

Presentazione progetto d'esame Tecnologie Web

di Eduard Rubio Cholbi

Traccia

Progetto ed implementazione di un'applicazione Web sviluppata con Django.

- Applicazione Web dinamica basata sull'interazione con un database.
- Richiesta presenza di utenti anonimi e registrati con diritto di accesso differenziato.
- Piattaforma di "admin" riservata all'amministratore.
- Possibilità di caricamento immagini da utente.
- Richiesta produzione di test.

Descrizione

Il presente progetto nasce dall'idea di sviluppare un sito web che consenta di mettere in contatto persone che necessitano di supporto. Lo scopo principale è offrire una piattaforma nella quale gli utenti possono accordarsi per trovare una soluzione alle loro richieste. Queste ultime possono essere di svariato tipo, come ad esempio: richiedere l'esecuzione di un compito manuale, farsi recapitare un bene al proprio domicilio, organizzare un'uscita in compagnia o un'attività sociale.

Il target principale del progetto sono persone che necessitano di fare richieste semplici e veloci, pertanto abbiamo deciso di sviluppare un sito ottimizzato per le piattaforme mobili.

La web app è stata concepita in modo generico per coprire una vasta gamma di necessità, che possono variare dal bisogno di un dizionario all'aiuto per un trasloco. Pertanto, l'applicazione si focalizza principalmente sul coordinamento tra le persone, ma offre anche una chat integrata per consentire loro di comunicare e decidere le modalità di interazione preferite.

Abbiamo prestato particolare attenzione all'intuitività della piattaforma, utilizzando colori accattivanti e un sistema di valutazione tramite stelle. Le informazioni sono presentate in modo chiaro e visibile, con un'interfaccia minimalista.

Per esprimere la chiarezza del layout, l'immediatezza dell'interazione e la praticità della UX abbiamo scelto il termine "Knock-Knock" per identificare le attività ed i lavori richiesti dagli utenti.

Dal punto di vista tecnico, abbiamo voluto realizzare un sito che potesse funzionare completamente offline (per evitare problemi durante le dimostrazioni) e che fosse facilmente portabile in diversi ambienti. Pertanto, abbiamo integrato le librerie front-end nel codice e adottato script bash per la creazione degli ambienti virtuali.

Tecnologie

Django

La scelta di Django rappresentava fin dall'origine un prerequisito per lo sviluppo del presente progetto, pertanto non verrà discussa in questo documento.

Git

La prima scelta che abbiamo effettuato riguarda il controllo di versione del codice, poiché era fondamentale mantenere una traccia delle modifiche effettuate e avere la possibilità di eseguire un "rollback" in caso di necessità. Abbiamo optato per l'utilizzo di Git per via della sua semplicità e praticità d'uso rispetto ad altri sistemi.

Abbiamo inoltre scelto di usufruire del servizio GitHub, nonché nota piattaforma ampiamente utilizzata in tutto il mondo, grazie alla quale abbiamo reso pubblico il codice e con cui abbiamo la possibilità di coinvolgere in futuro membri della community che potranno contribuire allo sviluppo di nuove funzionalità o alla correzione di eventuali errori.

Inoltre, abbiamo aggiunto al progetto uno strumento chiamato "pre-commit" che ci ha permesso di integrare un elenco di estensioni per il controllo della qualità del codice ad ogni aggiornamento. Di seguito sono elencate alcune delle estensioni che si sono rivelate particolarmente utili durante lo sviluppo del progetto:

- Autoflake: ci ha consentito di rimuovere il codice obsoleto, concentrandoci sempre sul codice più rilevante.
- Autopep8: ci ha permesso di mantenere una formattazione coerente, il che è molto utile quando si lavora in team, ma anche nel contesto di uno sviluppo individuale. Inoltre, ha consentito a Git di ridurre il numero di modifiche causate solo dalla formattazione del codice.
- Isort: ha garantito un ordinamento coerente delle librerie nel codice, offrendo gli stessi benefici che abbiamo ottenuto con autopep8.
- Djlnt e Pylint: hanno consentito il controllo degli errori comuni nel codice, aiutandoci a evitare piccoli errori di implementazione.

SQLite3

Abbiamo scelto di utilizzare il database relazionale SQLite3 per la sua facilità di sviluppo e le buone prestazioni iniziali, considerando che il progetto attualmente non gode di una grande visibilità. Questa scelta ci ha consentito di accelerare lo sviluppo iniziale senza sacrificare le prestazioni.

Abbiamo l'idea di migrare a un database PostgreSQL in futuro, poiché ci permetterebbe di utilizzare i moduli GIS per implementare un sistema di coordinate e ricerca basato sulla prossimità. Tuttavia, al momento la mancanza di Docker come tecnologia potrebbe influire sulla portabilità e l'installazione del progetto. Pertanto, abbiamo deciso di rimandare la migrazione fino a quando avremo implementato Docker per semplificare la gestione dell'ambiente di sviluppo.

Django apps

Oltre a Django, abbiamo sfruttato diverse app (plugin o librerie) per ottenere funzionalità aggiuntive e risparmiare tempo di sviluppo. Di seguito sono elencate le app che abbiamo principalmente utilizzato:

Django Components

Questa applicazione ci ha consentito di implementare un rendering basato su "componenti" per il front-end del nostro sito web. Questo approccio ci ha permesso di evitare la duplicazione del codice in diversi punti del progetto, evitando così la necessità di apportare modifiche manualmente in ogni punto in cui viene utilizzato il componente.

