

Queueify

2022

Lorenzo Rossi

Matr. 142422

Unimore FIM, Informatica

Progetto di Tecnologie Web

Indice

Indice	1
Introduzione	2
Requisiti	3
Descrizione progetto	4
Utente	4
Code	4
Ticket	4
Schema UML	6
Database	7
Scelte di sviluppo	11
Organizzazione	11
AJAX	11
Gestione delle dipendenze	11
Database di test	11
Test	12
Queue	12
Users	12
Risultati	13
Problemi riscontrati	16

Introduzione

Queuify è un'applicazione web che mira a rimpiazzare le code tradizionali per diminuire i tempi di attesa tramite l'uso di "ticket", prenotazioni che si possono creare in precedenza alla data d'uso.

L'applicazione offre strumenti per visualizzare, prenotare e gestire diverse tipologie di code, insieme agli strumenti amministrativi per la moderazione.

Requisiti

L'applicazione dev'essere in grado di:

1. Visualizzare le code disponibili anche senza registrazione
2. Creare e gestire profili personali, con foto e descrizione
3. Gestire amicizie
4. Creare code
 - a. Più utenti possono gestire delle code, con distinzione tra amministratori ed impiegati
 - b. Le code possono essere di diverso tipo: pubbliche, a condivisione di link, ad invito o solo per amici
 - c. Devono avere orari di apertura-chiusura con eventuali eccezioni
 - d. Devono poter essere "confidenziali" (ex. medico)
5. Prenotazioni
 - a. Possibilità di prenotare e cancellare un ticket
 - b. Possibilità di avere slot precisi di tempo
 - c. Notifica quando un ticket è prossimo in coda
6. Moderazione
 - a. Possibilità di segnalare code
 - b. Possibilità di review delle segnalazioni da parte di un gruppo di moderatori
 - c. Visualizzazione di statistiche riguardo alla coda
7. Ricerca di code ed utenti

Descrizione progetto

Il sistema deve gestire i seguenti aspetti:

Utente

Gli utenti sono gli attori del programma, loro possono creare code, gestire le code create o a cui lavorano, segnalare code, gestire le amicizie e gli inviti, prenotare ticket e gestire i ticket prenotati.

Inoltre alcuni utenti, chiamati moderatori, sono in grado di vedere le segnalazioni e decidere se sono o meno importanti, nel caso in cui lo fossero possono segnalarlo a degli amministratori per farle rimuovere.

Code

Le code sono il cuore di Queueify, rappresentano la coda fisica che si farebbe, per esempio ad un negozio, dal dottore o ad un evento.


Esse possono essere create da chiunque ed hanno quattro modalità di visualizzazione: pubblica, a condivisione di link (stile google drive), sono per amici o ad invito. Le code confidenziali inoltre non usano i dati riguardanti alle prenotazioni per suggerimenti.

Le code possono avere un orario di apertura diverso per ogni giorno della settimana e devono anche avere delle "eccezioni" per giorni in cui la coda non può operare o opera al di fuori degli orari standard.

In alcuni ambiti è necessario avere degli slot di tempo precisi da poter prenotare, si pensi ad esempio a tutto l'ambito medico. Nel caso in cui tale configurazione sia attiva le prenotazioni potranno essere effettuate solamente in slot vuoti.

