1 Introduzione

Il progetto consiste nella realizzazione di due web app per la gestione di un sito di e-commerce. Le funzionalità comprendono la ricerca e la selezione per categorie e/o tipologia dei prodotti, la gestione del profilo utente (dati personali, informazioni per le spedizioni e visualizzazione degli ordini) e gestione del carrello (solo se l'utente è autenticato). La presenza di due applicazioni ha lo scopo di separare il *lato client* dal *lato server* del progetto complessivo. In questo modo, ricordando il modello MVC di una generica applicazione web, si hanno 3 parti:

- **Model:** si occupa della gestione dei dati dell'applicazione, dunque effettua le operazioni di ricerca e modifica sul database e gestisce l'autenticazione degli utenti. In questo progetto, il model è definito nell'applicazione lato server, insieme alle funzioni per la comunicazione con il database.
- **View:** si occupa della rappresentazione visuale dei dati, ossia la vera e propria interfaccia grafica che permette all'utente effettuare le varie operazioni. In particolare, nell'applicazione lato client è presente una cartella contenente tutte le pagine, che vengono poi renderizzate insieme ai dati.
- **Controller:** concettualmente fa da tramite tra model e view. Ha il compito di gestire le richieste effettuate dall'utente comunicando con il model per il retrieve dei dati e renderizzando tali risultati tramite le view. L'applicazione lato client, come quella lato server, prevede la gestione delle richieste definendo degli express routers, uno per ogni pagina del sito. Questi si occuperanno delle richieste da fare all'applicazione lato server e, una volta ottenuti i dati, del rendering delle pagine stesse.

1.1 Tecnologie Utilizzate

Il linguaggio di programmazione principale utilizzato sia per il lato client che per il lato server è il Javascript. Le due applicazioni utilizzano il runtime Node.js ed il framework Express. Per la realizzazione della struttura e dell'interfaccia grafica responsive delle pagine sono stati utilizzati i linguaggi HTML, CSS ed il framework Semantic UI. Tutte le animazioni delle pagine sono state gestite utilizzando la libreria JQuery. Il rendering delle pagine viene gestito dal linguaggio per il templating EJS. Per quanto riguarda il lato server, per il database viene utilizzato MongoDB Atlas, mentre l'autenticazione viene effettuata usando jwt (json web token). Il package manager utilizzato per la gestione delle dipendenze del progetto è npm.

1.2 Procedura di avvio dell'applicazione

Per avviare le due applicazioni è necessario eseguire il comando `npm install` sia nella root directory del progetto (applicazione lato server), sia nella cartella `/client` (applicazione lato client). Una volta installate tutte le dipendenze è necessario eseguire il comando `npm start` in entrambi i percorsi. Infine, una volta che le applicazioni sono in esecuzione, basterà andare alla pagina `http://localhost:5000` per raggiungere il sito.

2 Pagine

2.1 Home Page

Questa è la pagina principale, nella parte superiore è presente un menù che consente la navigazione tra le varie sezioni e pagine del sito. Dopo aver effettuato il login compariranno i pulsanti per accedere alla pagina del profilo, al carrello e per effettuare il logout. Sono inoltre presenti una barra di ricerca dei prodotti disponibili (è disponibile solo la ricerca per nome esatto, maiuscole comprese), uno slideshow con 3 immagini, navigabile dall'utente e con scorrimento automatico e i 9 prodotti più comprati. Questi ultimi sono visualizzati secondo una griglia, dove ogni elemento è provvisto di nome, categoria, tipo, descrizione a comparsa, prezzo e disponibilità. Infine, nel footer di ogni pagina è presente il collegamento denominato "About us" che consente la visualizzazione della pagina di informazioni.

2.2 Shop Page

In questa pagina è presente sempre il menù superiore con le stesse funzionalità descritte in precedenza, insieme alla barra di ricerca. Lateralmente è presente un menu verticale che consente di scegliere categoria e tipo di prodotto da visualizzare nella pagina. I prodotti ora sono disposti orizzontalmente e consentono la visualizzazione più comoda delle informazioni del prodotto. Tramite la scritta "Learn more" è possibile espandere la descrizione del prodotto

e di visualizzarne maggiori dettagli. La pressione del bottone "Add to cart" permette ad un utente autenticato di aggiungere al carrello i prodotti desiderati.

2.3 Shopping Cart Page

In questa pagina è sempre presente il menù superiore con disponibile il collegamento al profilo (con relativo pulsante di logout). In questa pagina vengono visualizzati i prodotti aggiunti al carrello dello specifico utente. Inoltre è possibile procedere all'ordine solo se sono presenti le informazioni di spedizione. Tramite gli appositi pulsanti è possibile modificare le quantità dei prodotti oppure eliminarli dal carrello.

2.4 User Profile Page

In questa pagina è sempre presente il menù superiore con il collegamento al profilo e al carrello. I campi presenti consentono di visualizzare e modificare i dati personali, le informazioni di spedizione, la password e di visualizzare gli ordini effettuati. Alla registrazione di un nuovo utente si viene reindirizzati a questa pagina e si viene invitati a inserire i dati personali. Per procedere alla modifica dei dati è necessario fare uso del pulsante "Edit". È presente la validazione dei campi che impedisce di inviare al server dati non compatibili con il formato del campo. Eventuali messaggi di errore della validazione dei campi vengono visualizzati sotto la tabella.

2.5 About Page

Lo scopo di questa pagina è quello di fornire delle informazioni riguardo al sito, all'azienda e agli eventuali dipendenti dell'azienda. Inoltre, è presente una mappa (Google Maps API) per conoscere la posizione fisica dell'azienda, insieme ad una lista di contatti. Siccome la pagina non presenta altre funzionalità, presenta il solito menù superiore, ma senza i collegamenti alle pagine di profilo o del carrello.