Uno dei principali vantaggi di questa libreria è la leggibilità del codice, poiché ci permette di incapsulare la logica di presentazione all'interno di ciascun componente, semplificando la lettura e la comprensione quando il componente viene utilizzato.

Channels

La libreria "Channels" è stata fondamentale per implementare la comunicazione sincrona tra gli utenti del sito, in particolare per sviluppare la chat integrata. Questa libreria offre un'ottima integrazione con Django, rendendo la sua scelta molto conveniente per il progetto.

Grazie a Channels, siamo stati in grado di gestire in modo efficiente la comunicazione istantanea tra gli utenti del sito. Questa funzionalità ha permesso agli utenti di interagire in tempo reale tramite la chat integrata, facilitando la collaborazione e migliorando l'esperienza complessiva degli utenti.

Django AP Scheduler

Il componente di esecuzione temporizzata ci permette di programmare l'esecuzione automatica di determinato codice in base a specifici intervalli di tempo. Attualmente, questa funzionalità viene utilizzata per la chiusura automatica delle attività una volta completate. Tuttavia, abbiamo previsto l'adozione di questo componente anche in ottica di future modifiche e ulteriori funzionalità.

L'adozione di questo componente tiene conto delle potenziali esigenze future e offre la possibilità di estendere le funzionalità del progetto senza dover apportare modifiche significative all'architettura esistente.

Django Money

La libreria che abbiamo utilizzato ci ha fornito una soluzione per gestire i dati di valuta nel nostro progetto Django. Poiché Django non offre un'integrazione nativa per la gestione delle valute, abbiamo adottato questa libreria per consentire attualmente il supporto per i lavori pagati in euro, ma con poche modifiche è possibile abilitare altri tipi di valuta.

Pillow

La libreria che abbiamo utilizzato ci ha consentito di effettuare operazioni sulle immagini nel nostro progetto. Tuttavia, nel contesto specifico del progetto, abbiamo utilizzato questa libreria principalmente per gestire le immagini del profilo degli utenti, poiché consentiva agli utenti di caricare immagini di diversi formati.

Faker

La libreria che abbiamo utilizzato ci ha fornito un meccanismo per generare dati casuali da utilizzare per scopi di test nel nostro progetto. Sebbene questa libreria non venga utilizzata direttamente nel funzionamento del progetto, è stata estremamente utile per generare dati di prova durante lo sviluppo e i test.

La struttura della libreria offre una serie di metodi che consentono la generazione di dati casuali sia in formato strutturato di Python che in formato testuale. Ad esempio, è possibile generare nomi di persone, date, numeri casuali o testi casuali in base alle nostre esigenze specifiche.

Questa libreria ci ha permesso di simulare dati realistici e variegati durante i test, fornendo un ambiente di sviluppo più realistico e consentendo di rilevare e correggere eventuali problemi o anomalie nel codice in modo più accurato.

Django Debug Toolbar

Abbiamo incluso un plugin nel progetto che, sebbene attualmente sia disattivato per la sua invasività iniziale, si occupa di generare report di profilazione per le varie pagine del sito al fine di identificare potenziali problemi di prestazioni. Questo plugin si è dimostrato estremamente utile nel monitorare la quantità e le performance delle query generate da Django.

Attraverso l'analisi dei report di profilazione, siamo stati in grado di valutare le query eseguite e individuare eventuali punti critici o aree che richiedevano miglioramenti per ottimizzare il codice e garantire una maggiore efficienza del sistema.

Docker

Nella progettazione del sito, abbiamo valutato l'utilizzo di Docker, ma alla fine non è stato integrato per una serie di motivi. L'idea principale era di sfruttare Docker per poter utilizzare altri strumenti come Nginx e PostgreSQL, al fine di beneficiare delle loro caratteristiche specifiche. Tuttavia, dopo aver valutato l'aumento della complessità dell'infrastruttura, il tempo richiesto per l'integrazione e le configurazioni necessarie per un utilizzo efficiente, abbiamo deciso che i benefici attesi non superavano gli svantaggi.

Organizzazione

Il codice del progetto è stato organizzato seguendo le best practice di Django per garantire una struttura delle cartelle chiara, una nomenclatura dei file standard e una corretta denominazione di metodi e classi. Ciò è stato fatto al fine di facilitare la lettura e la comprensione del codice da parte di altri sviluppatori.

Inoltre, grazie alle estensioni menzionate in precedenza, sono stati applicati gli standard PEP8 per la formattazione del codice, cercando di mantenere una nomenclatura ordinata per variabili e metodi.

Per accelerare lo sviluppo del software, non è stata eseguita la suddivisione in app di Django durante la fase di prototipazione. Tuttavia, è stata considerata la chat interna come un potenziale caso d'uso per la suddivisione delle strutture in app separate. Nonostante ciò, questa implementazione è stata rimandata come possibile sviluppo futuro.

Motivazioni

Di seguito elencherò alcune delle scelte che sono state prese durante lo sviluppo del progetto, insieme al percorso successivo che è stato intrapreso:

Come possiamo generare dati casuali da amministratore?

