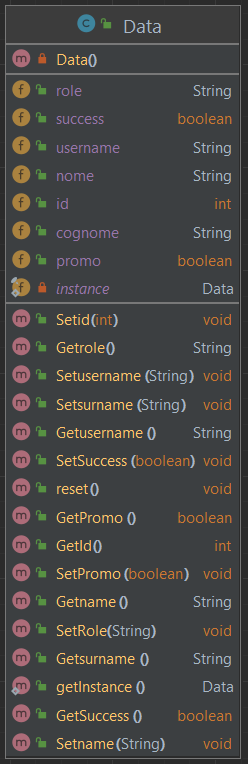
These was extracted directly from the code and there might be a couple of words in Italian.

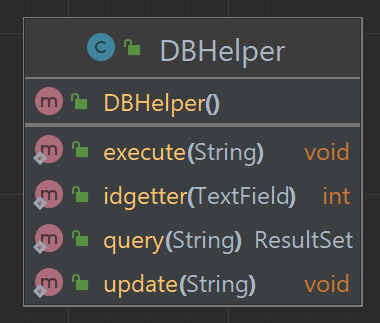
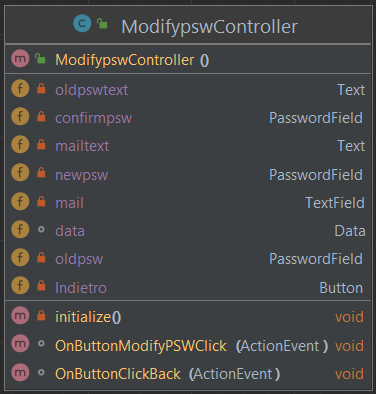
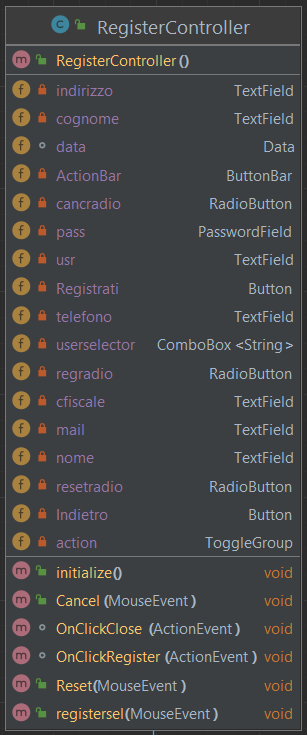
****

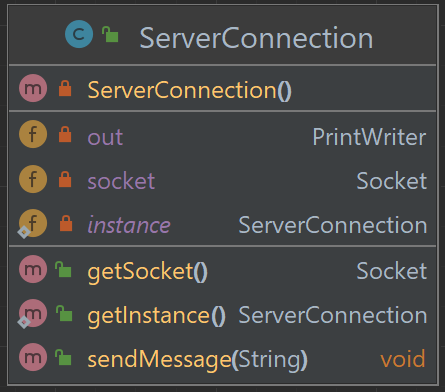
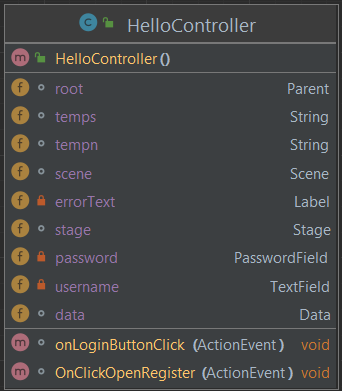
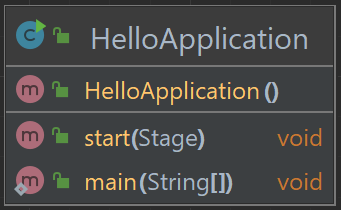
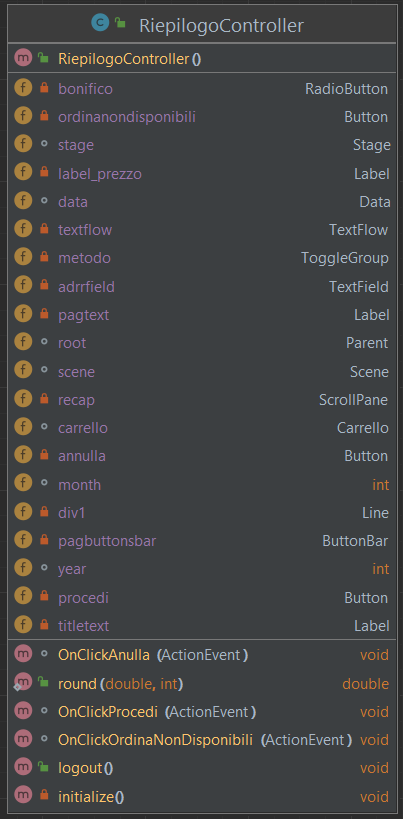
****