Ticket

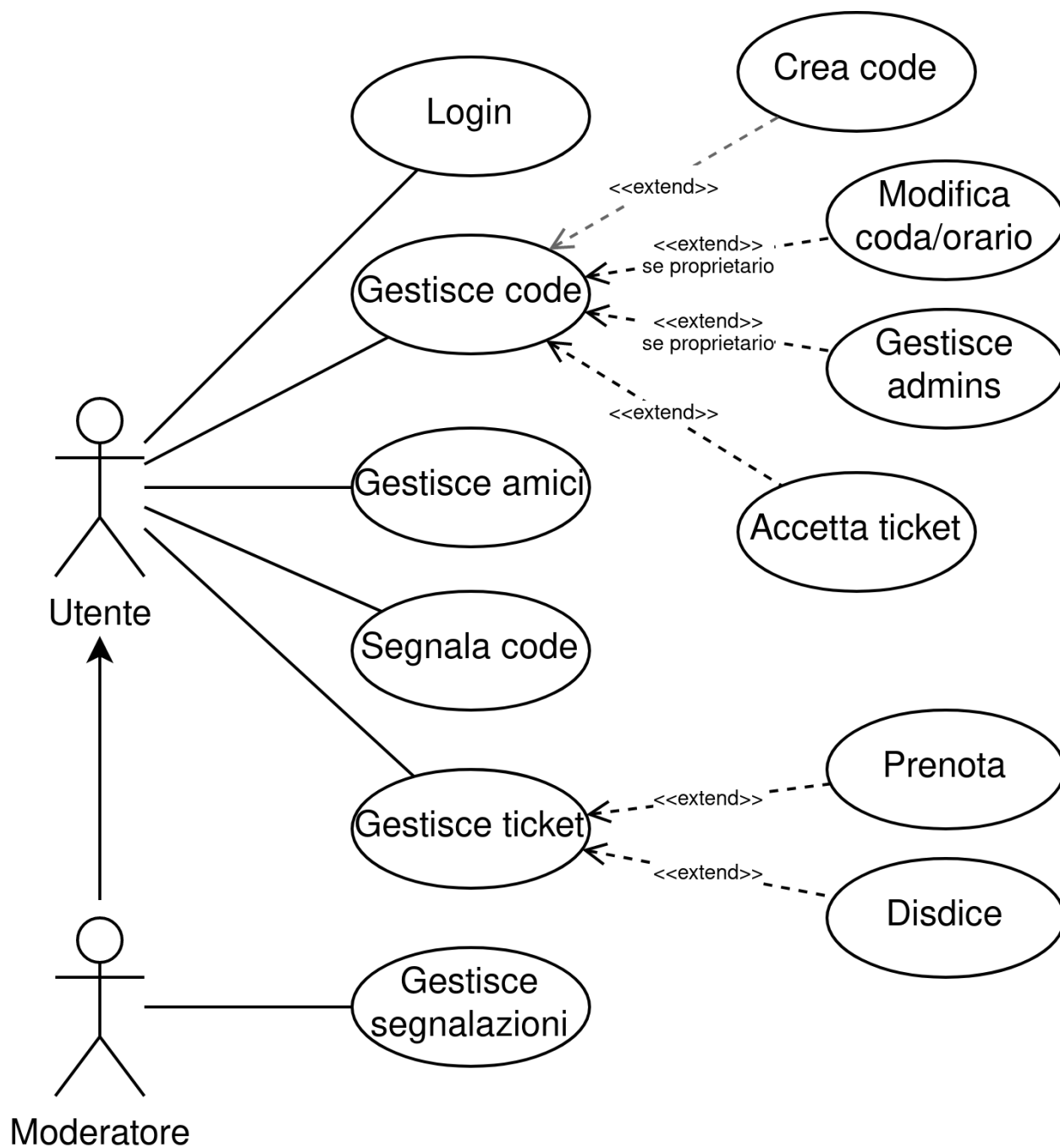
I ticket rappresentano una prenotazione nella coda rispetto ad un momento preciso nel tempo, quello preferito dall'utente.



Un ticket è disdicibile da parte dell'utente a cui appartiene o dagli amministratori della coda, in tal caso il ticket non verrà eliminato ma segnato come disdetto per mantenere uno storico dei biglietti.

Quando un ticket è il prossimo ad essere gestito in una coda, verrà inviata una notifica al suo possessore.

Schema UML



Database

users_user	
password	varchar(128)
last_login	datetime
is_superuser	bool
username	varchar(150)
first_name	varchar(150)
last_name	varchar(150)
email	varchar(254)
is_staff	bool
is_active	bool
date_joined	datetime
description	varchar(1024)
image	varchar(100)
tz	varchar(63)
pronouns	varchar(32)
id	integer

L'utente rimpiazza la classe standard di Django aggiungendole una descrizione, l'immagine di profilo, una timezone, ed i pronomi.

