

Progetto Ing. del
Software:

Worklink

Andrea Moressa 1074124
Loris Iacoban 1074130
Amin Borqal 1073928



Cosa è Worklink?

Worklink offre un'esperienza intuitiva e facile da usare per gli utenti che cercano lavoro, consentendo loro di trovare rapidamente le offerte più pertinenti e candidarsi tramite la piattaforma.

Nel frattempo, le aziende possono utilizzare Worklink per pubblicare le loro offerte di lavoro, esaminare i candidati e selezionare il candidato ideale.

Obiettivi e scopi

Semplificare la ricerca di lavoro per il
singolo individuo e facilitare la
selezione dei candidati per le aziende

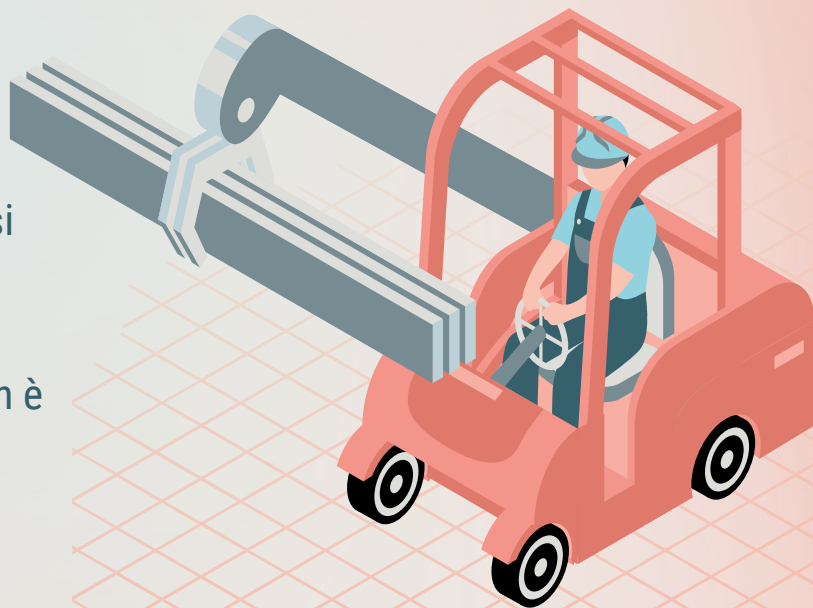
Difficoltà incontrate - prog.

Programmazione :

- Interazione tra gli oggetti e il database
- Implementazione delle varie funzioni

Le varie difficoltà di programmazione incontrate si potevano risolvere con una conoscenza approfondita dei linguaggi.

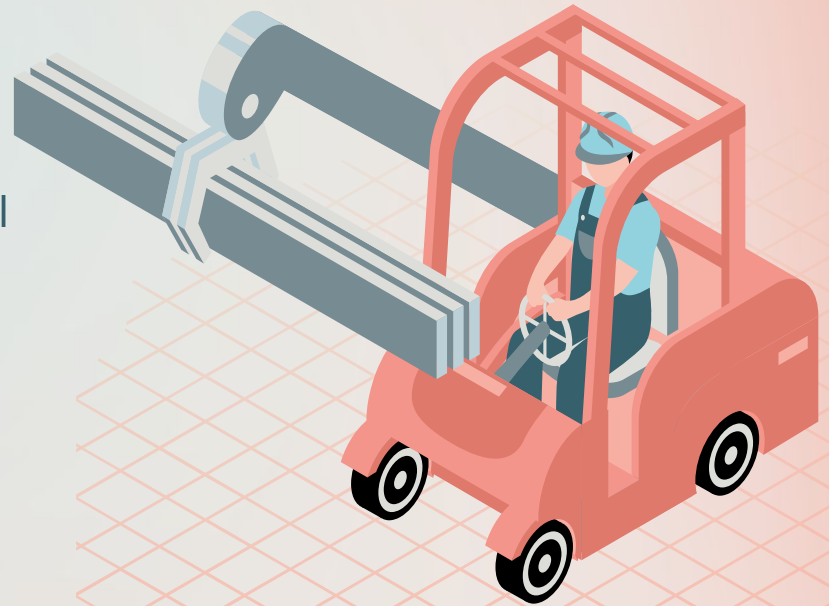
Avendo nel nostro caso una conoscenza base non è stato immediato superare l'ostacolo.



Difficoltà incontrate - tempo

Rispetto delle tempistiche:

Causa problemi personali di ogni componente del gruppo ci è venuto meno il rispetto dei tempi



Contenuti

01	02	03	04
Paradigmi di programmazione e modellazione	Software Configuration management	Software life cycle	Requisiti

05	06	07	08	09
Architettura e design	Modellazione	Implementazione	DEMO	Testing





Paradigmi di programmazione e modellazione

Paradigmi di programmazione e modellazione, linguaggio di programmazione e tool utilizzati



Paradigmi di prog./modellazione



Programmazione

- Programmazione ad oggetti (OOP) in python

Modellazione

- UML

Linguaggi di prog. e tools utilizzati



Linguaggio

- Python
- Flask framework
- HTML,CSS
- SQLITE

Tools

- Visual Studio Code
- GitHub
- SQLITEONLINE



Software configuration management

GitHub



GitHub



Per assicurare che il nostro sistema sia sempre funzionante e che il lavoro svolto sia tracciato in modo efficace ,abbiamo utilizzato GitHub come SCM tool.