3 Funzionamento

3.1 Applicazione Lato Server

3.1.1 Database

Il servizio Atlas di MongoDB permette l'utilizzo di un database NoSQL orientato ai documenti. Questo significa che il database corrisponde ad un insieme di collezioni, dove ogni collezione corrisponde ad un insieme di documenti. Questi documenti sono in formato JSON, nei quali i dati sono organizzati secondo una struttura campo-valore. La libreria utilizzata per interfacciarsi con MongoDB è Mongoose. Questa permette di definire, per ogni documento, uno *schema*, ossia una struttura che il documento deve rispettare per poter essere salvato e manipolato nel database. Con *model* si fa riferimento al modello del documento, creato a partire dallo schema. Il documento vero e proprio viene creato tramite un'istanza del modello. In questa applicazione sono definiti due modelli chiamati `User` e `Product`, basati sugli schemi `UserSchema` e `ProductSchema`, nella directory `/models`. Per la operazioni di ricerca, modifica, eliminazione e salvataggio sono state utilizzate delle funzioni della libreria Mongoose quali: `findOne, findById, updateOne, findByIdAndUpdate, findByIdAndDelete` e `save`.

3.1.2 Autenticazione

Per quanto riguarda l'autenticazione degli utenti, vengono utilizzati i cosiddetti *json web token*. Questi sono dei token di autenticazione univoci creati appositamente per un utente quando effettua l'operazione di login. Di fatto il token corrisponde ad una stringa di caratteri formata da tre parti:

1. HEADER: corrisponde alla codifica Base64Url di un documento JSON che contiene due campi, uno per il tipo di algoritmo utilizzato per creare la signature e uno per indicare il tipo di token.

2. PAYLOAD: anche questa corrisponde ad un documento JSON codificato in Base64Url. Questa parte del token può contenere dei dati personalizzati come la data di creazione, un'eventuale data di scadenza del token, l'id dell'utente che sta effettuando l'operazione di login.

3. SIGNATURE DI VERIFICA: unica per ogni token, creata a partire da header, payload e da una chiave segreta utilizzando un algoritmo di hashing o di cifratura (specificato nello header). Questa signature permette di verificare l'autenticità del token (data la chiave segreta).

In particolare, nell'applicazione i token presentano uno header di default previsto dalla libreria *jsonwebtoken* (algoritmo HS256 e tipo JWT) ed un payload con l'id dell'utente che ha richiesto l'autenticazione e una scadenza del token dopo 14 giorni dalla creazione. Una volta creato il token, questo viene aggiunto al documento dell'utente nel database insieme al tipo di piattaforma in uso (computer con sistema windows, smartphone con sistema android, ecc). In questo modo, l'utente può rimanere autenticato su più dispositivi contemporaneamente. Infine, per la verifica dei token, è stato scritto un express middleware che ne controlla la validità e, nel caso in cui il token sia scaduto, si occupa anche di rimuoverlo dal documento dell'utente.

3.1.3 Express Routers

Sono stati realizzati cinque express routers, uno per ogni "aspetto" dell'applicazione. Si hanno i seguenti:

- `/auth`: gestione delle operazioni di registrazione, login e logout.
- `/users`: gestione delle operazioni di manipolazione delle informazioni dell'utente.
- `/products`: gestione delle operazioni di ricerca e modifica dei prodotti.
- `/orders`: gestione degli ordini dell'utente.
- `/cart`: gestione del carrello dell'utente.

3.1.4 Dipendenze

In aggiunta ad Express, Mongoose e jsonwebtoken, sono state usate ulteriori librerie quali *validator* per la validazione di alcuni campi (email, id), *bcryptjs* per la cifratura delle password e *express-useragent* per il parsing dello header "User-Agent" delle richieste.

3.2 Applicazione Lato Client

L'applicazione client viene inizializzata alla porta 5000. Il sito prevede varie funzionalità, gestite tramite degli express routers. Ognuno di questi si occupa delle richieste che l'utente fa interagendo con gli elementi della pagina. La gestione di queste routes consente il passaggio dei dati immessi nelle pagine all'applicazione server e la successiva rappresentazione dei risultati per mezzo delle views.

3.2.1 Dipendenze

Come già accennato in precedenza, per il rendering delle pagine viene usato EJS. Per l'interfaccia grafica viene usato il framework Semantic UI (CDN). In aggiunta, il sito prevede la notifica del risultato di alcune operazioni mediante dei messaggi (errore, successo o info) grazie all'utilizzo delle librerie *express-messages, connect-flash* e *express-session*. Per la lettura e la modifica dei cookies è stata utilizzata la libreria *cookie-parser*. Infine, per effettuare le richieste http all'applicazione server, è stata usata la libreria *request*.

3.2.2 Express Routers

Sono stati realizzati cinque express routers, ognuno che gestisce le richieste di una pagina del sito:

- `/auth`: gestione delle operazioni di registrazione, login e logout.
- `/profile`: gestione delle operazioni di modifica delle informazioni dell'utente e visualizzazione degli ordini.
- `/index`: gestione della visualizzazione dei prodotti della homepage.
- `/shop`: gestione della ricerca, visualizzazione prodotti e aggiunta al carrello.
- `/cart`: gestione del carrello dell'utente e creazione degli ordini (operazione di checkout).

3.2.3 Note

Per le immagini dei prodotti sono state utilizzate le *Lorem Picsum API*, che permettono di ottenere delle immagini casuali di dimensioni prefissate. *Possibili cause di mancato caricamento di tali immagini o di rallentamenti nella visualizzazione delle stesse sono pertanto da ricercare nel servizio in questione.*