****

**Immagine che contiene testo

Descrizione generata automaticamente**

****

****

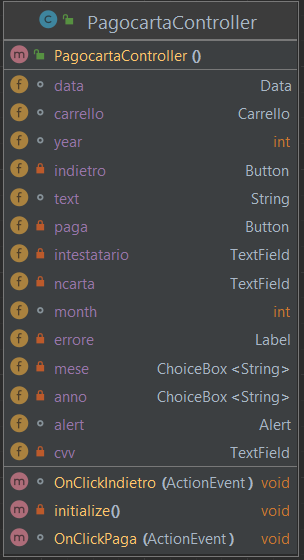
**Immagine che contiene testo

Descrizione generata automaticamente**

**Immagine che contiene testo, schermata, telefono, telefono cellulare

Descrizione generata automaticamente**

**Immagine che contiene testo

Descrizione generata automaticamenteImmagine che contiene testo

Descrizione generata automaticamente**

**Immagine che contiene testo

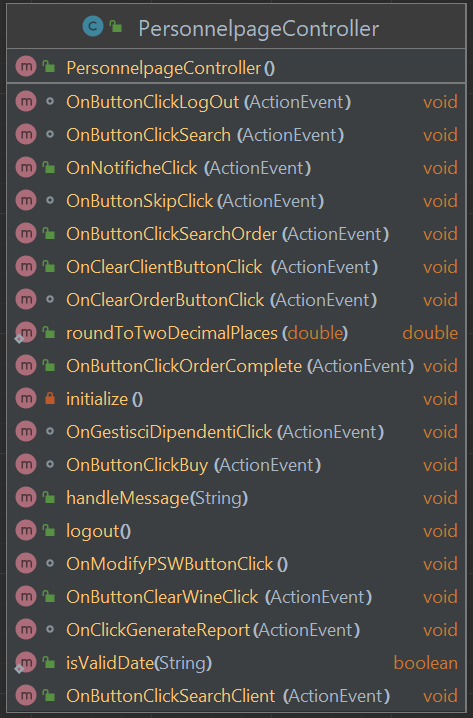
Descrizione generata automaticamente**

**Immagine che contiene testo

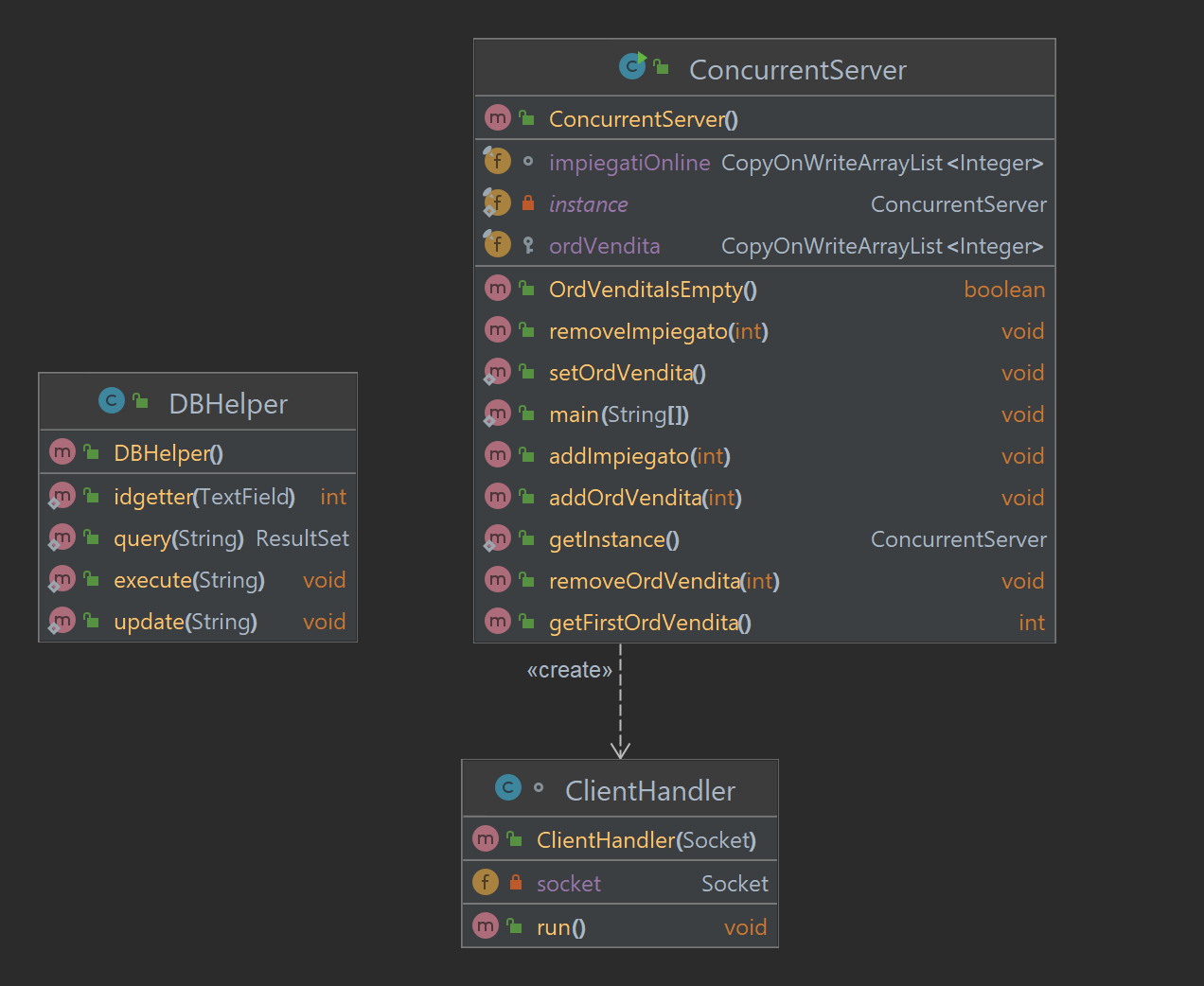
Descrizione generata automaticamente**

**Immagine che contiene testo

Descrizione generata automaticamente**

Immagine che contiene testo

Descrizione generata automaticamentePersonnelpageController was the biggest controller of all and it was split in 2 screenshots for better readability.

* Class Diagram Concurrent Server:

# **Functional Tests**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test name** | **Steps** | **Expected result** | **Test result** | **Comments** |
| 1. | Unit Test Carrello.java | 1. Setup pre test 2. Testo i metodi set/get/clear/non-disponibile della classe | pass | pass |  |
| 2. | Unit Test Data.java | 1. Testo tutte le funzioni per vedere se funzionano correttamente | pass | pass |  |
| 3. | Unit Test PersonnelPageController.java | 1. Testo nei casi limite le funzioni roundtodecimalplaces e isValidDate | Pass | pass |  |
| 4 | Unit Test ConcurrentServer.java | 1. Setup con clear delle variabili che rimangono da un test all’altro. 2. Testo le funzioni che Aggiungono/rimuovono/ottengono gli impiegati e gli ordini di vendita registrati nel server | Pass | pass |  |
| 5 | Test UI | 1. Entrare in una pagina 2. Testare il corretto funzionamento di tutti i suoi elementi 3. Ripetere per ogni Controller | Pass | Pass |  |
| 6 | Test Generale Sviluppatori | 1. Testare tutto il sistema in maniera insolita per evidenziarne i bug | Pass | Pass | Eseguito da Elia e Federico non sono stati trovati bug malgrado ci fossero; quindi il test è passato malgrado bug nascosti (i test non assicurano l’assenza di bug) |
| 7 | Alpha Test 1 | 1. Testare tutto il sistema in maniera insolita per evidenziarne i bug | Pass | Fail | Eseguito da un amico degli sviluppatori per avere visioni differenti sul programma. Ha evidenziato un bug quindi il test conta come fallito |
| 8 | Alpha Test 2 | 1. Testare tutto il sistema in maniera insolita per evidenziarne i bug | Pass | Pass | Eseguito dopo l’Alpha Test 1 e dopo il fix del bug trovato. |

# **System Manual**

**Installation**

* Installare XAMPP sul proprio sistema da <https://www.apachefriends.org/it/index.html>
* Installare Java 19.0.2 sul proprio sistema da <https://www.oracle.com/java/technologies/javase/jdk19-archive-downloads.html>
* Andare sulla pagina GitHub del progetto e scaricare l’ultima Release <https://github.com/FedericoRisoli/Assegnamento>
* Estrarre il file .zip dove si preferisce sul proprio sistema, questa sarà la cartella dell’applicazione
* Nella cartella d’istallazione aprire la cartella javafx-sdk-19.0.2.1 e copiare il percorso della cartella “lib”
* Modificare il file RunClient.bat sostituendo [PERCORSO CARTELLA LIB] con quello appena copiato. Il risultato dovrebbe essere simile a questo:

#!/bin/bash

java --module-path C:\Users\Elia\IdeaProjects\AssegnamentoWineShop\out\artifacts\Client\javafx-sdk-19.0.2.1\lib --add-modules javafx.controls,javafx.fxml -jar Client.jar

* Aprire XAMPP, attivare Apache e MySQL, premere sul pulsante Admin di MySQL
* Creare un nuovo Database nella pagina appena aperta nel Browser
* Prendere il file wineshop.sql dalla cartella dell’applicazione e trascinarlo nel browser, questo farà l’upload del Database
* (opzionale) cancellare il file wineshop.sql dalla cartella dell’applicazione

**Configuration**

Il sistema è già pronto per essere usato, tuttavia ci sono dei passaggi opzionali che sono consigliati:

* Aprire XAMPP, attivare Apache e MySQL, premere sul pulsante Admin di MySQL per sostituire il database di esempio con quello che effettivamente verrà utilizzato
* Se non si vuole eseguire il passaggio qui sopra si consiglia almeno di: aprire XAMPP, attivare Apache e MySQL. Aprire RunServer.bat e successivamente RunClient.bat. Eseguire l’accesso con le credenziali dell’amministratore (usr: admin, psw: admin) e modificare la password dell’amministratore di sistema.