Per generare dati casuali da amministratore, sono state prese alcune scelte strategiche durante lo sviluppo del progetto. Ecco un riassunto delle scelte e dei percorsi intrapresi:

1. Integrazione con i comandi CLI di Django: è stata adottata l'integrazione di Django con i comandi da linea di comando (CLI) per la generazione dei dati casuali. Questa scelta è stata motivata dalla volontà di limitare l'accesso a coloro che hanno privilegi di amministratore al server, riducendo così la superficie di attacco. L'utilizzo dei comandi CLI ha consentito la generazione di dati casuali direttamente da ambiente amministrativo senza la necessità di esporre API REST specifiche.
2. Generazione dei dati in varie fasi: per agevolare la generazione di scenari diversi, è stata adottata la strategia di suddividere la generazione dei dati casuali in varie fasi. Ad esempio, è stato possibile generare rapidamente una grande quantità di "Knock-Knock" finiti per testare la votazione tra utenti, ma bloccando la votazione stessa. Questa suddivisione in fasi ha consentito di creare casi di test specifici e di effettuare verifiche mirate durante lo sviluppo.
3. Script eseguibile per automatizzare la generazione: al fine di semplificare ulteriormente il processo di generazione dei dati casuali, è stato preparato uno script eseguibile che automatizza questa fase. Questo script può essere eseguito per generare dati casuali in modo ripetitivo, fornendo un supporto prezioso durante lo sviluppo e le sessioni di demo.

La combinazione di queste scelte ha consentito di generare dati casuali da amministratore in modo efficace e ripetibile, fornendo un sistema comodo e flessibile per le demo e il testing.

Come possiamo eseguire script di inizializzazione dei dati in automatico?

Per eseguire automaticamente gli script di inizializzazione dei dati, è stato creato un comando da terminale (CLI) dedicato. Questo comando viene incluso negli script eseguibili per automatizzare il processo di configurazione del sistema. Gli amministratori di sistema possono scegliere se eseguire manualmente le operazioni o utilizzare lo script per una configurazione automatica. Ciò garantisce la portabilità del sistema e semplifica il monitoraggio e la correzione di eventuali errori durante il processo di inizializzazione dei dati.

Come possiamo gestire gli eventi che avvengono temporalmente senza necessità di interazione dell'utente?

In questo progetto è emersa la necessità di concludere automaticamente i Knock-Knock al termine dell'attività per permettere alle persone di poter effettuare una valutazione degli altri utenti.

La strada scelta è stata quella di controllare costantemente e in tempi configurabili lo stato di questi ed aggiornarlo di conseguenza. Per questo ci siamo affidati ad una libreria sopracitata.

Abbiamo voluto evitare di affidare la chiusura dei Knock-Knock all'attività diretta dell'utente, data la possibilità da parte di questi ultimi di non rientrare più nell'app a seguito del termine dell'attività. In tal maniera siamo riusciti ad evitare la presenza di blocchi in stadi intermedi.

Come possiamo mettere in comune il template base del front-end?

Per mantenere uno stile grafico comune e ridurre la disorientazione degli utenti, è stato creato un template base con librerie grafiche essenziali e un header pre-impostato. Questo template viene utilizzato in tutte le pagine del sito, consentendo agli sviluppatori di concentrarsi sul contenuto specifico di ciascuna pagina, senza dover ripetere la logica del template. Questa soluzione assicura uno stile coerente e semplifica lo sviluppo e la manutenzione del sito.

Come possiamo far comunicare gli utenti tra di loro?

Per consentire la comunicazione tra gli utenti, sono state valutate diverse opzioni, tenendo conto delle esigenze specifiche del progetto. Ecco un riassunto delle scelte prese:

1. Sistema di messaggistica tipo email: inizialmente, è stato considerato l'utilizzo di un sistema simile alla posta elettronica per consentire una rapida implementazione. Tuttavia, questa soluzione presentava dei limiti in termini di reattività tra gli interlocutori, poiché spesso è necessario uno scambio di messaggi veloci e concisi.
2. Chat asincrona e chat sincrona: successivamente, è stata considerata l'opzione di implementare una chat, eliminando la necessità di specificare titoli e destinatari e semplificando il processo dal punto di vista grafico. La chat poteva essere implementata in due modi: "asincrona" o "sincrona". Nella chat asincrona, gli utenti potevano inviare messaggi in qualsiasi momento, ma ricevevano le risposte solo quando navigavano nel sito, simili a notifiche. Nella chat sincrona, invece, i messaggi venivano ricevuti immediatamente dall'altro utente. Tra le due opzioni, la chat sincrona offriva i benefici necessari, ed il tempo di implementazione è stato valutato come simile.

Pertanto, è stata scelta la soluzione di una chat "diretta", che permetteva una comunicazione sincrona tra gli utenti. Questa scelta è stata fatta per garantire una

comunicazione immediata e reattiva, consentendo agli utenti di scambiarsi messaggi in tempo reale durante la navigazione sul sito.

Quali sarebbero i dati che un utente vorrebbe vedere nella propria homepage?

La homepage è stata progettata in modo da offrire una semplice webapp con una visualizzazione pulita e un'organizzazione delle funzionalità che non sia eccessivamente carica. Ecco un riassunto dei dati che un utente vorrebbe vedere nella propria homepage:

1. Utenti Guest: per gli utenti non registrati (Guest), l'obiettivo principale della homepage è incoraggiarli a registrarsi sul sito. Pertanto, la visualizzazione è focalizzata sulla promozione delle interazioni tra gli utenti, senza limitare la visibilità a attività specifiche. L'obiettivo è creare un ambiente accogliente che stimoli gli utenti a partecipare e interagire con gli altri.
2. Utenti loggati: per gli utenti registrati e loggati, la homepage è suddivisa in tre categorie:
 - a. Attività personali: questa sezione mostra le attività che l'utente ha creato o a cui sta partecipando, inclusi quelli che potrebbero essere già passati ma non conclusi. Questa sezione incoraggia l'utente a partecipare alle attività e a votare gli altri partecipanti, creando un senso di coinvolgimento e interazione all'interno della comunità.
 - b. Attività in sospeso: in questa sezione vengono visualizzate le attività che l'utente potrebbe desiderare accettare, ma che non sono ancora state confermate. Questa sezione offre la possibilità di annullare la partecipazione a queste attività se l'utente cambia idea o ha dubbi, rendendo l'impegno meno vincolante fino a quando non viene effettivamente confermato.
 - c. Knock-Knock degli altri utenti: questa sezione mostra i "Knock-Knock" degli altri utenti, offrendo all'utente la possibilità di offrire il proprio aiuto o supporto. Questo invita l'utente a partecipare attivamente e a fornire assistenza ad altri membri della comunità.