Grazie alla metodologia **Agile**, abbiamo organizzato il nostro lavoro in sprint, e GitHub ci ha permesso di tenere traccia di tutte le modifiche svolte, garantendo che il nostro processo di sviluppo sia stato trasparente e tracciabile.

In questo modo, il nostro team è stato in grado di collaborare in modo efficiente e coordinato, garantendo sempre il massimo livello di qualità nel prodotto finale



Software life cycle

Obiettivi,
organizzazione del team
e divisione del lavoro



Obiettivi da raggiungere

Scelta del progetto

MILESTONE 1



MILESTONE 2

Pianificazione

Esecuzione

MILESTONE 3



MILESTONE 4

Rilascio del progetto

Chiusura del progetto

MILESTONE 5



Organizzazione del Team

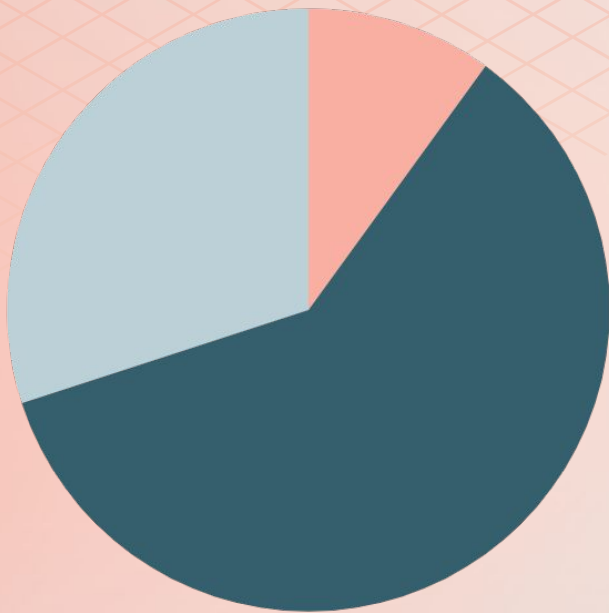
Il progetto sarà gestito da un team coeso con un approccio **SCRUM** con enfasi sulla relazione tra i membri.

Tutti i membri avranno abilità simili e svolgeranno ruoli di **Scrum Master**, **Product Owner** e **Development Team**.

Gli **sprint** dureranno tra 2-4 settimane con **Daily Scrum** di 15-20 minuti e incontri settimanali per discutere problemi e raccogliere idee.



Divisione del lavoro



10%

Chiusura

Consegna del progetto e chiusura

60%

Pianificazione

Scelta del progetto e decisioni strutturali

30%

Esecuzione

Scrittura del codice e dei test



Requisiti

Requisiti, requisiti
funzionali, requisiti non
funzionali e sulle
performance



Requisiti

Interfaccia utente

Il sistema software deve possedere un'interfaccia semplice, intuitiva e chiara per poter essere utilizzata da più persone possibili

Interfaccia Hardware

Il sistema software **non** necessita nessun sistema hardware

Requisiti Funzionali

Autenticazione utente/azienda

Login:

L'accesso al proprio account è realizzato inserendo l'apposita e-mail e password fornite durante la fase di registrazione (per ogni e-mail deve esistere un unico account)

Area personale utente

Cerca offerte per professione:

mostra le offerte relative alla mansione richiesto dato in input

Cerca offerte per salario:

mostra le offerte relative allo stipendio richiesto da input

Area personale Azienda

Carica offerta:

permette di pubblicare un'offerta di lavoro

Vedi candidatura:

permette di visualizzare l'elenco dei candidati ai vari annunci

Requisiti non funzionali/performance

Requisiti non funzionali

Effettua un reclamo:
Crea un reclamo

Requisiti sulle performance

Essendo una webapp relativamente semplice, **non** necessita di performance elevate