Per far partire il sistema bisogna comunque aprire XAMPP, attivare Apache e MySQL e aprire RunServer.bat

**User Guide**

**Indice**

**Clienti**

1. Login
2. Comprare/Ordinare/Ricerca Vini
3. Visualizzare I Propri Ordini
4. Notifiche-Ordini Pronti

**Impiegati**

1. Login
2. Lavoro
3. Comprare/Ordinare/Ricerca Vini
4. Notifiche-Ordini Pronti
5. Ricerca Ordini
6. Ricerca Cliente
7. Modifica Password

**Amministratore**

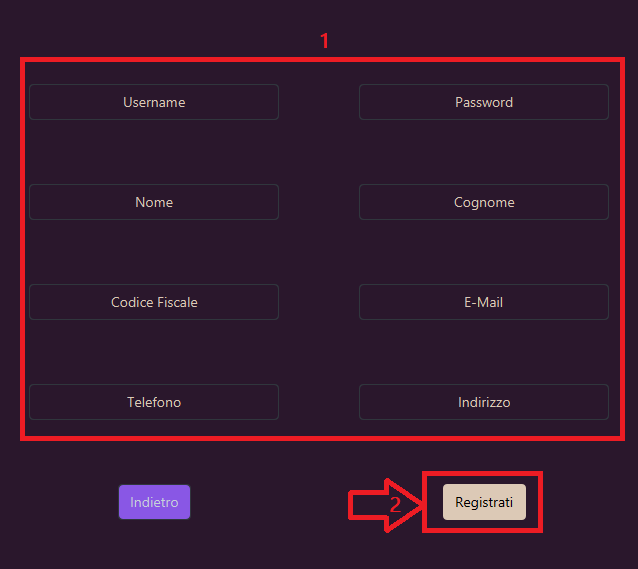
*Per funzionalita’ base vedere manuale dipendente*

*Per funzionalita’ specifiche admin*

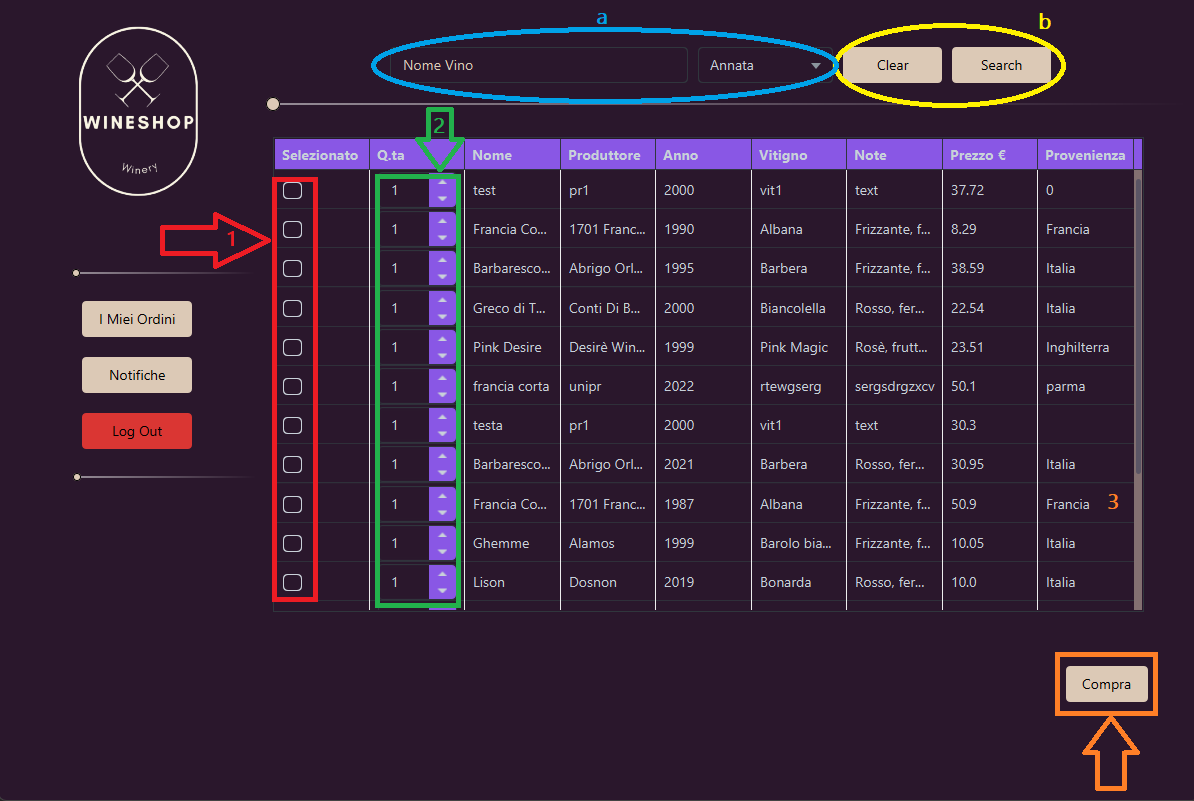
1. Generare Report Mensile
2. Gestione Dipendenti
   1. Registrare un dipendente
   2. Cancellare un dipendente
   3. Resettare Credenziali
3. Modifica Password

**Clienti**

**1-Login**

* Utenti Già Registrati
  1. Aprire RunClient.bat
  2. Inserire username e password
  3. Cliccare su Login
* Utenti non Registrati
  1. Aprire RunClient.bat
  2. Cliccare su “Registrati Cliente”
  3. Compilare tutti i campi
  4. Cliccare su “Registrati”

**2-Comprare/Ordinare/Ricercare Vini**

1. Selezionare i vini che si desiderano comprare con l’apposita checkbox sul lato sinistro della tabella
   1. Si possono ricercare I vini compilando gli apposite campi ”Nome Prodotto” o selezionare l’annata
   2. Per Resettare la ricerca cliccare su “Clear”
2. Inserire la quantità per ogni prodotto selezionato
3. Cliccare il tasto “Compra”
4. Selezionare il metodo di pagamento e cliccare su “procedi al pagamento”
5. Compilare i campi e cliccare su “Paga”

* **NB se i vini selezionati non sono disponibili**
  + - 1. Inserire l’indirizzo di consegna nell’ apposito campo
      2. Cliccare su “Ordina Vini Non Disponibili”

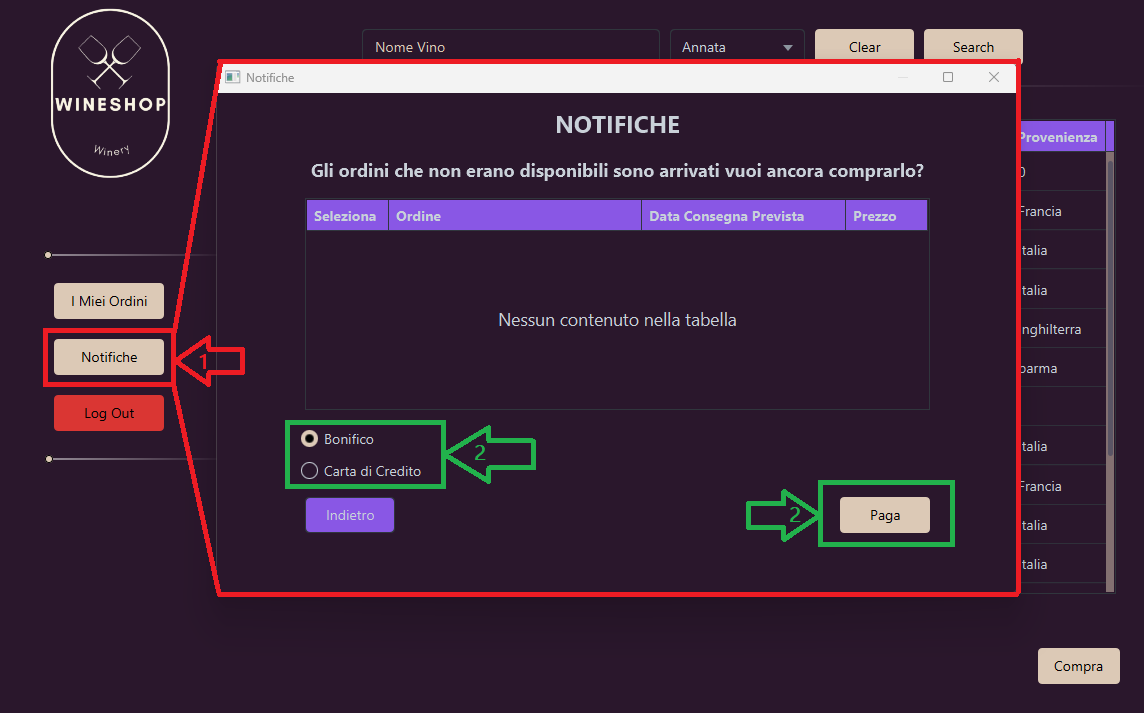


**3-Visualizzare i propri ordini**

1. Immagine che contiene testo, monitor, schermata, nero

   Descrizione generata automaticamenteNella pagina principale cliccare su “I miei Ordini”

