

Simple Contacts

Applicazioni e Servizi Web

Matteo Venturi - 0001029389 {matteo.venturi7@studio.unibo.it}

25 Luglio 2022

0.1 Introduzione

Il progetto "Simple Contacts" vuole portare online la ben nota **rubrica** di contatti in chiave "**social**". L'obiettivo è quello di unire le funzionalità di una comune rubrica con alcuni semplici aspetti dei social. L'intera applicazione sarà protetta da autenticazione e offrirà gli strumenti base per poter gestire in toto una semplice rubrica di contatti. La peculiarità di questa rubrica sarà il fatto che gli utenti verranno avvisati qualora il loro contatto sarà aggiunto o visionato da altri utenti.

0.2 Requisiti

Per la stesura dei requisiti vengono principalmente prese in esame le rubriche già comunemente utilizzate. Fin da questa fase si decide di lavorare a stretto contatto con un rappresentante dell'utenza a cui è destinata l'applicazione. Il risultato che se ne ottiene è un elenco di requisiti direttamente ottenuti dall'esperienza dell'utente.

0.2.1 Requisiti utente

L'utente dell'applicazione si aspetta di poter:

- registrarsi all'applicazione
- accedere con le proprie credenziali all'applicazione
- modificare i dati del proprio profilo
- modificare la propria password
- aggiungere un contatto in rubrica inserendo nome, cognome, email, telefono, città, CAP, indirizzo del contatto
- modificare i dati di un contatto
- cancellare un contatto
- visualizzare i dati di un contatto
- cercare un contatto tramite ricerca testuale
- visionare dalla dashboard il numero totale di voci in rubrica

0.2.2 Requisiti funzionali

- registrazione utente con nome, cognome, email e password
- login utente con email e password
- modifica dati utente

- modifica della password di accesso
- creazione contatto contenente nome, cognome, email, telefono, città, CAP, indirizzo
- validazione dei dati inseriti durante la creazione del nuovo contatto
- aggiornamento dei dati di un contatto
- eliminazione di un contatto
- visualizzazione dei dettagli di un contatto
- visualizzazione di tutti i contatti contenuti in rubrica
- ricerca di un contatto tramite ricerca testuale libera
- statistiche in dashboard: numero totale di contatti in rubrica
- quando un contatto viene creato, notificare tale contatto (se esiste all'interno dell'applicazione ed è online)
- quando un contatto viene visualizzato, notificare tale contatto (se esiste all'interno dell'applicazione ed è online)

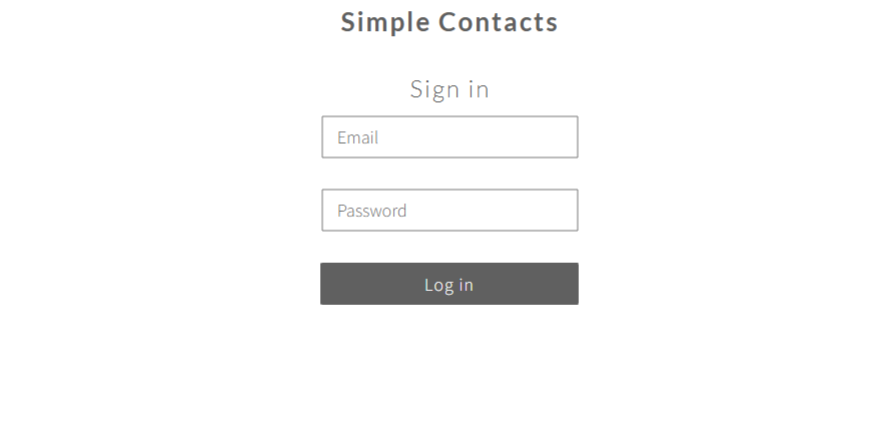
0.2.3 Requisiti non funzionali

- l'applicazione deve essere il più semplice possibile e permetterne l'utilizzo anche agli utenti meno esperti
- l'applicazione deve essere sicura e provvista di un meccanismo di autenticazione
- l'applicazione deve poter essere estendibile per future implementazioni e migliorie
- l'applicazione deve essere fruibile da dispositivi mobile oltre che da computer
- l'applicazione deve offrire un'esperienza reattiva

0.3 Design

0.3.1 Design dell'interfaccia utente

La fase di design dell'interfaccia utente viene svolta affiancandosi continuamente con un esponente dell'utenza target dell'applicazione, proponendo di volta in volta i **mockup** realizzati e simulando il comportamento dell'applicativo, raccogliendo feedback e suggerimenti dall'utente stesso. Al termine di questo processo vengono prodotti i mockup definitivi dell'applicazione e qui riportati.