Quest'ultimo è un campo libero e può essere usato con i pronomi desiderati per le lingue anglofone ("he/him", "she/her", "they/them"...) e anche in italiano ("maschili", "femminili").

Si hanno inoltre altre due tabelle legate strettamente all'utente riguardanti alle amicizie.

users_user_friend_requests	
from_user_id	bigint
to_user_id	bigint
id	integer

Si nota che se l'utente A è amico di B allora sarà salvata anche la relazione inversa (B -> A) per semplicità nell'uso dell'ORM Django.

users_userfriend	
user_from_id	bigint
user_to_id	bigint
id	integer

queues_qqueue	
name	varchar(50)
description	varchar(1024)
image	varchar(100)
is_privacy_hidden	bool
join_mode	varchar(3)
expected_time_per_ticket	real
ticket_stats_count	integer
fixed_ticket_time_minutes	integer
tz	varchar(63)
id	char(32)

La coda¹ ha diversi campi interessanti, innanzitutto ha una timezone che rappresenta i suoi orari (cioè dove opera), la **join_mode** che può essere 'PUB' (pubblica), 'URL' (a condivisione di link), 'FRI' (solo per amici), 'INV' (ad invito).

Per supportare le code a condivisione di link si usa un UUID generico come id del modello.

Ha inoltre alcuni campi che racchiudono statistiche riguardanti alle persone servite (`expected_time_per_ticket`,

`ticket_stats_count`).

QueueUser salva i proprietari, dipendenti e gli invitati ad una coda, a seconda del campo 'role'

queues_queueopenrange	
day	smallint unsigned
from_time	time
to_time	time
queue_id	char(32)
id	integer

QueueOpenException invece rappresenta un'eccezione per un giorno specifico.

queues_queueuser	
role	varchar(3)
queue_id	char(32)
user_id	bigint
id	integer

QueueOpenRange rappresenta gli orari d'apertura di una coda per un singolo giorno della settimana (0-6).

queues_queueopenexception	
day	date
from_time	time
to_time	time
queue_id	char(32)
id	integer

¹ La coda è stata chiamata QQueue per evitare conflitti involontari con queue.Queue, la coda FIFO di python.

I ticket uniscono utenti e code, requested_time è l'orario che l'utente preferisce, oppure l'inizio dello slot prenotato nel caso in cui la coda sia a slot temporanei. cancel_message è popolato solamente se il ticket viene cancellato da un operatore della coda o da un utente. closure_time e wait_time_secs sono invece popolati quando il ticket viene servito correttamente.

queues_userqueuereport	
creation_time	datetime
reason	varchar(256)
reviewed	bool
queue_id	char(32)
user_id	bigint
id	integer

queues_ticket	
creation_time	datetime
state	varchar(3)
closure_time	datetime
wait_time_secs	integer
cancel_message	varchar(128)
queue_id	char(32)
user_id	bigint
requested_time	datetime
id	integer

UserQueueReport gestisce le segnalazioni delle code da parte degli utenti, il campo reviewed diventa true solamente quando un moderatore controlla i report.

Nella pagina seguente è riportato lo schema completo, senza tabelle generate automaticamente da django o delle librerie esterne.



Scelte di sviluppo

Organizzazione

Il progetto è stato diviso in tre applicazioni: queues, users e core. Core contiene solamente il comando per la creazione del database di prova, users contiene la gestione degli utenti, profili ed amicizie mentre queues contiene la parte principale dell'applicazione: la gestione delle code, dei ticket, delle statistiche e dei report.

AJAX

È stato usato AJAX in due punti del progetto, il primo punto è durante le prenotazioni, per avere un'interazione più veloce con l'utente riguardo agli orari di apertura per i diversi giorni. Il secondo è per la visualizzazione delle statistiche, dato che un'interazione dinamica sembra più naturale per un caso d'uso simile.

Gestione delle dipendenze

È stato preferito l'uso di Poetry rispetto a pipenv per questioni di performance, in ogni caso è stato fornito file Docker quindi non è necessario installare Poetry per avviare il progetto.