**4-Notifiche-Ordini Pronti**

* 1. Cliccare su Notifiche
  2. Selezionare metodo di pagamento se si vuole acquistare e cliccare su “Paga”
  3. Compilare i campi richiesti dal metodo di pagamento e cliccare su “Paga”

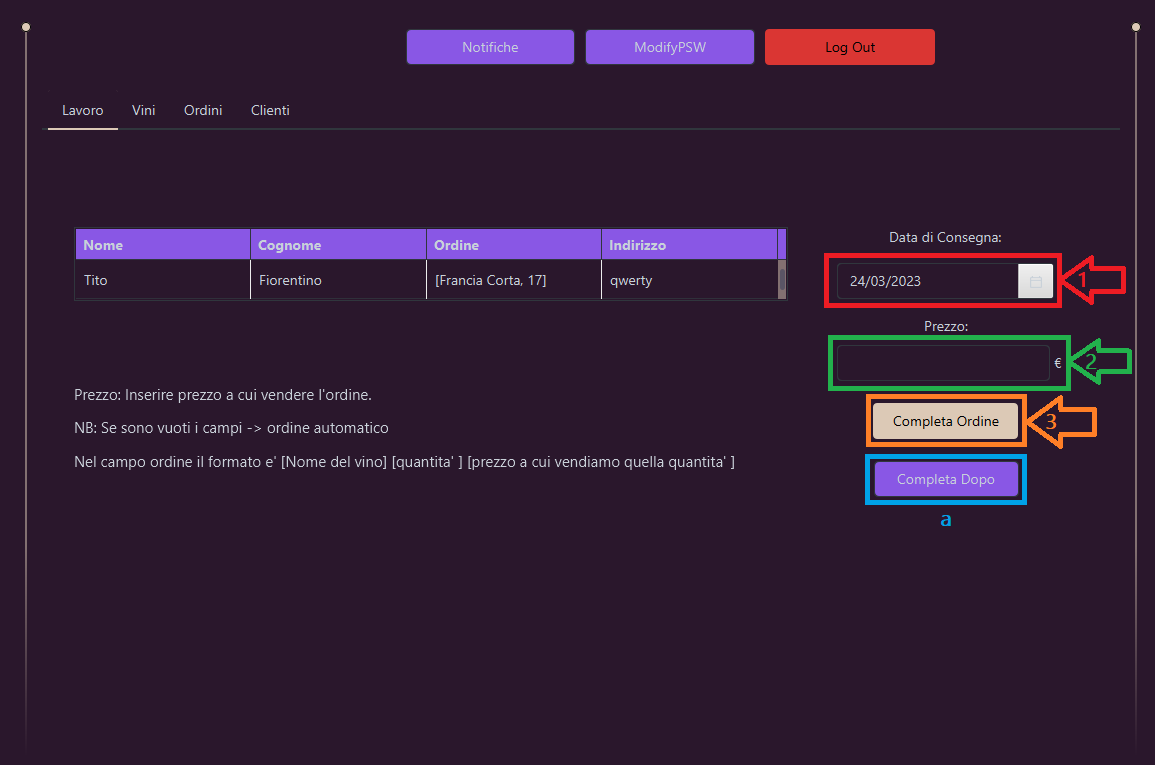
**Impiegati**

**1-Login**

1. Aprire RunClient.bat
2. Inserire username e password
3. Cliccare su Login

**2-Lavoro**

Per Completare il Lavoro (merce arrivata)

1. Selezionare data di consegna
2. Inserire Prezzo a cui vendere il vino
3. Cliccare su ”Completa Ordine”
   1. Se la merce non e’ arrivata cliccare su “Completa Dopo”

**3-Comprare/Ordinare/Ricercare Vini**

1.Selezionare i vini che si desiderano comprare con

l’apposita checkbox sul lato sinistro della tabella

* 1. Si possono ricercare I vini compilando gli apposite campi ”Nome Prodotto” o selezionare l’annata
  2. Cliccare su “Search”
  3. Per Resettare la ricerca cliccare su “Clear”

2.Inserire la quantità per ogni prodotto selezionato

3.Cliccare il tasto “Compra”

Immagine che contiene testo, monitor, nero, schermo/paravento

Descrizione generata automaticamente

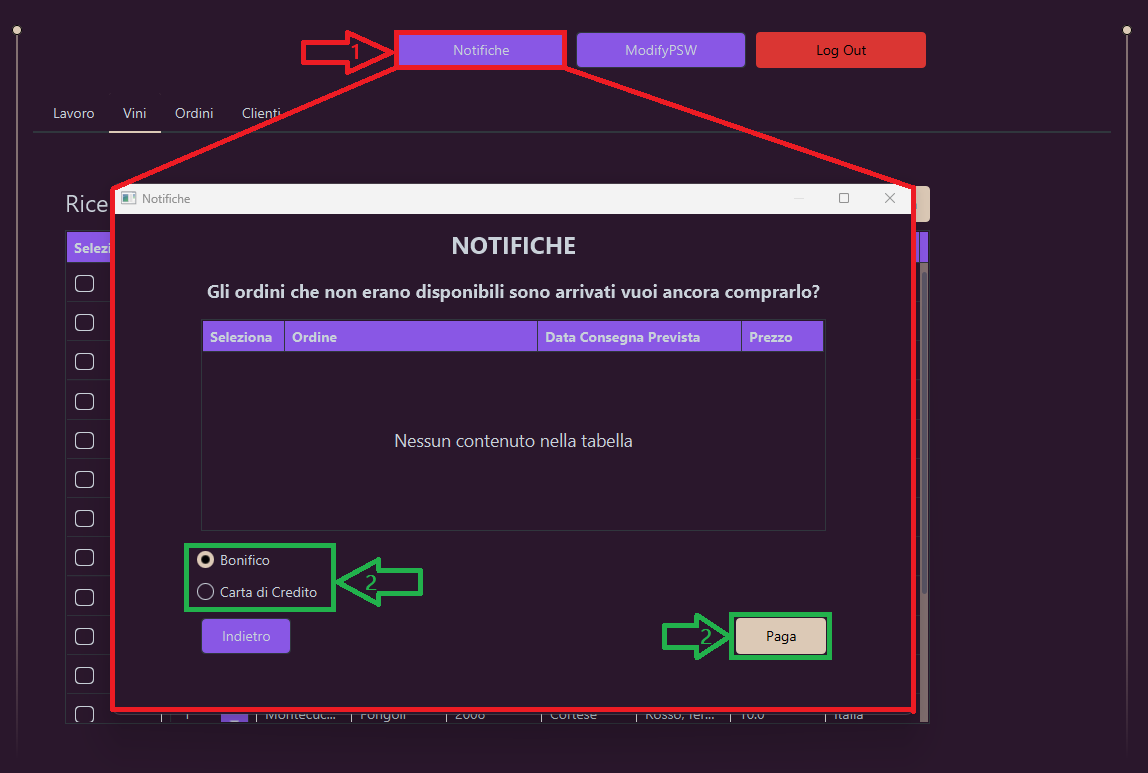
Immagine che contiene testo

Descrizione generata automaticamente4.Selezionare il metodo di pagamento e cliccare su “procedi al pagamento”