L'obiettivo principale nella progettazione della homepage è fornire una visualizzazione chiara e intuitiva delle interazioni e delle opportunità presenti nel sito, sia per gli utenti Guest che per gli utenti registrati.

Come deve funzionare il flusso di tutto un Knock-Knock?

Il flusso di un "Knock-Knock" nel progetto segue il seguente processo:

1. Creazione della richiesta: gli utenti hanno la possibilità di creare liberamente richieste e renderle pubbliche. Queste richieste rappresentano le attività che gli utenti hanno bisogno di svolgere o di cui hanno bisogno di assistenza.
2. Richiesta di informazioni aggiuntive: gli altri utenti interessati possono opzionalmente richiedere ulteriori informazioni sulla richiesta avviando una chat con l'utente che ha creato la richiesta. Questo consente di ottenere dettagli specifici e chiarire dubbi.
3. Manifestazione dell'interesse: gli utenti interessati possono manifestare la propria volontà di partecipare ad una specifica attività. Questo dimostra l'interesse e la disponibilità a fornire il supporto richiesto.
4. Accettazione della proposta: L'utente che ha creato la richiesta ha la possibilità di chattare con gli utenti interessati per avere ulteriori informazioni o conoscerli meglio.

- Quando interessato, l'utente può accettare la proposta di uno degli utenti interessati.
5. **Votazione dell'altro utente:** Una volta terminata l'attività, il sistema propone la votazione dell'altro utente coinvolto. La votazione non è bloccante, ma viene incentivata tramite la scomparsa dell'attività conclusa dalla homepage dell'utente. In questo modo, si incoraggia attivamente la partecipazione e la valutazione reciproca degli utenti.
 6. **Consultazione del "Knock-Knock" concluso:** Il "Knock-Knock" concluso può essere consultato nel proprio profilo o da altri utenti interessati. Questo consente di visualizzare il dettaglio dell'attività svolta e le valutazioni degli utenti coinvolti.

Com'è meglio che gli utenti si valutino tra di loro?

Per permettere agli utenti di valutarsi reciprocamente, sono state valutate diverse opzioni considerando le esigenze specifiche del progetto. Alla fine, è stato scelto un sistema di valutazione basato sul rating "a stelle" per i seguenti motivi:

1. **Flessibilità nella valutazione:** un sistema di rating "a stelle" consente agli utenti di esprimere una valutazione generale dell'esperienza, senza limitarsi a commenti specifici. Questo offre una maggiore flessibilità e facilità di utilizzo per gli utenti che desiderano valutare l'altro utente in base al loro livello di soddisfazione complessivo.
2. **Rappresentazione visiva chiara:** l'utilizzo di stelle come simbolo di valutazione fornisce una rappresentazione visiva immediata della valutazione media di un utente. Questo permette agli altri utenti di avere una visione rapida e intuitiva delle prestazioni e della reputazione di un utente.
3. **Votazione positiva e negativa:** il sistema di rating "a stelle" consente agli utenti di esprimere sia valutazioni positive che negative. Ciò significa che possono valutare negativamente le esperienze in cui l'utente non si presenta, non svolge il lavoro richiesto o il richiedente non paga. D'altra parte, possono anche fornire una valutazione positiva per indicare un lavoro ben svolto.
4. **Valutazione media:** il sistema di rating "a stelle" consente di calcolare una valutazione media per ogni utente in base alle valutazioni ricevute. Questo offre un modo per mostrare un voto medio che può essere utilizzato per fornire un'indicazione generale della reputazione dell'utente e può essere visualizzato nei nuovi "Knock-Knock" o nei profili degli utenti.

Come possiamo dividere in categorie i vari Knock-Knock?

La gestione delle attività all'interno del sito, considerando la vasta gamma di possibilità, è stata affrontata attraverso la categorizzazione. Inizialmente, si è scelto di avere delle macro-categorie generali che rappresentano le idee più plausibili per il lancio iniziale del sito. Queste macro-categorie comprendono:

1. **Comprare qualcosa al posto tuo:** questa categoria consente agli utenti di richiedere assistenza per l'acquisto di beni o servizi. Ad esempio, un utente potrebbe richiedere a qualcuno di acquistare generi alimentari o prenotare un appuntamento al suo posto.
2. **Incontrarsi per effettuare qualcosa insieme:** questa categoria riguarda le attività in cui gli utenti desiderano incontrarsi e svolgere un'attività insieme. Ad esempio, potrebbe riguardare la ricerca di qualcuno con cui fare una passeggiata, giocare a un gioco da tavolo o fare esercizio fisico.

3. Fare qualcosa al posto mio: questa categoria consente agli utenti di chiedere a qualcuno di svolgere un'attività al loro posto. Ad esempio, potrebbe riguardare la richiesta di qualcuno che si occupi del giardinaggio, faccia delle riparazioni o svolga altre attività domestiche.