Database di test

Per aiutare lo sviluppo ad iterazioni rapide è stato creato un generatore di database di prova, usando il comando resetdb (eseguito automaticamente se si usa docker) il database verrà popolato con alcuni utenti d'esempio, delle queue ed un mese di dati antecedenti, utili per guardare le statistiche generate.

Questa scelta pur avendo un costo iniziale alto ha portato ad un notevole aumento della velocità di progettazione e di testing manuale.

Test

In questo capitolo si presentano e descrivono i test fatti sul progetto.

Queue

Lo sviluppo di alcune feature più complesse delle code è stato realizzato tramite test-driven development, questi però sono stati fatti direttamente sui modelli, non sulle loro view.

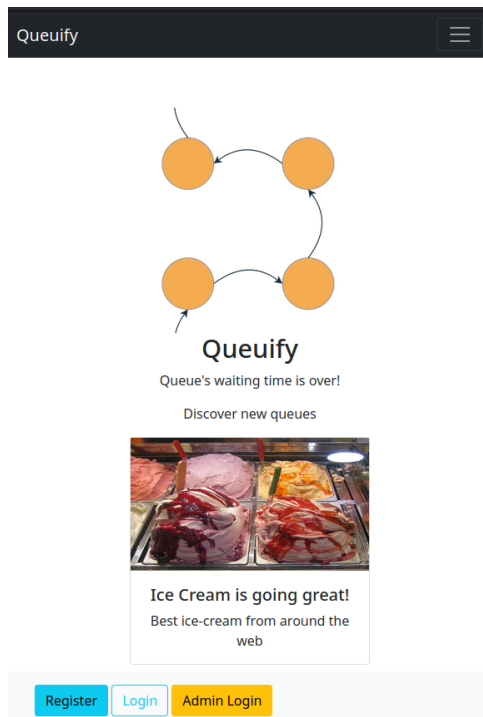
- **test_is_open_at**: controlla la funzione `is_open_at` delle queue per sapere se una queue è aperta in un dato punto nel tempo
- **test_visibility**: controlla la visibilità di diversi tipi di coda dal punto di vista di diversi utenti (ex.. code ad invito o solo ad amici).
- **test_queue_fixed_time_booking**: controlla uno degli aspetti più complessi della coda, la prenotazione a slot di tempo fissi

Users

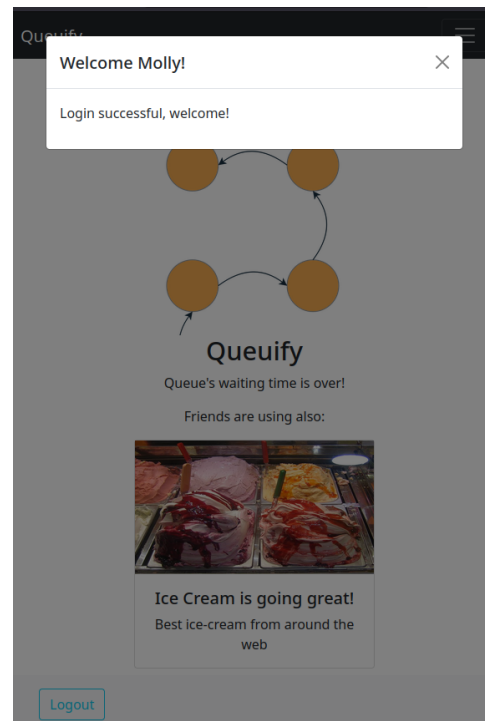
A lato utenti invece sono state testate le view esposte.

- **test_search**: controlla la ricerca di persone
- **test_make_request**: controlla il chiedere l'amicizia ad un'altro utente
- **test_accept_request**: controlla l'accettare una richiesta di amicizia
- **test_re_request_friendship**: controlla che non si possa richiedere l'amicizia a qualcuno che ha già una richiesta dallo stesso utente
- **test_request_friendship_to_friend**: controlla che non si possa richiedere l'amicizia ad un amico