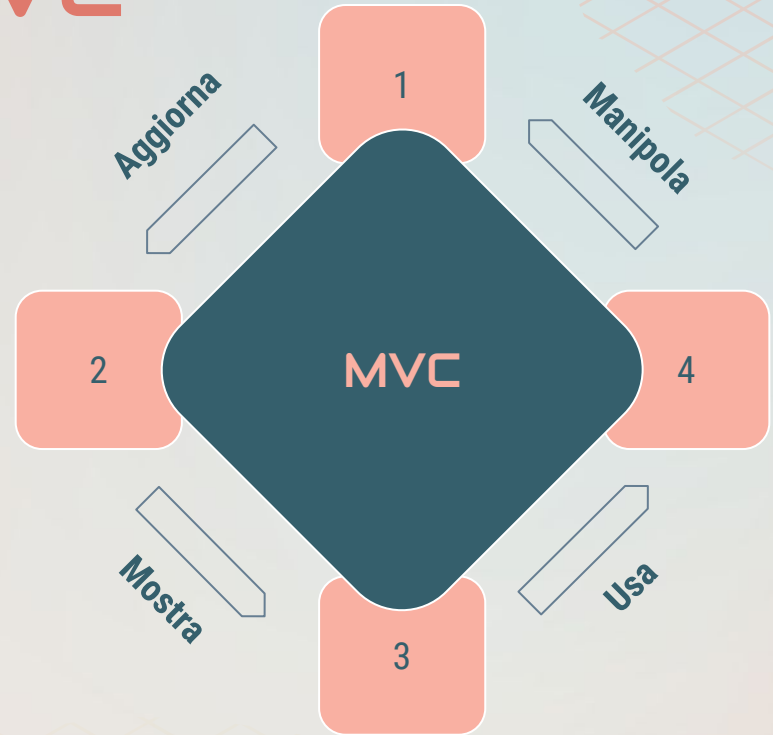
Architettura

Mvc



Architettura - MVC

- 1 **Utente** si occupa della gestione dei dati
- 2 **Controller** si occupa della gestione delle regole dell'applicazione e come rappresentare i dati
- 3 **Model** si occupa della gestione dei dati
- 4 **View** si occupa della gestione dell'interfaccia utente





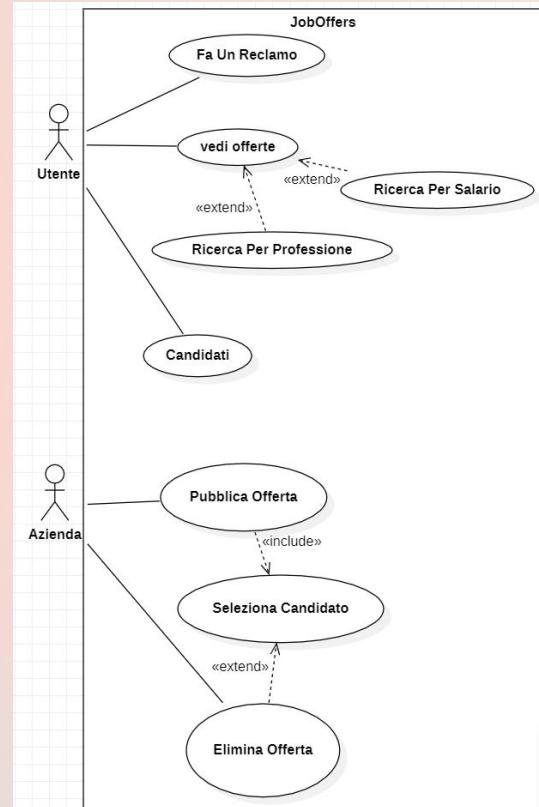
Modellazione

Use Case Diagram,
Class Diagram,
Sequence Diagram,
State Machine Diagram,
Activity Diagram.



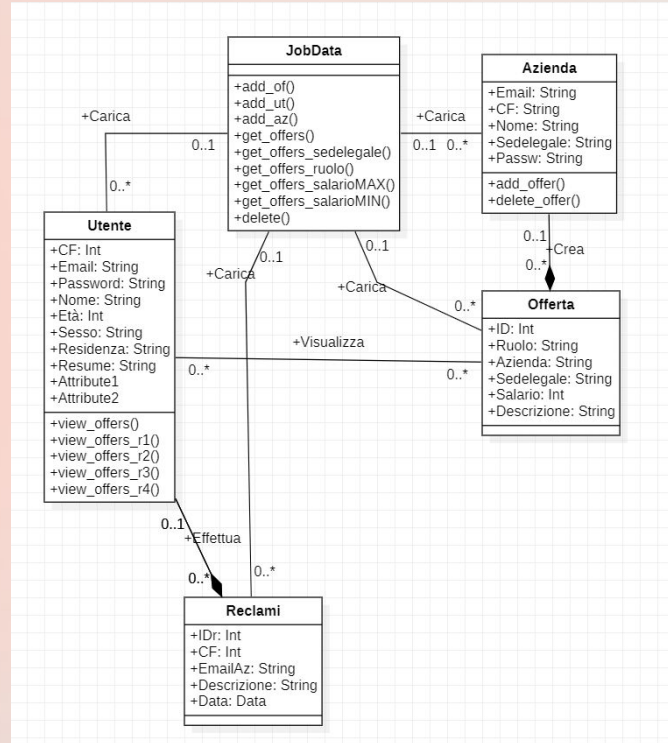
Use Case Diagram

Con questo diagramma si descrivono le varie funzioni che gli utenti e l'azienda possono effettuare



