Progetto Sviluppo Applicazioni WEB

Studente: Torchia Matteo

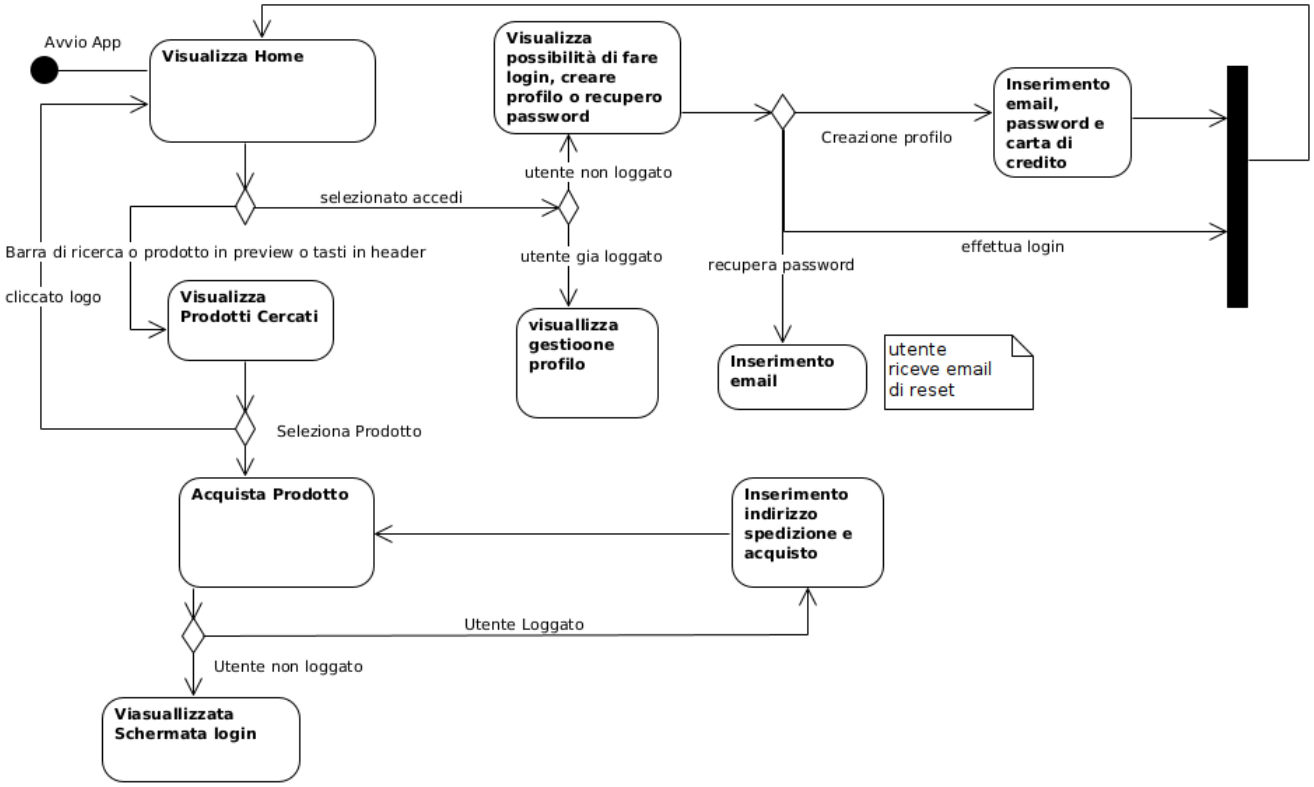
Matricola: 599899

Introduzione

My-Ecommerce

L’ app consiste in una semplice implementazione di un e-commerce di vestiti, non sono implementate tutte le caratteristiche tipiche di un app professionale. Una volta avviata, l’utente visualizzerà una schermata home contenente una piccola preview di alcuni prodotti e i pulsanti necessari per la navigazione, fra questi si ha la possibilità di selezionare i prodotti riguardanti uomo, donna o bambini, la possibilità di cercare i prodotti attraverso l’apposita barra di ricerca e di effettuare l’accesso, quest’ultimo necessario per l’acquisto di un prodotto.

Flusso di esecuzione dell'App Generale



Specifiche

L'architettura dell'app è composta da un client implementato in React e da un service worker necessario per rendere l'app una Progressive Web App, non è stato implementato alcun programma server in quanto la logica necessaria per processare i dati richiesti non sfrutta troppe risorse di calcolo.

La maggior parte delle operazioni effettuabili su My-Ecommerce si poggia sui servizi offerti da Firebase, fra questi vengono utilizzati: il servizio di autenticazione tramite e-mail e password, il servizio di Firebase storage per il

salvataggio delle immagini visualizzate dall'utente, il servizio di recupero password e il servizio di cloud storage.