L'idea principale è di monitorare l'evoluzione del sito e valutare mano a mano con l'aumento dei dati raccolti le categorie che si rivelano più utili in base all'utenza e alle richieste effettuate. Questo processo consentirà di adattare e ottimizzare le categorie esistenti e, se necessario, introdurre eventuali sotto-categorie specifiche per rispondere meglio alle esigenze degli utenti.

Test

Durante lo sviluppo del progetto, sono stati effettuati alcuni test, sebbene in numero limitato a causa delle restrizioni di tempo. I test sono stati focalizzati su due aree principali:

1. Test della homepage: è stato verificato che la homepage del sito funzionasse correttamente e che fossero ricevuti i dati necessari. Questo test è stato importante per assicurarsi che il sito fosse in grado di mostrare correttamente le informazioni agli utenti e consentire loro di interagire con le funzionalità principali.
2. Test della logica di calcolo delle stelle del profilo: data l'importanza del sistema di valutazione basato sul rating "a stelle", è stato necessario testare la logica di calcolo delle stelle del profilo. È stata verificata la corretta esecuzione di questa procedura, che viene attivata ad ogni nuovo voto. È stato fondamentale assicurarsi che la logica di calcolo funzionasse correttamente e che fosse in grado di fornire una valutazione accurata e aggiornata del profilo dell'utente. Durante lo sviluppo, questa funzionalità è stata disattivata temporaneamente in diverse occasioni, quindi è stato importante verificare che fosse ancora funzionante dopo ogni riattivazione.

Risultati

Il sito si propone con una homepage dove si possono vedere:



- Barra di navigazione con presente il logo, il tasto per creare nuovi contenuti (visibile solo ad utenti loggati) e il menu a discesa.
- Elenco dei miei Knock-Knock: dove sono presenti le attività in cui ho ancora interazioni da fare.
- Elenco dei Knock-Knock in corso: dove sono presenti attività per cui ho fatto richiesta
- Elenco dei Knock-Knock degli altri utenti: dove un utente può vedere le richieste aperte degli altri utenti ed registrarsi a nuovi lavori che vorrebbe svolgere

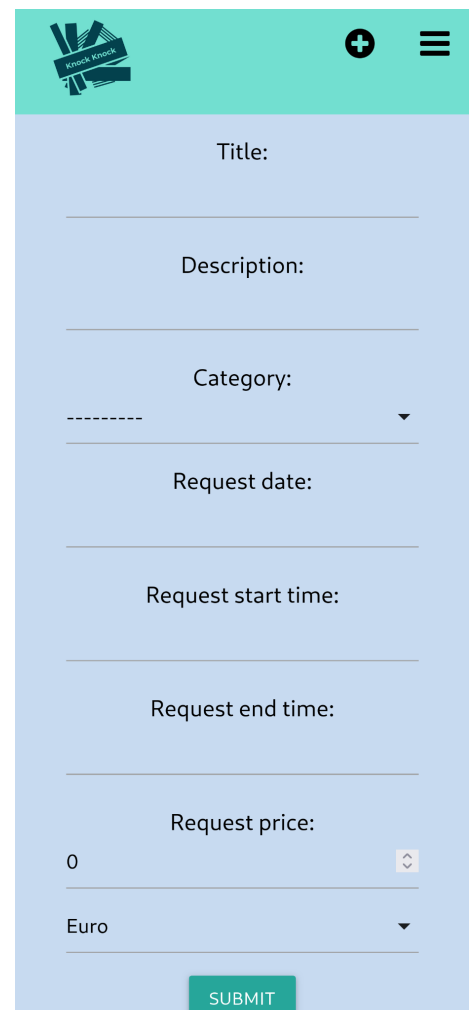
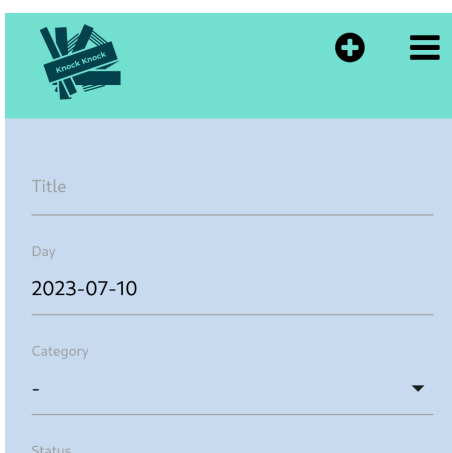
Per ogni Knock-Knock sono visibili varie informazioni:

- Titolo breve
- Periodo della richiesta
- Descrizione dell'attività
- Icona del richiedente (a sinistra) con relative stelle, nel caso l'altro utente abbia già votato
- Prezzo, se presente
- Categoria (BUY/MEET/DO)
- Icona con il numero di chat attive che diventa un utente appena il richiedente gli assegna il lavoro
- Colore di sfondo: indica una prima anteprima dello stato del Knock-Knock, l'idea è quella di abituare gli utenti ai colori del sito in modo da identificare velocemente come procede ogni attività

Cliccando l'icona del "+" in alto a destra si riescono a creare nuove richieste di attività:

Questo è un classico form di inserimento dati.