The image shows a login form for an application titled "Simple Contacts". The form is centered and consists of the following elements: a title "Simple Contacts" in bold black text; a "Sign in" label in a light gray font; two input fields, one for "Email" and one for "Password", both with light gray borders and placeholder text; and a dark gray "Log in" button with white text. The entire form is set against a light gray background.

Figure 1: Login

In sintesi, dopo una fase di **login** (fig.1) o di **registrazione** (fig.2), l'utente viene portato all'interno dell'applicazione. Una volta fatto l'accesso all'applicazione all'utente viene mostrata la **dashboard** (fig.3) nella quale è possibile vedere alcune **statistiche** riguardanti la rubrica.

Tramite il menu laterale l'utente può navigare le sezioni dell'applicazione e quindi arrivare all'**elenco dei contatti** (fig.4) che si presenta come una lista di contatti con alcuni dati mostrati. In questa schermata l'utente può **ricercare** un contatto scrivendo nello spazio dedicato in alto o visionare il **dettaglio** del contatto cliccando sulla riga interessata. La schermata del dettaglio dell'utente (fig.5) mostra i dati del contatto memorizzati, cliccando sul pulsante di modifica l'utente viene portato alla pagina dedicata per la **modifica** dei dati e il seguente salvataggio. Sempre dalla schermata dell'elenco contatti l'utente può cliccare il pulsante per aggiungere un **nuovo contatto**: verrà portato in una pagina dove potrà inserire i dati del nuovo contatto. (fig.6)

Nella parte alta dell'applicazione è presente un menu che riporta il nome dell'utente e che contiene i link per visionare il profilo dell'utente o per effettuare il **log out**. All'interno della pagina del **profilo** sono presenti due tab: una tab per la modifica dei dati del profilo utente (fig.7) e una tab per la modifica della **password** di accesso dell'utente. (fig.8)

Seguendo i requisiti raccolti, l'applicazione viene pensata fin da subito per essere sfruttata su dispositivi mobili. Per questo motivo si sceglie di seguire lo sviluppo dell'interfaccia grafica con modalità **mobile first**. Per favorire l'utilizzo anche agli utenti meno esperti o con problematiche fisiche si sceglie di seguire i principi **KISS** e "**Less Is More**".

Simple Contacts

Sign up

Figure 2: Registrazione

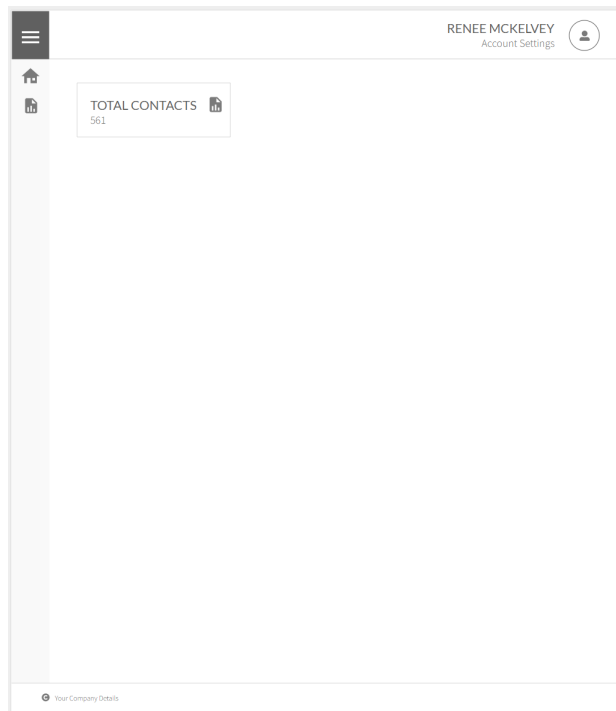


Figure 3: Dashboard

RENEE MCKELVEY
Account Settings

CONTACTS LIST

Filter

Reset

| | | | | |
|------------------|----------------------|------------------------------|---------------------------|---------|
| Jerry Mattedi | 3956565484 Mobile | 251-661-5362 Phone Number | New York Location | Details |
| Elianora Vasilov | 3956565484 Mobile | 351-661-3252 Phone Number | Ontario Location | Details |
| Marcos Anguiano | 3956565484 Mobile | 251-661-5362 Phone Number | Milan Location | Details |
| Alvis Daen | 3956565484 Mobile | 351-661-3252 Phone Number | Las Vegas Location | Details |
| Lissa Shipsey | 3956565484 Mobile | 251-661-5362 Phone Number | San Francisco Location | Details |