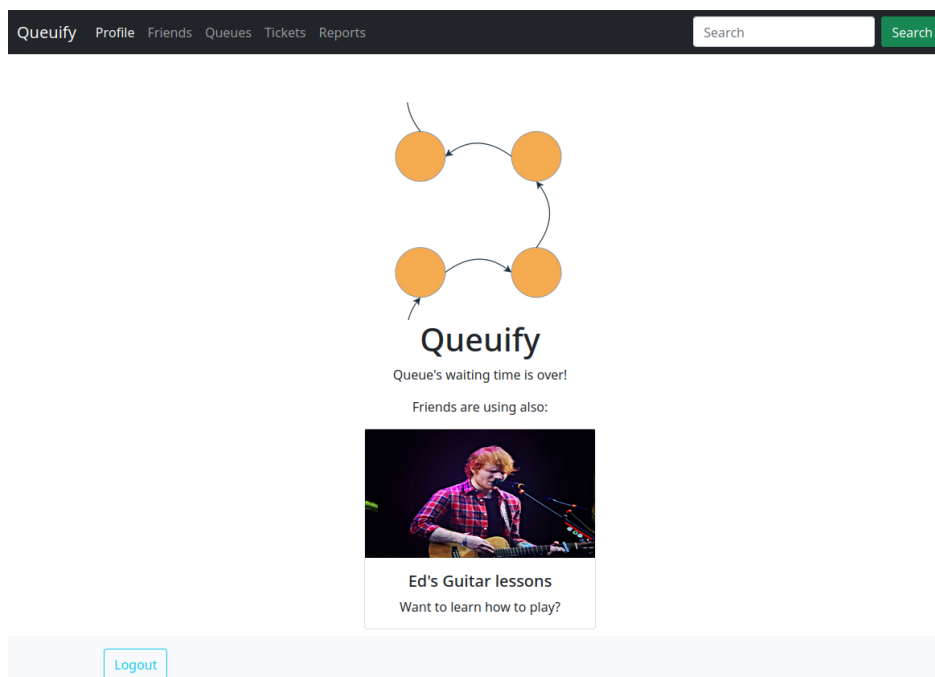
Risultati



Homepage anonima



Homepage (dopo il login)



Pagina principale
Desktop da parte
di un moderatore
(notare i Reports)

View d'utente di una coda pubblica



Ice Cream is going great!

Best ice-cream from around the web

Public

Book! Report

Today: 14:00 to 20:00

Day	Range
Monday	14:00-20:00
Tuesday	Closed
Wednesday	Closed
Thursday	Closed
Friday	14:00-20:00
Saturday	14:00-20:00
Sunday	14:00-20:00

Expected wait time: 3m54s (by 126 tickets)
Also used by: Margaret, Linus

Your queues



Ed's Guitar lessons

Want to learn how to play?

Used recently



Local Plumber

Is a pipe slow ehm leaking? Your ol' Linus can patch it



Ice Cream is going great!

Best ice-cream from around the web

New!
Create a new queue

Pagina di lista delle proprie code



Pagina di statistiche di una coda

Day

07/12/2022

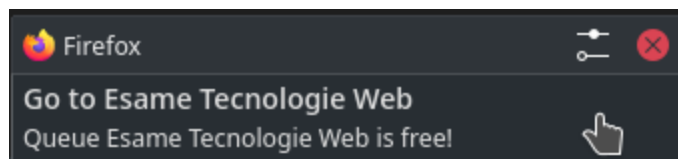
Available times

14:00-15:00

Book!

Pagina di prenotazione code

Notifica generata quando l'utente è
prossimo nella coda



Problemi riscontrati

Non sono stati riscontrati grossi problemi di sviluppo, la maggior parte dei conflitti sono dovuti alla scarsa documentazione delle librerie e alla tipizzazione dinamica di Python che non permette di fare alcun controllo statico del codice.

Un problema riscontrato e non risolto è stato il setup di un database con funzionalità di ricerca posizionale, dato che era una feature opzionale si è optato per escludere la feature della ricerca per locazione, dando spazio ad altre feature più semplici.

Una feature che richiederebbe molti più test è il supporto completo delle timezone, ma data la sua complessità non è stato possibile eseguire testing opportuno in tempi brevi.

È stato interessante scoprire i limiti dell'ORM di Django implementando le statistiche, dato che le feature richieste ricadono fuori dal suo scopo di utilizzo.