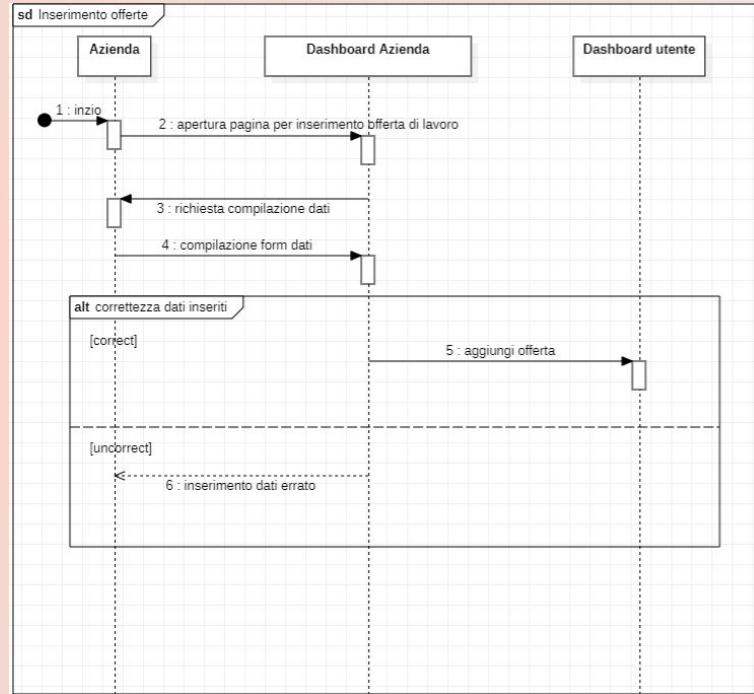
Class Diagram

Con questo diagramma descriviamo i vari stati che un utente percorre dalla candidatura alla comunica della selezione