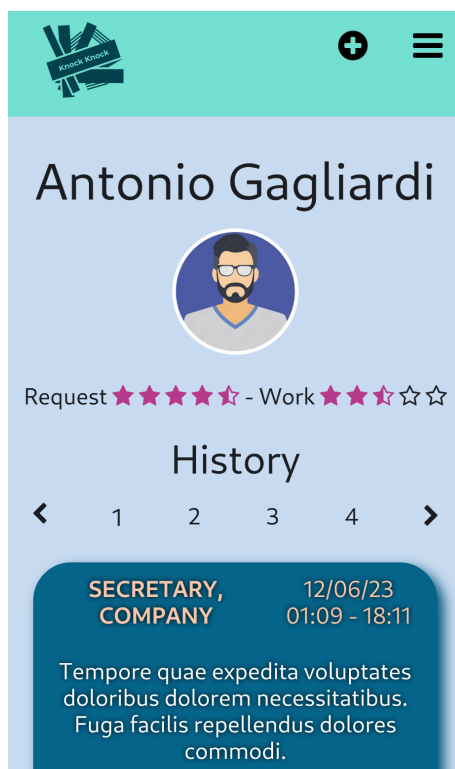
Sono state scelte alcune librerie grafiche conosciute per l'inserimento di dati come categoria, data e orari in modo da semplificare l'inserimento dei dati da parte dell'utente.

Tramite il menu a discesa (l'hamburger) è possibile accedere alla pagina di ricerca.

Attualmente ci sono quattro possibili modi per cercare nuove attività:

- Tramite titolo, viene effettuata una ricerca sul titolo in modalità “contiene questo testo”
- Tramite data del attività
- Tramite categoria di attività
- Tramite stato corrente delle attività

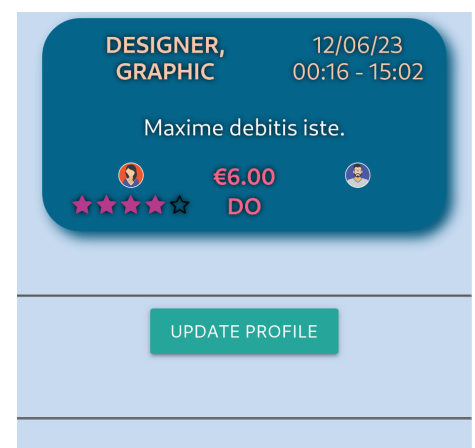


Cliccando su una qualunque icona di un utente si può accedere al suo profilo:

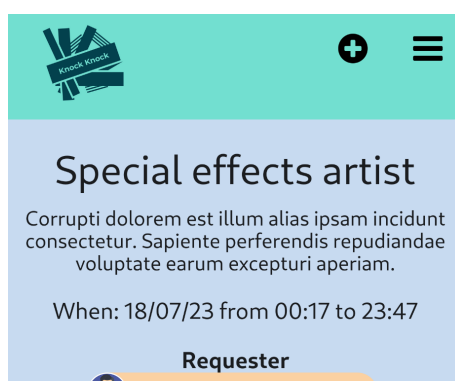
Qui si possono vedere le informazioni dell'utente, il voto attuale per quanto riguarda come richiedente e come lavoratore e la storia dei suoi Knock-Knock.

L'idea di questa pagina è quella di offrire una panoramica agli altri utenti delle mie attività recenti e permettere agli altri utenti di farsi un'idea di che tipo di lavori chiede o svolge. Invece per quanto riguarda la propria utenza è utile per vedere il proprio storico di attività.

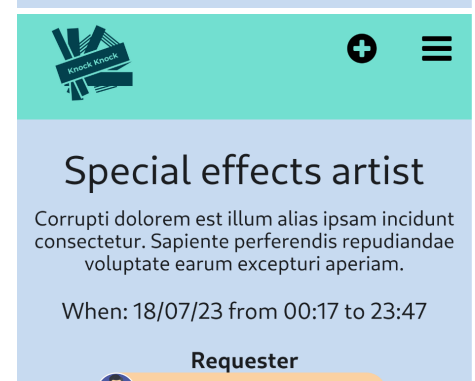
In fondo alla pagina è possibile accedere alla modifica delle informazioni del proprio profilo.



Questa è la pagina principale dove avvengono la maggior parte di attività:



A sinistra si può vedere l'utente che ha creato il Knock-Knock



dove ha la possibilità di cancellare l'attività nel caso si renda conto che non ha più bisogno. Quest'ultima operazione è possibile soltanto finché non ha accettato un altro utente ad aiutarlo.

A destra si vede lo stesso evento visto da parte di un altro utente.


Gli altri utenti possono decidere di registrarsi direttamente all'evento oppure chiedere più informazioni tramite la chat.

Special effects artist

Corrupti dolore est illum alias ipsam incidunt consectetur. Sapiente perferendis repudiandae voluptate earum excepturi aperiam.

When: 18/07/23 from 00:17 to 23:47


Requester

 ANTONIO GAGLIARDI - 4.25 ★




Free

OPEN RESERVED IN PROGRESS DONE CLOSED

Open chats

 NATALIA NIBALI - 3.00 ★

User submissions

 NATALIA NIBALI - 3.00 ★  




July 6, 2023, 4:05 p.m.

DELETE

Qui possiamo vedere come l'utente abbia aperto sia una chat ed abbia anche richiesto di lavorare per questa attività.

A sinistra vediamo comparire l'elenco degli utenti che ci hanno scritto e la gente che si propone per l'attività con tanto di bottone per iniziare una chat, bottone per accettare il lavoratore e data in cui ha effettuato la richiesta.

A destra vediamo come gli utenti possono vedere quante richieste ci sono attive al momento ed è comparsa la possibilità di annullare l'iscrizione, valido soltanto finché non si viene assegnati.