 5.Compilare i campi e cliccare su “Paga”

* **NB se i vini selezionati non sono disponibili**
  + - 1. Inserire l’indirizzo di consegna nell’ apposito campo
      2. Cliccare su “Ordina Vini Non Disponibili”

**4-Notifiche-Ordini Pronti**

1.Cliccare su Notifiche

2.Selezionare metodo di pagamento se si vuole acquistare e cliccare su “Paga”

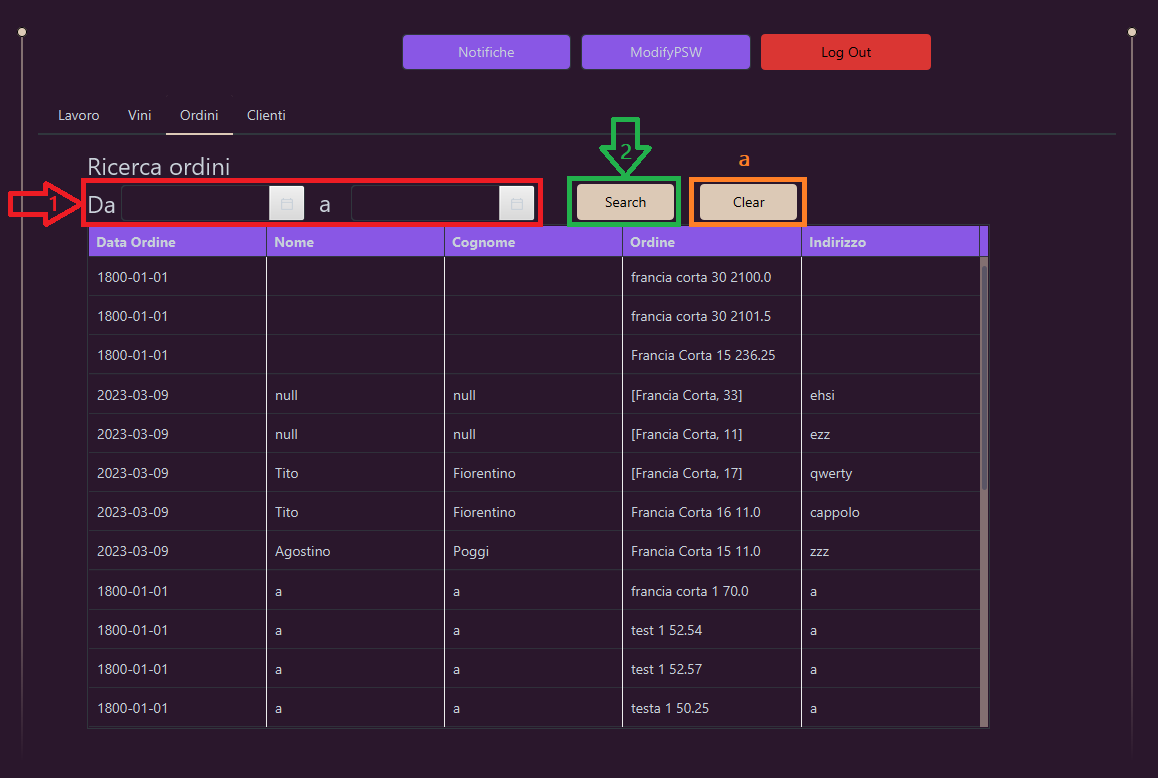
3.Compilare i campi richiesti dal metodo di pagamento e cliccare su “Paga”

**5-Ricerca Ordini**

1.Inserire Data iniziale e data finale negli appositi campi per definire l’intervallo di ricerca

2.Cliccare su “Search”

a. Cliccare su “Clear” per resettare la ricerca

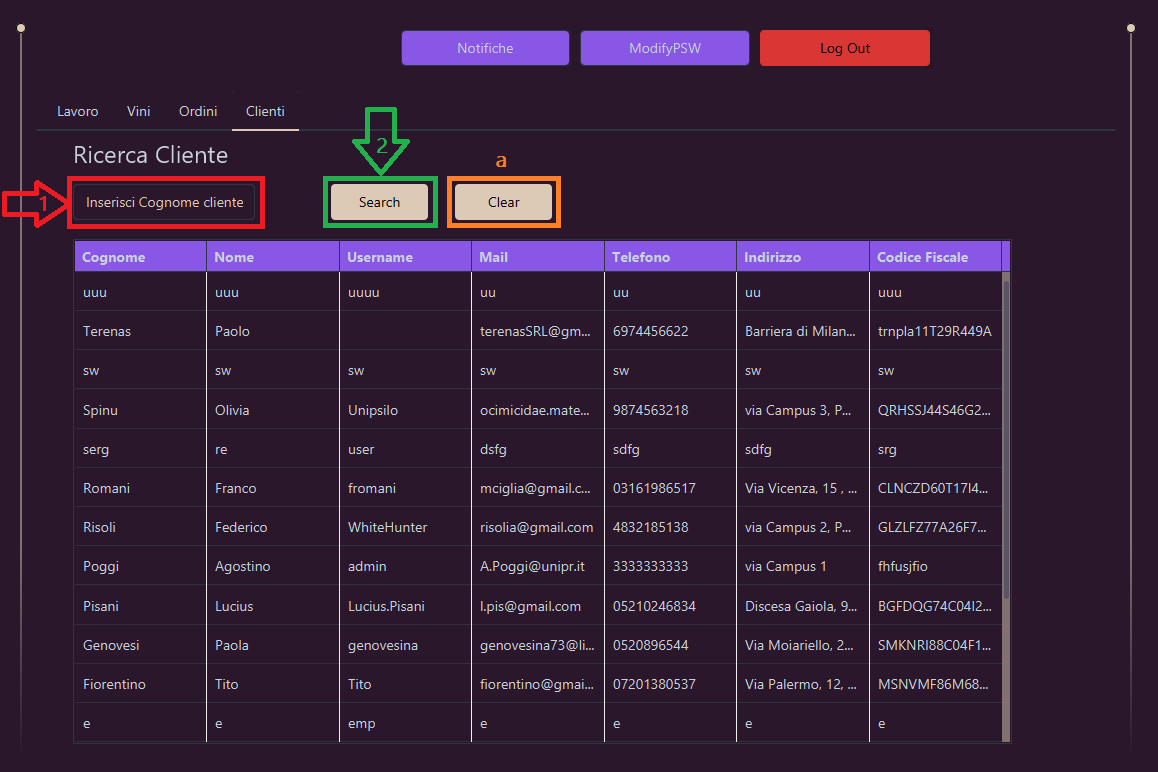


**6-Ricerca Clienti**

1.Inserire Cognome nell’apposito campo

2.Cliccare su “Search”

a. Cliccare su “Clear” per resettare la ricerca



**7-Modifica Password**

1.Cliccare su “Modify PSW”