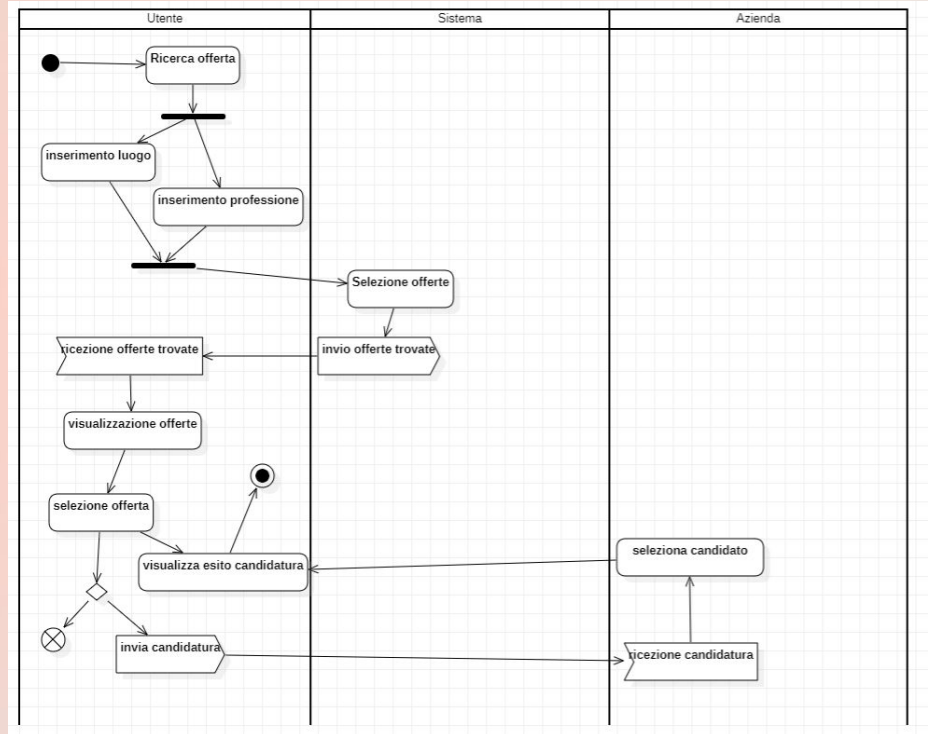
Sequence Diagram

Con questo diagramma descriviamo la sequenza di inserimento di un'offerta

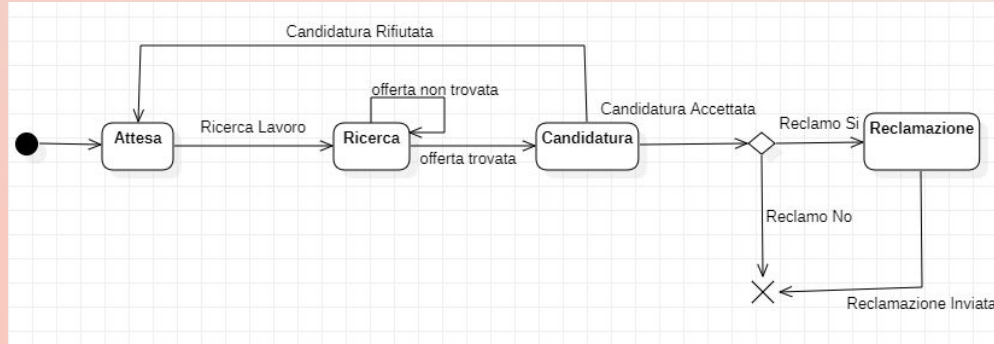


Activity Diagram

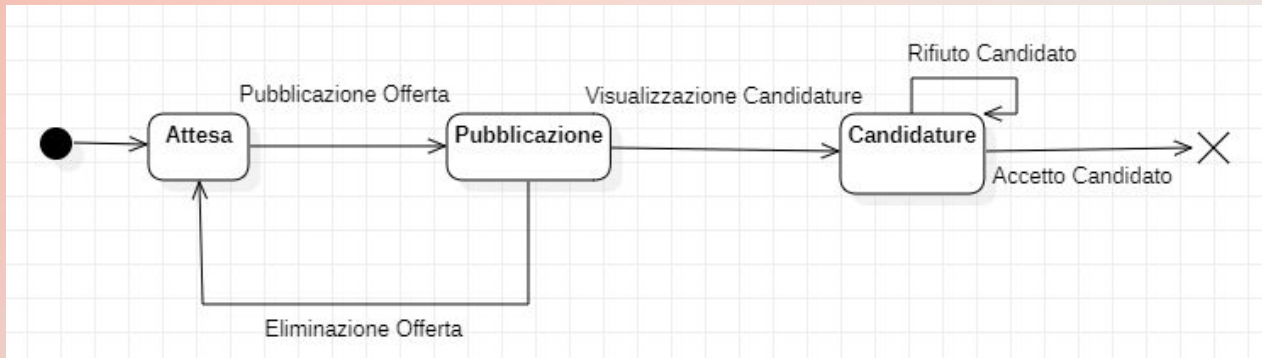
Con questo diagramma descriviamo l'attività della nostra web-app dalla ricerca offerta dell'utente alla selezione candidato dell'azienda



State Machine Diagram



Con questo diagramma descriviamo i vari stati che un utente percorre dalla candidatura alla comunica della selezione



Con questo diagramma descriviamo i vari stati che un'azienda percorre dalla pubblicazione dell'offerta alla selezione del candidato