  

Special effects artist

Corrupti dolore est illum alias ipsam incidunt consectetur. Sapiente perferendis repudiandae voluptate earum excepturi aperiam.

When: 18/07/23 from 00:17 to 23:47


Requester

 ANTONIO GAGLIARDI - 4.25 ★

Free

OPEN RESERVED IN PROGRESS DONE CLOSED


User submissions

 NATALIA NIBALI - 3.00 ★


UNSUBMIT **CHAT WITH A.GAGLIARDI**

When: 18/07/23 from 00:17 to 23:47

Requester

 ANTONIO GAGLIARDI - 4.25 ★


Assigned to

 NATALIA NIBALI - 3.00 ★

Free

OPEN RESERVED IN PROGRESS DONE CLOSED


Open chats

 NATALIA NIBALI - 3.00 ★


Una volta accettato un utente questo fa progredire lo stato bloccando la possibilità ad altri utenti di registrarsi a questo Knock-Knock.

When: 18/07/23 from 00:17 to 23:47

Requester

 ANTONIO GAGLIARDI - 4.25 ★

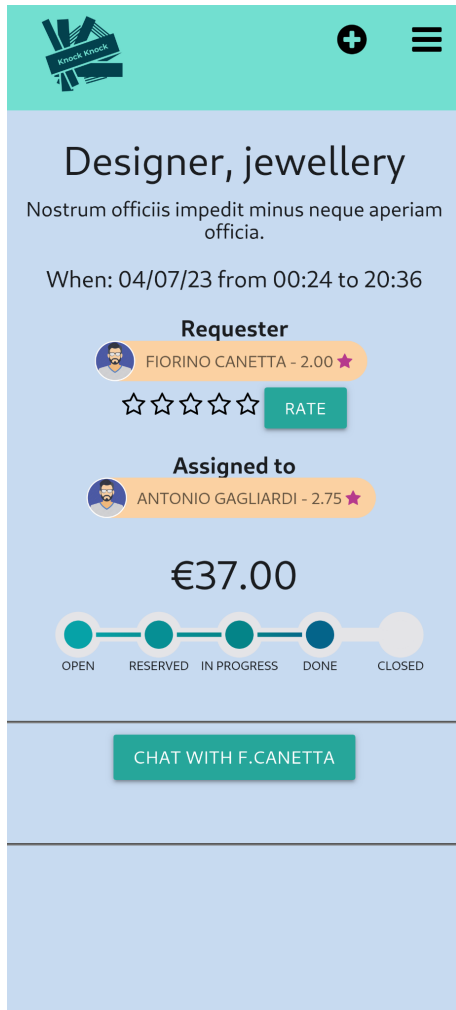
Assigned to

 NATALIA NIBALI - 3.00 ★

Free

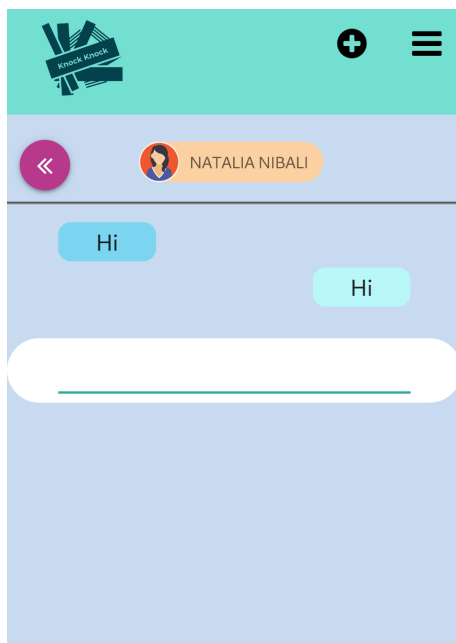
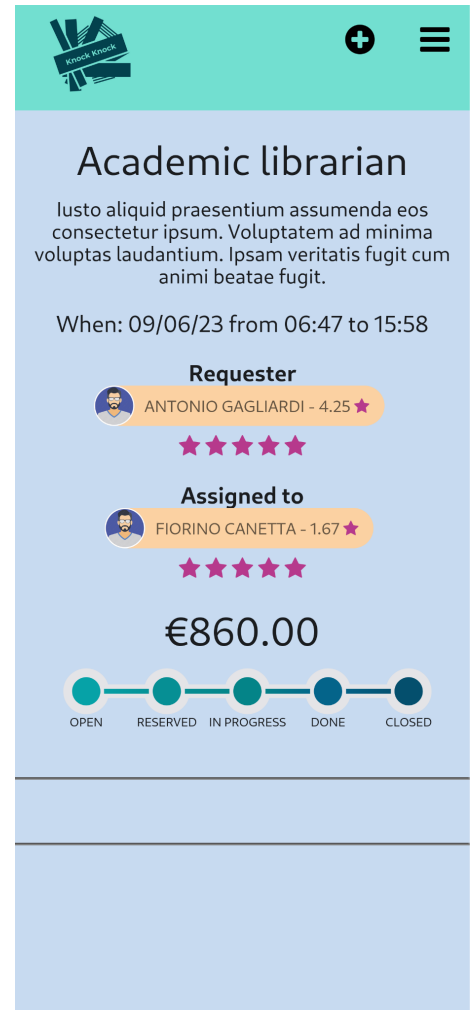
OPEN RESERVED IN PROGRESS DONE CLOSED

CHAT WITH A.GAGLIARDI



Una volta completato l'evento è possibile dare un voto all'altro utente e concordare le ultime attività tramite la chat.