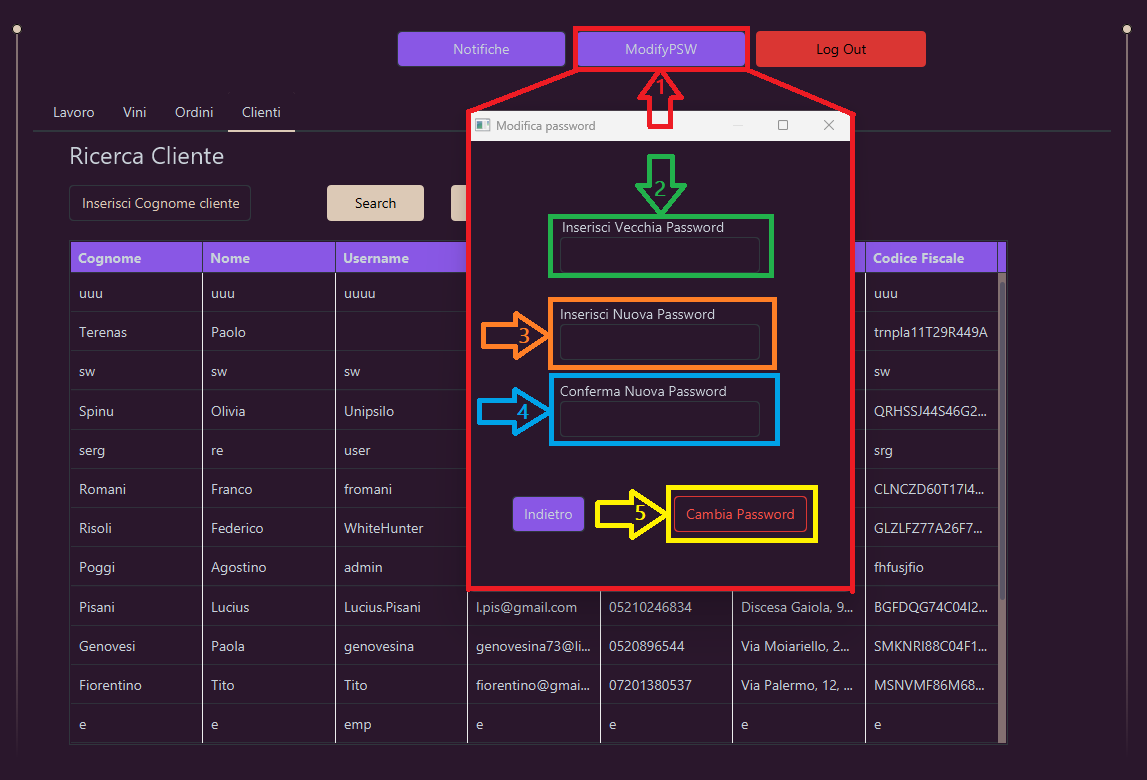
2.Inserire Password attuale

3.Inserire nuova Password

4.Confermare Nuova Password

5.Cliccare su “Cambia Password”

**NB. Se non si ricorda la password contattare l’admin per il Reset delle credenziali**



**Amministratore**

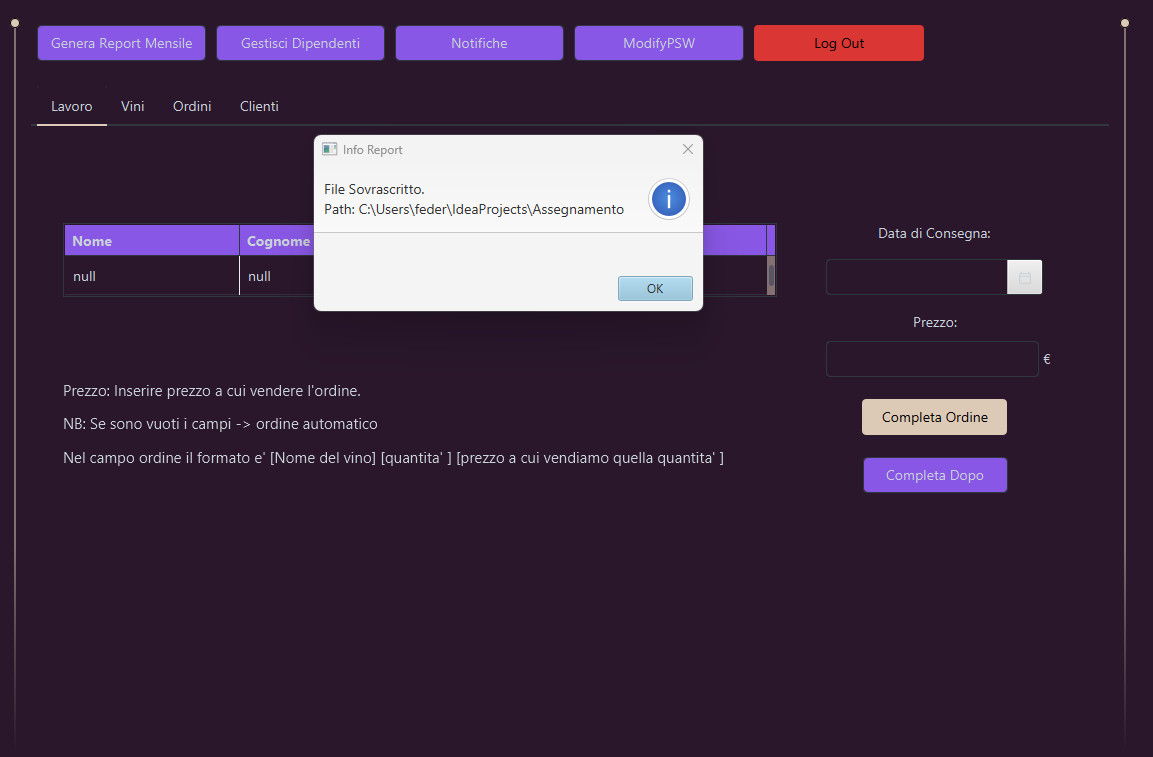
L’ amministratore è considerato come un impiegato particolare, per le funzioni base Vedere manuale utente per impiegati, qui sotto verrà mostrato il manuale delle funzionalità specifiche dell’Admin.

**1-Generare Report Mensile**

**Immagine che contiene testo, schermata, monitor, nero

Descrizione generata automaticamente**1.Cliccare su “Genera report Mensile”

Il report si trova nel percorso specificato dal popup

****

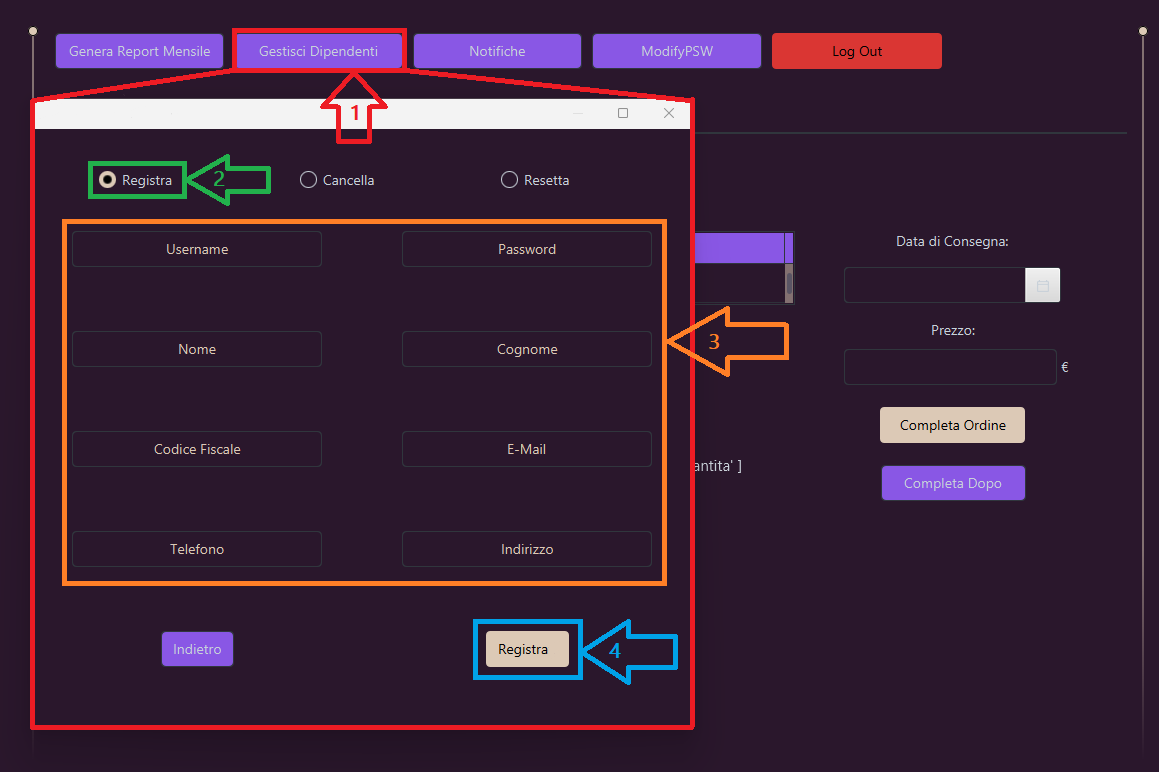
**2-Gestione dipendenti**

**Registrazione dipendente**

1.Cliccare su “Gestione Dipendenti”