Implementazione

Funzionalità per
l'azienda e per l'utente



Funzionalità azienda



**Registrazione
Profilo**



**Creazione e
pubblicazione
offerte**



**Possibilità di
accettare o
rifiutare le
offerte**



**Login
Profilo**



**Visualizzazione
candidati e
offerte**

Funzionalità utente



Registrazione
Profilo



Ricerca tramite
keyword delle
offerte



Possibilità di
candidarsi alle
offerte e
effettuare
reclami



Login
Profilo



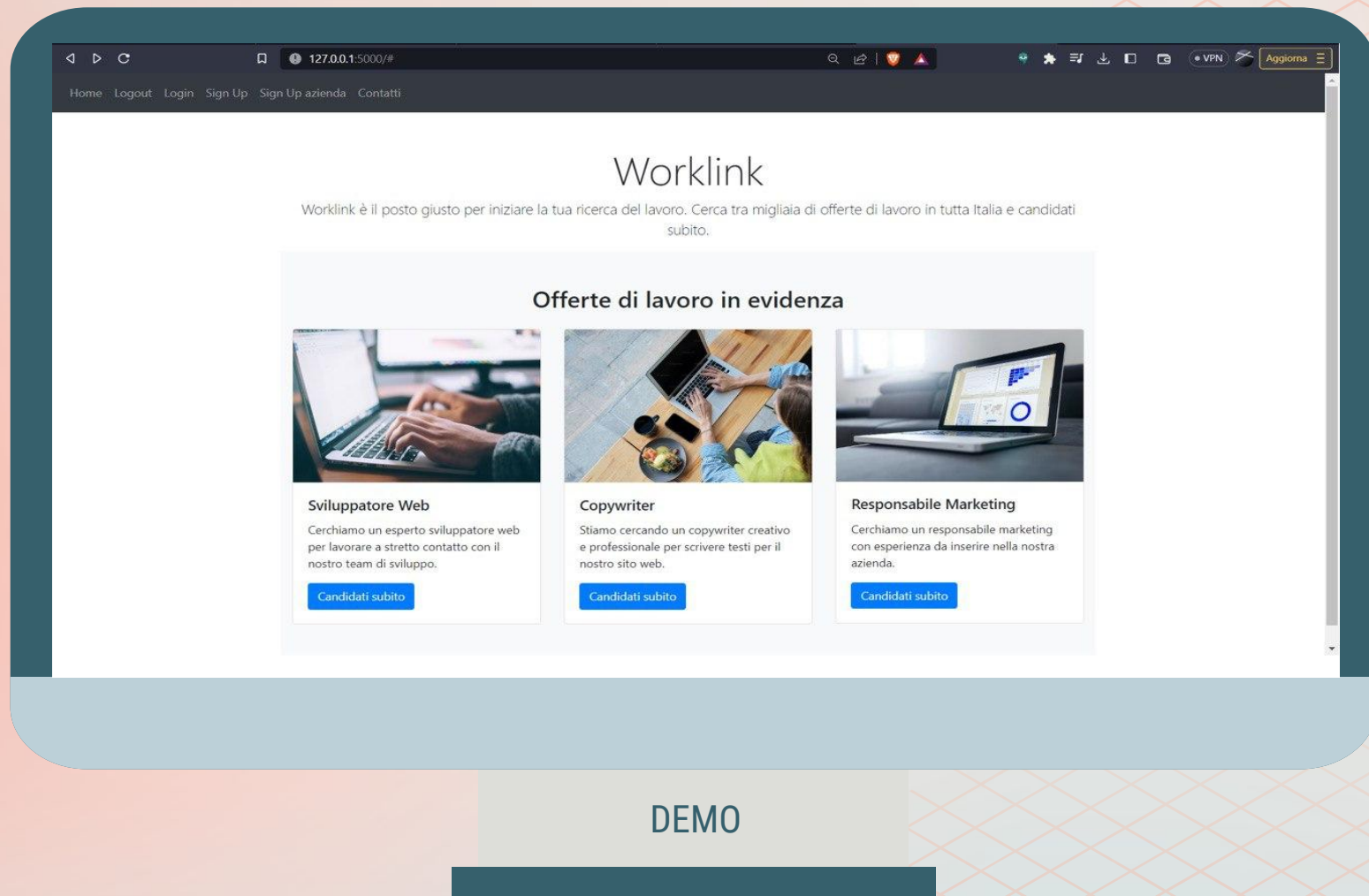
Visualizzazione
delle offerte



DEMO

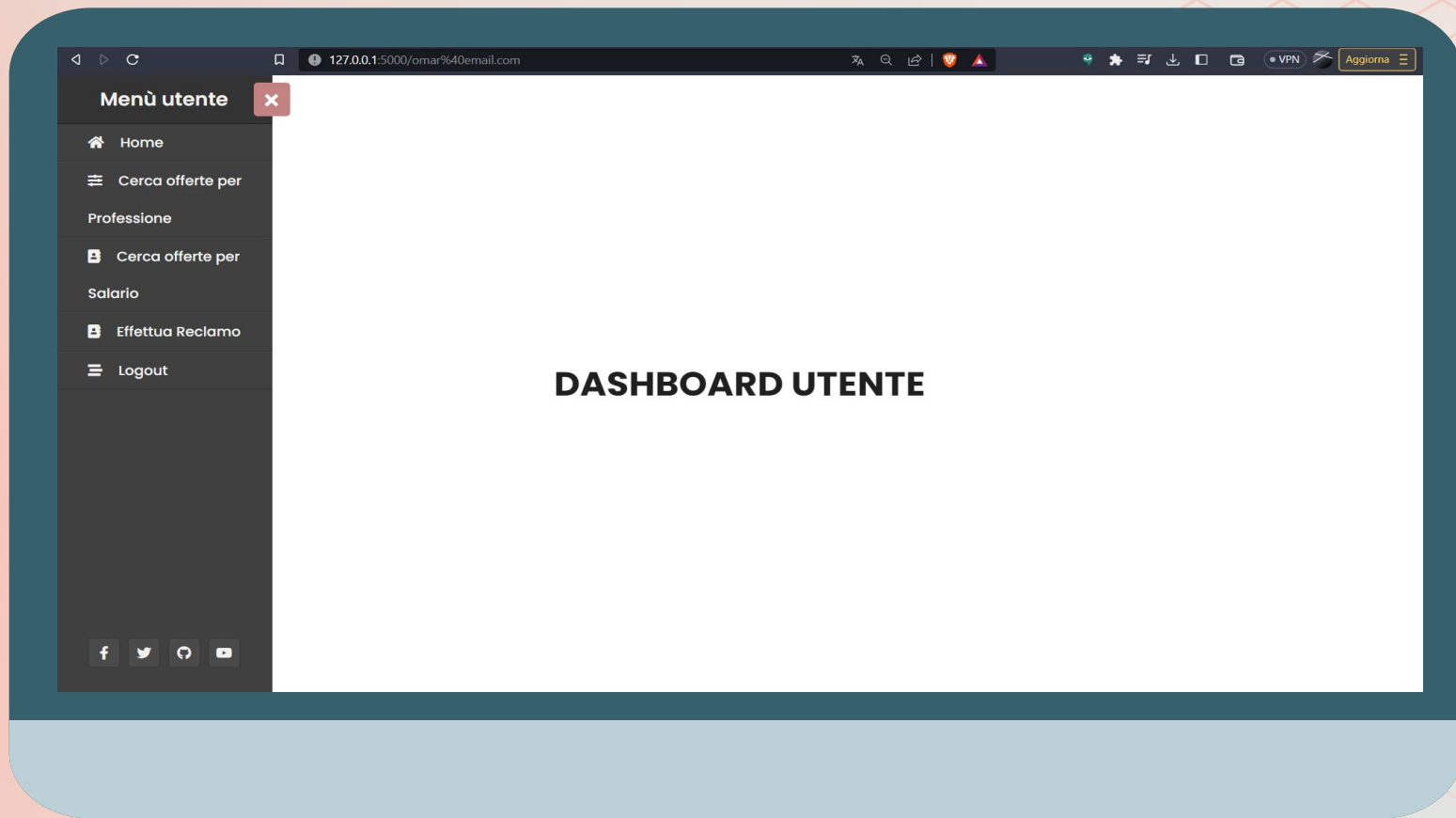
DEMO e un mock-up
della webapp



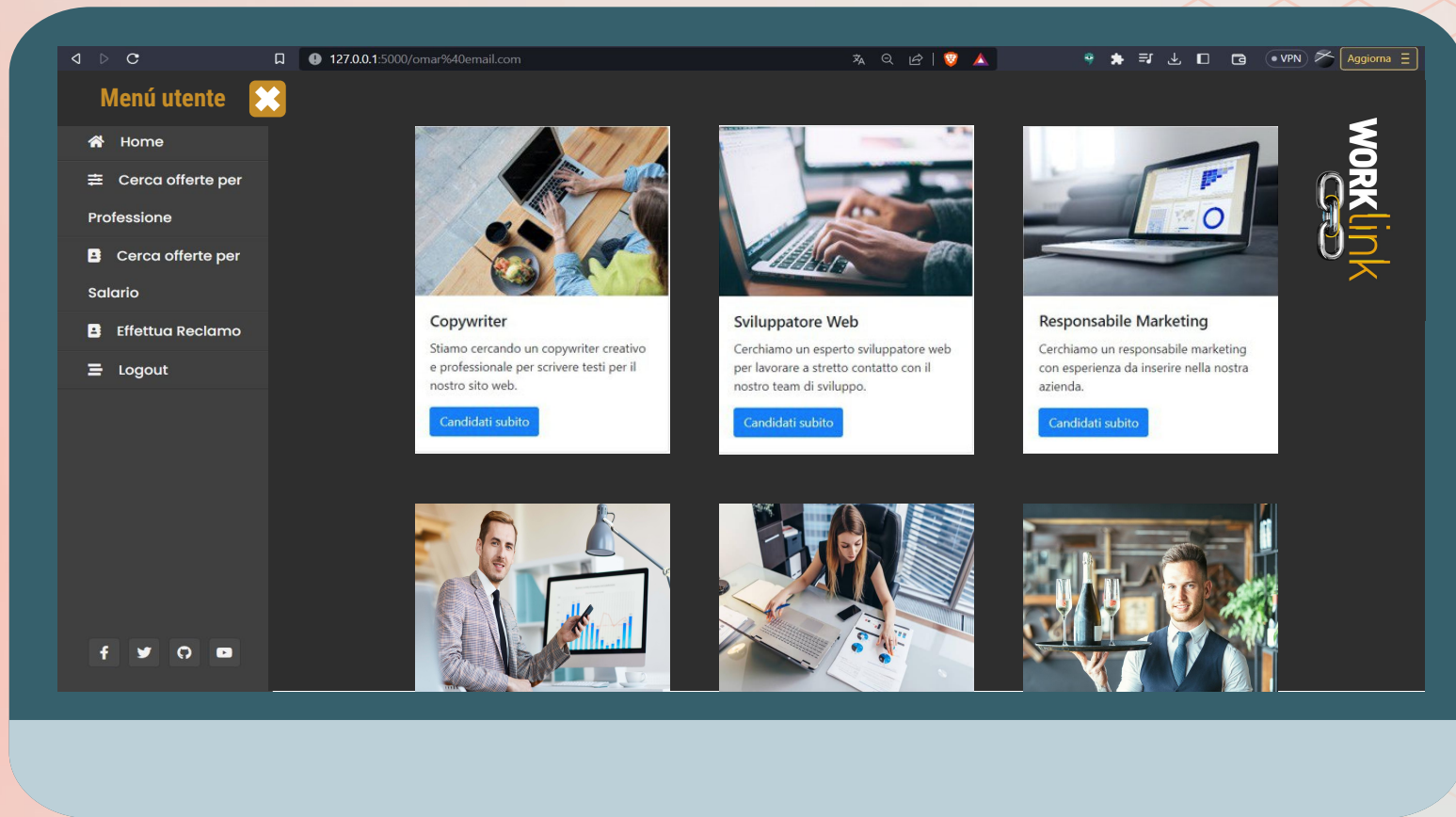


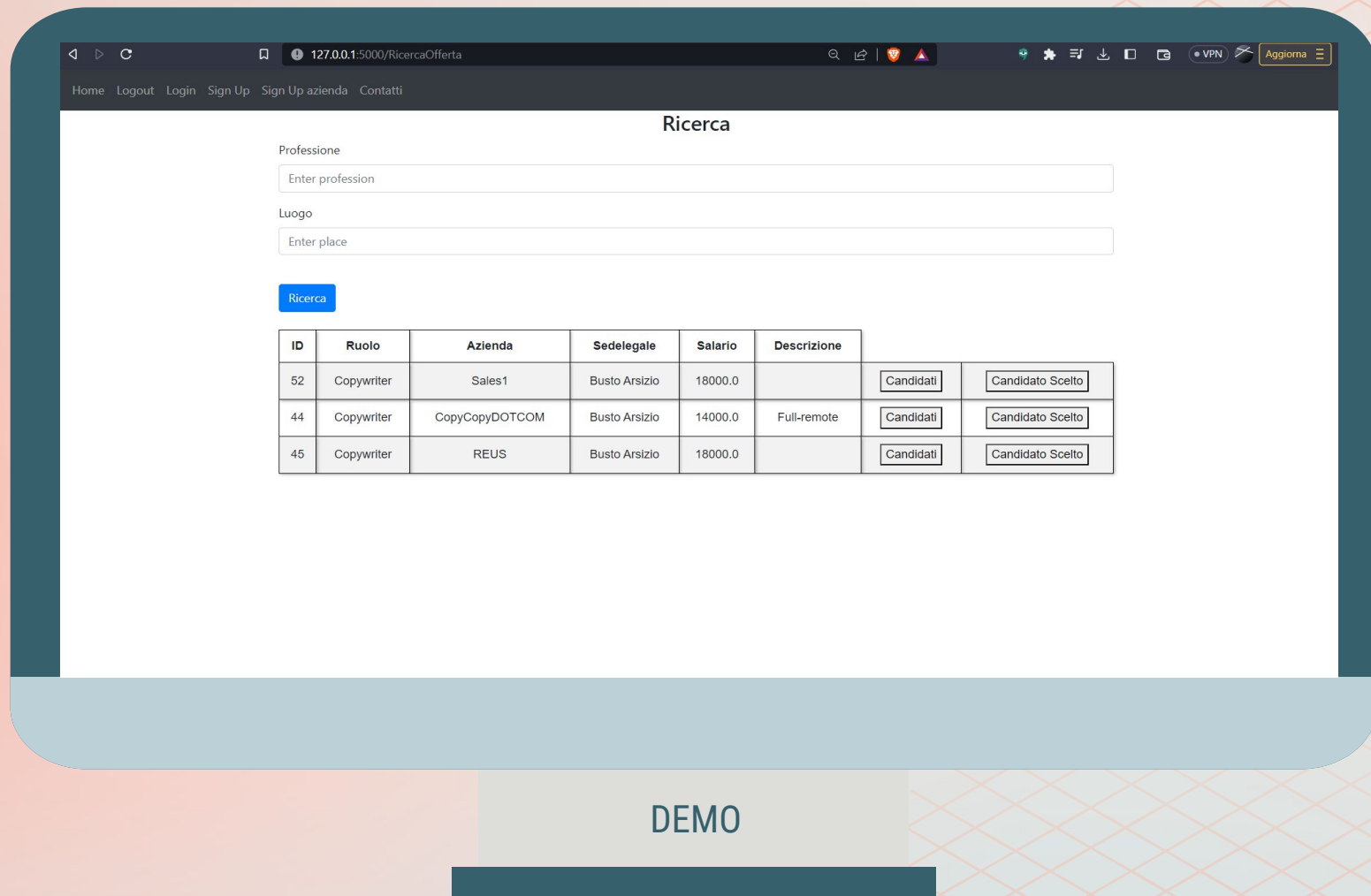


DASHBOARD



DEMO





127.0.0.1:5000/RicercaOfferta

Aggiorna

Home Logout Login Sign Up Sign Up azienda Contatti

Menú utente

Home

Cerca offerte per

Professione

Cerca offerte per

Salario

Effettua Reclamo

Logout

RICERCA

Professione:

Scrivi la tua professione

Luogo:

Scrivi il luogo

RICERCA

ID	Ruolo	Azienda	Sede legale	Salario	Descrizione
52	CopyWriter	Sales1	Milano	18.000	
44	CopyWriter	Copy2	Milano	14.000	Full-Remote

CANDIDATI

CANDIDATO SCELTO