A destra invece possiamo vedere un evento concluso in cui entrambi gli utenti hanno votato.



Infine mostro un esempio di chat in quanto rilevante per il progetto poiché la particolarità nella creazione di questa chat è stata la poca adozione di testi nella parte grafica.

Infatti è stato fatto uso principale dell'intuizione e delle abitudini degli utenti per capirne l'uso.

E' stato adottato un sistema di colori ma soprattutto di posizioni per indicare i messaggi scritti (a destra) ed i messaggi ricevuti (a sinistra) intuitivi come le altre applicazioni chat presenti attualmente.

Infine l'uso sempre in basso di una barra di inserimento per far intendere che è il form dove inserire il testo da mandare.

Problemi

Come possiamo velocizzare il calcolo del punteggio di un utente?

Per velocizzare il calcolo del punteggio medio di un utente, è stata adottata una soluzione basata su segnali (signals) che scattano ad ogni nuovo voto. Questo approccio evita il calcolo del punteggio medio ad ogni richiesta dell'utente, riducendo così il carico computazionale.

La soluzione prevede quanto segue:

1. Alla ricezione di un nuovo voto, viene generato un segnale che avvia la funzione di calcolo del nuovo punteggio medio dell'utente.
2. La funzione di calcolo del punteggio medio prende in considerazione tutti i voti ricevuti dall'utente e calcola il nuovo punteggio medio in base a questi dati.
3. Il nuovo punteggio medio viene quindi salvato nel profilo dell'utente, sostituendo il valore precedente.
4. Da questo momento in poi, il punteggio medio dell'utente sarà aggiornato e disponibile senza la necessità di ricalcolarlo ad ogni richiesta.

Questo approccio riduce significativamente il carico computazionale poiché il calcolo del punteggio medio viene eseguito solo quando si riceve un nuovo voto. Gli accessi ai profili utente o altre operazioni non richiederanno più il calcolo del punteggio medio, ma accederanno direttamente al valore salvato nel profilo utente.

Sfruttando i segnali per inizializzare il calcolo del punteggio medio solo quando necessario, si migliora l'efficienza e la velocità complessiva del sistema.

Come possiamo gestire le date avendo comunque un'interfaccia web gradevole?

Per garantire un'interfaccia web gradevole, abbiamo affrontato la sfida di gestire le date in modo flessibile e intuitivo. Tuttavia, abbiamo riscontrato diversi problemi in diverse situazioni.

Uno dei problemi principali riguardava la ricerca delle attività. Se volevamo supportare date e orari diversi, il metodo di ricerca avrebbe dovuto essere altrettanto complesso. Ad esempio, se un utente cercava eventi per la settimana corrente, era considerata valida un'attività che iniziava questa settimana ma finiva la settimana successiva? Inoltre, se un utente cercava un giorno specifico, era considerata valida un'attività che finiva proprio quel

giorno?

Un altro aspetto da considerare era il limite massimo per la durata di un'attività. Volevamo evitare situazioni in cui gli utenti richiedessero lavori "part-time" che non fossero allineati con l'obiettivo della piattaforma. Abbiamo quindi deciso di impostare il limite massimo alla durata di una giornata, semplificando così l'utilizzo del sito e mantenendo l'idea di una webapp per attività semplici.

Siamo consapevoli che queste decisioni potrebbero non coprire tutte le possibili casistiche, ma abbiamo cercato di trovare un equilibrio tra la flessibilità delle date e la facilità d'uso dell'interfaccia.

Come possiamo ricercare eventi vicino?

Per implementare la ricerca di eventi vicini nel contesto del progetto, si è valutata l'integrazione delle estensioni GIS di PostgreSQL, che offrono funzionalità avanzate per la geolocalizzazione. Tuttavia, a causa delle attuali limitazioni e della mancanza di Docker come tecnologia, non è stato possibile integrare completamente queste estensioni nel progetto.

L'idea principale era quella di consentire agli utenti di cercare eventi in base alla loro posizione geografica, utilizzando moduli per la ricerca di città e indirizzi durante la creazione dei nuovi Knock-Knock e nella fase di ricerca. Questo avrebbe permesso di utilizzare ricerche e ordinamenti basati sulla vicinanza geografica. Questa funzionalità sarebbe stata molto utile per gli utenti finali.

Come avviene il pagamento tra gli utenti?

La gestione degli stati dei Knock-Knock è stata una sfida complessa, in quanto ha introdotto diversi casi d'uso e worst case scenario, con il problema comune di potenziali utenti malintenzionati che possono comportarsi in modo scorretto senza conseguenze dirette sul sito.

L'introduzione di queste sfide rende il sistema di pagamento un'importante componente necessaria per porre un vincolo di "potere" sugli utenti malintenzionati, consentendo agli amministratori di disabilitare i loro account.

Tuttavia, l'implementazione del sistema di pagamento all'interno del progetto ha affrontato due principali ostacoli. Inizialmente, uno dei requisiti del progetto era la disponibilità offline, che sarebbe stata compromessa da un sistema di pagamento esterno. In secondo luogo, l'integrazione con vari sistemi di pagamento avrebbe richiesto lo sviluppo di un middleware specifico, che richiederebbe ulteriore tempo di sviluppo.

A causa di queste sfide, il sistema di pagamento non è stato valutato come parte integrante dell'applicazione. La sua implementazione all'interno dell'applicazione avrebbe comportato gravi problematiche di sicurezza in caso di utenti malintenzionati e potenziali implicazioni legali.