2.Selezionare il radio “Registra”

3.Compilare i campi richiesti

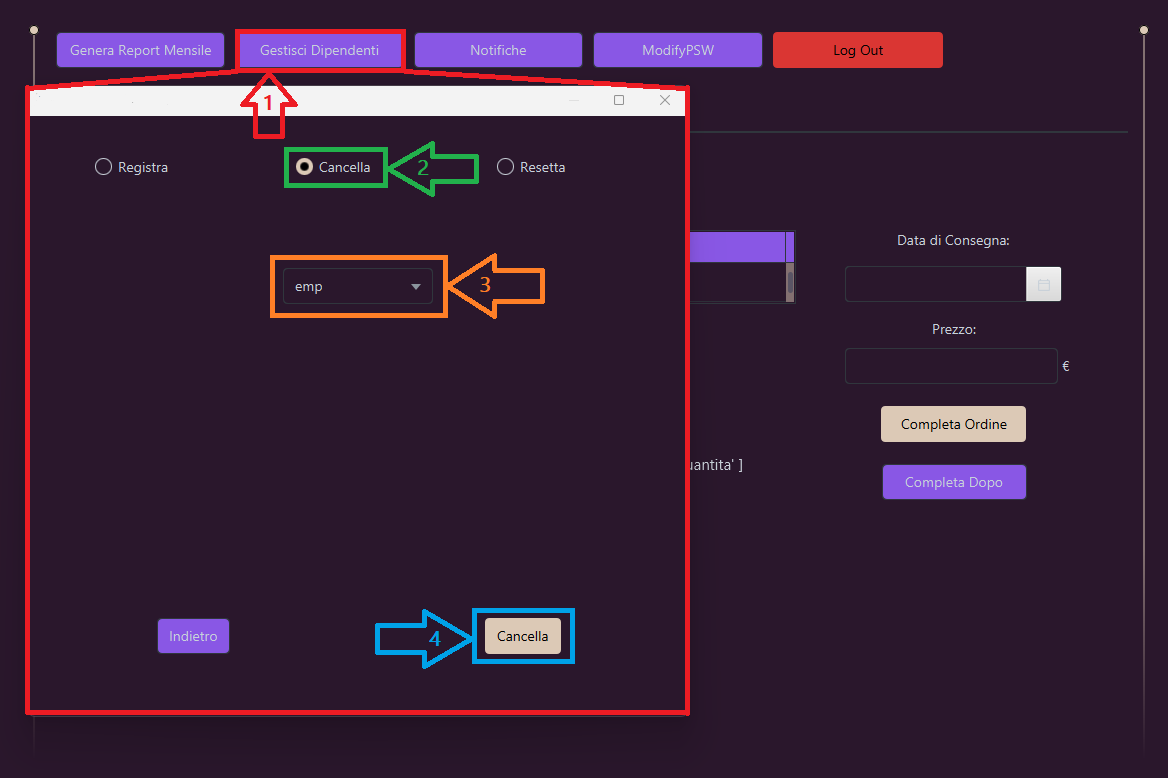
4.Cliccare su “Registra”

**Cancellazione dipendente**

1.Cliccare su “Gestione Dipendenti”

2.Selezionare il radio “Cancella”

3.Selezionare il dipendente da eliminare

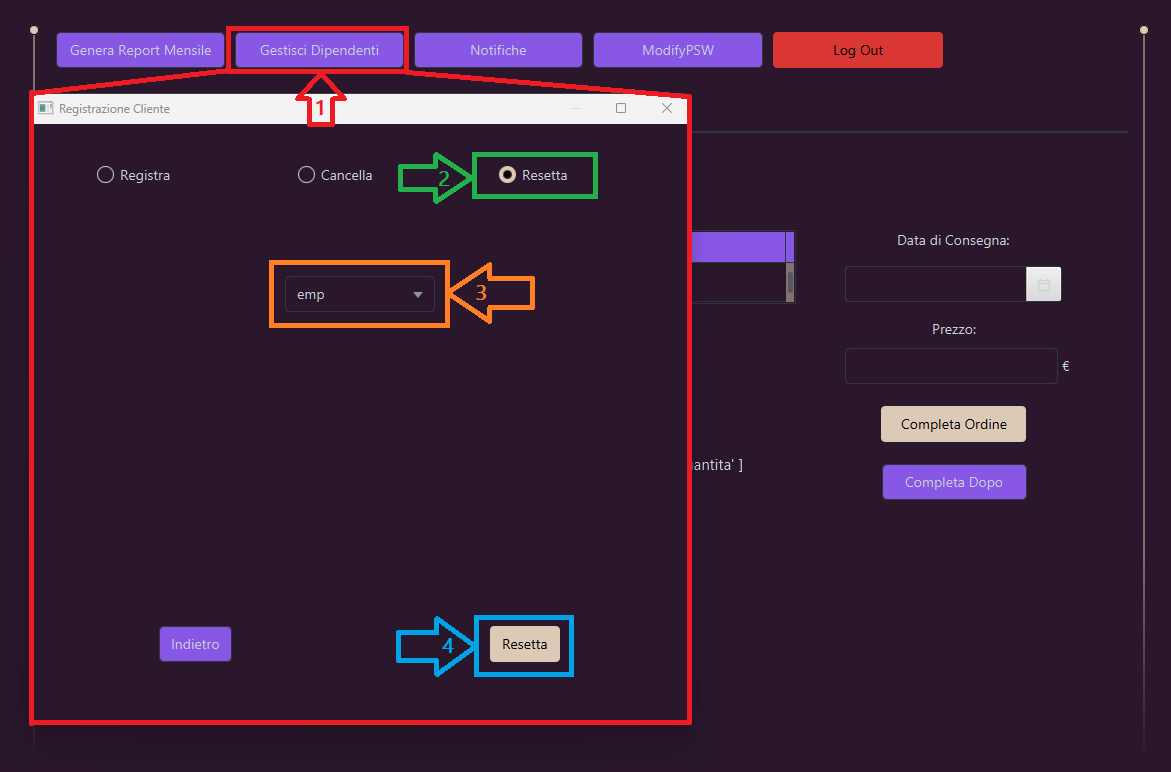
****4.Cliccare su “Cancella”

**Reset Credenziali dipendente**

1.Cliccare su “Gestione Dipendenti”

2.Selezionare il radio “Resetta”