CANDIDATI

CANDIDATO SCELTO

CANDIDATI

CANDIDATO

WORKlink

f

DEMO



Testing

You can enter a subtitle
here if you need it



Test

Abbiamo effettuato test principalmente sui metodi delle classi relative al database alle sue identità, che costituiscono una parte cruciale del codice.



Test

Abbiamo effettuato test principalmente sui metodi delle classi relative al database e alle sue identità, che costituiscono una parte cruciale del codice.

```
Code Blame 63 lines (57 loc) · 2.97 KB
1 import unittest
2 import sqlite3
3 from JobDatabase import JobData
4
5
6 class TestJobDatabase(unittest.TestCase):
7     def test_get_offers(self):
8         #crea database di test con sqlite3
9         conn = sqlite3.connect('test.db')
10        c = conn.cursor()
11        #crea tabella di test
12        c.execute('CREATE TABLE OfferteLavoro (ID TEXT PRIMARY KEY ,Titolo TEXT,Azienda TEXT,sedelegale TEXT,Ruolo TEXT,Salario TEXT)')
13        #inserisce dati di test
14        c.execute('INSERT INTO OfferteLavoro (ID,Titolo,Azienda,sedelegale,Ruolo,Salario) VALUES (?,?,?,?,'','','1','Titolo1','Azienda1','Sedelegale1','Ruolo1','Salario1)')