Le raccolte presenti all'interno del cloud storage sono le seguenti:

1) cards: contiene all'interno i documenti che rappresentano le carte di credito caricate dall'utente nel momento della creazione del profilo o in seguito ad un'aggiunta successiva, l'id associato ad ogni documento è l'indirizzo e-mail dell'utente utilizzato per l'iscrizione e ogni documento in fine contiene un array di oggetti che a sua volta contengono i dati cifrati delle carte di credito.
2) dizionario: documento contenente un array di parole, tali parole vengono utilizzate dall'algoritmo di correzione dell'input in caso di errore di battitura quando si usa la barra di ricerca.
3) gender: contiene un array con le parole uomo, donna e bambino, queste stringhe vengono utilizzate dall'algoritmo di ricerca per effettuare il controllo su cosa l'utente ha inserito nella barra di ricerca e per effettuare un query corretta a Firestore.
4) maglie, pantaloni, scarpe: contengono i dati relativi ai prodotti venduti, ogni collezione contiene diversi documenti e ognuno di essi mantiene le informazioni sul prodotto come, ad esempio, la disponibilità del prodotto o la sua descrizione.
5) product: analogamente a gender contiene un array di stringhe, in questo caso però sono contenute le parole uomo, donna e bambino, anche in questo caso queste informazioni vengono usate per effettuare per implementare l'algoritmo di ricerca.
6) preview: mantiene salvate le url delle immagini da visualizzare presenti su Firebase Storage.
7) orders: viene utilizzata per mantenere gli ordini effettuati dagli utenti in passato, tali dati vengono utilizzati all'interno dell'opzione visualizza ordini nella sezione di gestione profilo.

Service worker

Nella fase di installazione del service worker effettuo pre-cacheing per inserire in cache il file index.html e la root del mio progetto, una volta completata la fase di pre-cacheing si passa alla fase di attivazione che segue la strategia di

cache versioning, con questa politica prima di attivare il service worker viene controllata la cache attuale, i dati al suo interno vengono eliminati se il nome assegnato alle cache presenti non viene trovato nell'array formato dai nomi delle cache, per far si che questa strategia funzioni correttamente ad ogni modifica di un file o comunque quando i dati in cache diventano obsoleti è necessario aggiornare i nomi della cache all'interno del service worker.

Completata la fase precedente, il service worker si mette in attesa di richieste dall'app web, ad ogni richiesta che non sia una richiesta post o una richiesta riguardante alcuni servizi di Firebase come, ad esempio, l'autenticazione o la creazione del profilo, il service worker prima controlla se i dati richiesti sono contenuti in cache e in tal caso restituisce i dati all'app, altrimenti la richiesta viene inoltrata in rete e prima di ritornare la risposta al client inserisce i dati in cache.

Flusso di esecuzione interna

All' avvio dell'app all'interno dei file index.js le prime operazioni che vengono fatte sono l'istallazione del service worker, inizializzare Firebase per la gestione del caching per l'utilizzo dell'app quando si è offline e la richiesta all'utente per l'invio di notifiche dall'app.

Superata questa fase preliminare viene renderizzato il componente app, all'interno di app viene effettuato il routing principale dell'progetto e il modulo che viene caricato per primo è Home, Home è composto da un footer in cui sono inserite informazioni generali per simulare un vero e-commerce, un componente che visualizza la preview dei prodotti e il componente MyHeader, quest'ultimo viene utilizzato per quasi tutta la navigazione all'interno del progetto.

In base all'operazioni che l'utente decide di effettuare viene utilizzato l'hook navigate per navigare nell'app.

Riguardo le operazioni di gestione del profilo dell'utente, in particolare per l'aggiunta della carta di credito, il modulo per visualizzare la form dell'inserimento dei dati della carta viene renderizzato in due moduli diversi, nella fase di creazione del profilo, e nel caso l'utente voglia aggiungere un'ulteriore carta di credito, in entrambi i casi si utilizza il modulo CreateAccount. Per poter gestire la navigazione all'indietro correttamente e

poter gestire il corretto funzionamento delle chiamate a FireBase nel modulo CreateAccount viene passato una props (onlyCardForm), quest'ultima permette il corretto rendering della form e il corretto uso delle API di Firebase in base al punto di navigazione da cui viene raggiunto.

Le operazioni di acquisto di un prodotto o di gestione del profilo possono essere effettuate soltanto se l'utente ha effettuato il login, quest'ultimo viene gestito internamente all'app per quanto riguarda la visualizzazione dell'utente, e sul server di Firestore abilitando la lettura e la scrittura delle carte di credito solo in caso di utente loggato:

```
service cloud.firestore {
  match /databases/{database}/documents {
        match /cards/{doc} {
              allow read: if request.auth != null;
              allow write: if request.auth != null;
        }
        match /orders/{doc} {
              allow read: if request.auth != null;
              allow write: if request.auth != null;
        }
    }
}
```

Compilazione e avvio

Versioni software usate

npm : 10.4.0

Node : v18.17.1

Per istallare le versioni corrette:

Digitare nel terminale:

1) nvm install 18.17.1
2) nvm use 18.17.1
3) npm install -g npm@10.4.0

Una volta scaricato il progetto, posizionarsi all'interno della cartella principale del progetto ed eseguire i seguenti comandi:

npm install <-- per installare i pacchetti necessari

npm start <-- per avviare l'app

Link repository

https://github.com/TMaTTeO99/SAW_E_COMMERCE/tree/master

Il browser utilizzato per testare l'app è Google Chrome