3.Selezionare il dipendente a cui resettare le credenziali

****4.Cliccare su “Resetta”

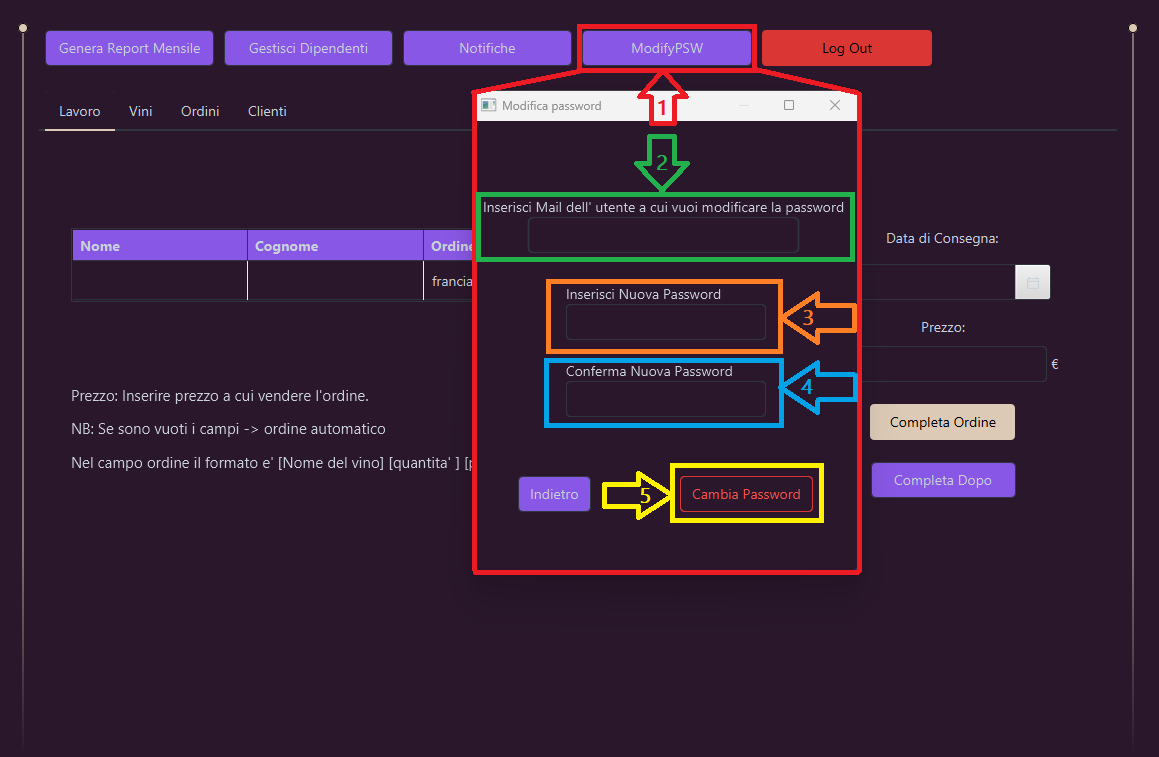
**3-Modifica password**

1.Cliccare su “ModifyPSW”

2.Inserire la mail dell’utente a cui si vuole cambiare password

3.Inserire Nuova password

4.Confermare la password

5. Cliccare su “Cambia password”

**Changelog**

In questa sezione mostreremo i cambiamenti apportati al programma con tanto di data e nome dell’autore in ordine dal più nuovo al più vecchio. Per maggiori informazioni visitare la pagina Github <https://github.com/FedericoRisoli/Assegnamento>

Elia Candida alias Elia, elia2904

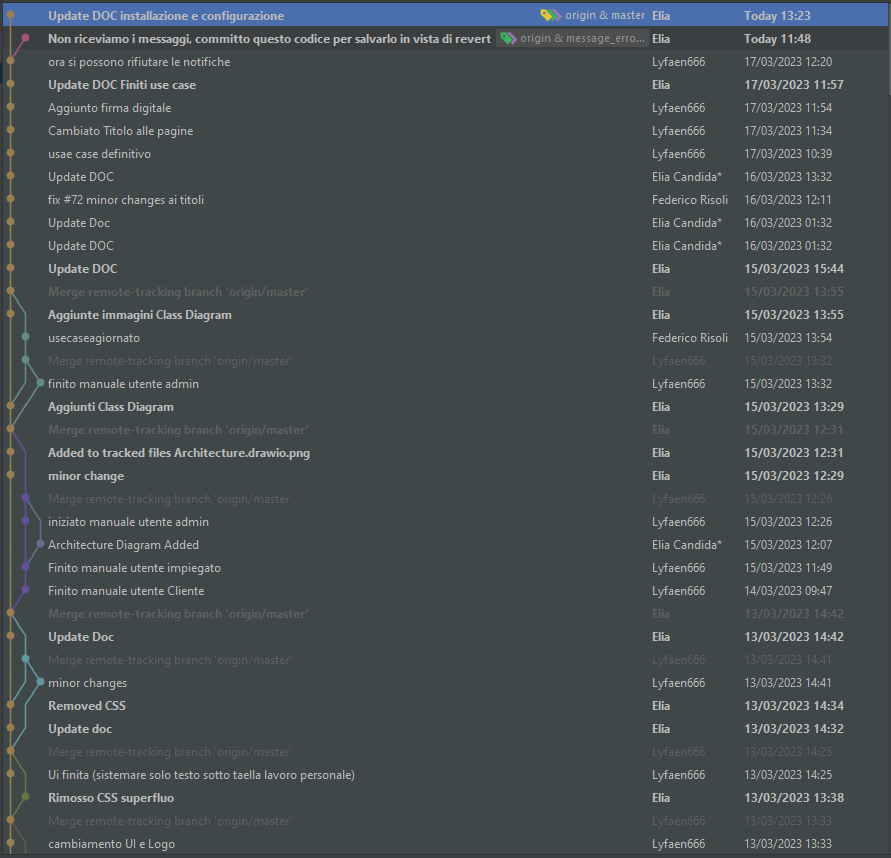
Federico Risoli alias Fede, Lyfaen666

Immagine che contiene testo, schermata, targa

Descrizione generata automaticamente

Immagine che contiene testo

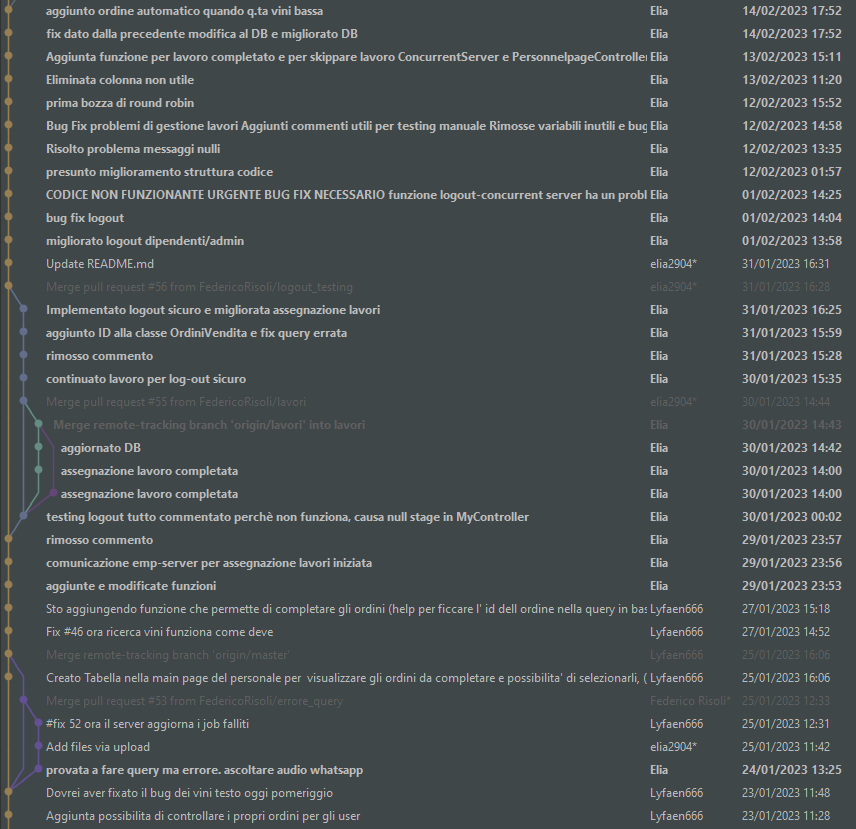
Descrizione generata automaticamente

Immagine che contiene testo, targa, schermata

Descrizione generata automaticamente