15        c.execute('INSERT INTO OfferteLavoro (ID,Titolo,Azienda,sedelegale,Ruolo,Salario) VALUES (?,?,?,?,'','','2','Titolo2','Azienda2','Sedelegale2','Ruolo2','Salario2)')
16        c.execute('INSERT INTO OfferteLavoro (ID,Titolo,Azienda,sedelegale,Ruolo,Salario) VALUES (?,?,?,?,'','','3','Titolo3','Azienda3','Sedelegale3','Ruolo3','Salario3)')
17        conn.commit()
18        conn.close()
19        #estraggo le offerte dal database di test
20        jobdata = JobData('test.db')
21        offers = jobdata.get_offers()
22        #verifico che le offerte estratte siano quelle inserite
23        self.assertEqual(offers[0][0],'1')
24        self.assertEqual(offers[1][0],'2')
25        self.assertEqual(offers[2][0],'3')
26        #elimino il database di test
27        conn = sqlite3.connect('test.db')
28        c = conn.cursor()
29        c.execute('DROP TABLE OfferteLavoro')
30        conn.commit()
31        conn.close()
```




Conclusioni

Punti di forza e
debolezze



Punti di Forza



Stesura
documenta
zione e
codice
parallele



Daily
scrum



GitHub

Debolezze



**Interfaccia
grafica**



Formazione



Casi di test