Your Company Details

Figure 4: Lista contatti

RENEE MCKELVEY
Account Settings

Contact details

Firstname

Lastname

Mobile

Email

City

Zip

Street

Save

Cancel

Your Company Details

Figure 5: Dettagli contatto

The screenshot shows a web application interface for 'RENE MCKELVEY Account Settings'. On the left is a dark sidebar with a hamburger menu icon, a home icon, and a document icon. The main content area is titled 'New contact' and contains a form with the following fields: 'Firstname' and 'Lastname' (side-by-side), 'Mobile' and 'Email' (side-by-side), 'City' and 'Zip' (side-by-side), and a single 'Street' field. At the bottom of the form are two buttons: 'Save' (dark grey) and 'Cancel' (light grey). The footer of the page includes a small icon and the text 'Your Company Details'.

Figure 6: Inserimento nuovo contatto

The screenshot shows the same web application interface, but the main content area is titled 'User profile'. The form contains four fields: 'Firstname', 'Lastname', 'Email', and 'Mobile', arranged vertically. At the bottom of the form are two buttons: 'Save' (dark grey) and 'Cancel' (light grey). The footer of the page includes a small icon and the text 'Your Company Details'.

Figure 7: Profilo utente

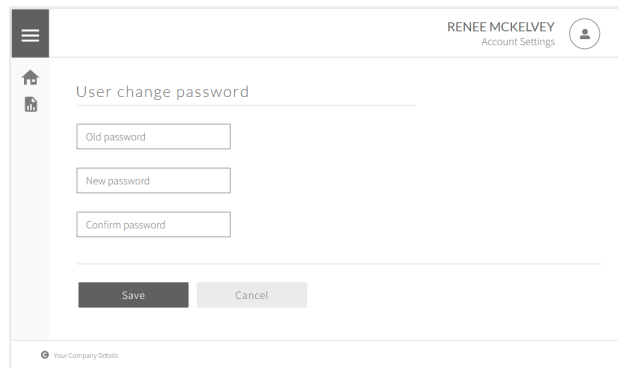


Figure 8: Cambio password utente

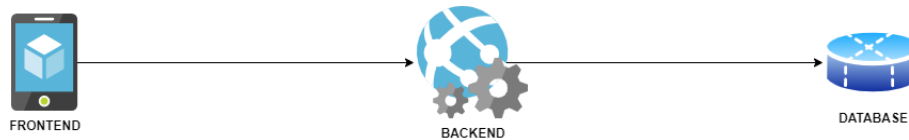


Figure 9: Diagramma architettura

0.3.2 Design dell'architettura del sistema

Il sistema viene pensato **modulare** fin dal principio, in modo da favorire le fasi di sviluppo, la separazione dei compiti, i test, il deploy e la scalabilità futura. L'applicazione è composta da due parti distinte: una parte di **frontend** lato client e una parte di **backend** lato server. Il backend offrirà al frontend degli endpoint di tipo **REST API** che il frontend potrà consumare effettuando chiamate HTTP con tecnica AJAX. Il backend dialogherà anche con il **database** per storicizzare e recuperare le informazioni. (fig. 9)

0.3.3 Design software

Dall'analisi dei requisiti emergono chiaramente due entità software: **Contact** e **User**. Il Contact rappresenta il singolo contatto memorizzato in rubrica e contiene tutte le informazioni inerenti. Lo User rappresenta l'utente che si registra e quindi accede all'applicazione e che gestisce la propria rubrica. Tale entità conterrà le informazioni necessarie all'autenticazione e all'utente stesso. (fig.10)

0.4 Tecnologie

Per il **frontend** viene scelta una tecnologia che permetta di rispettare i requisiti, quindi che permetta l'interazione dinamica con l'utente e al contempo

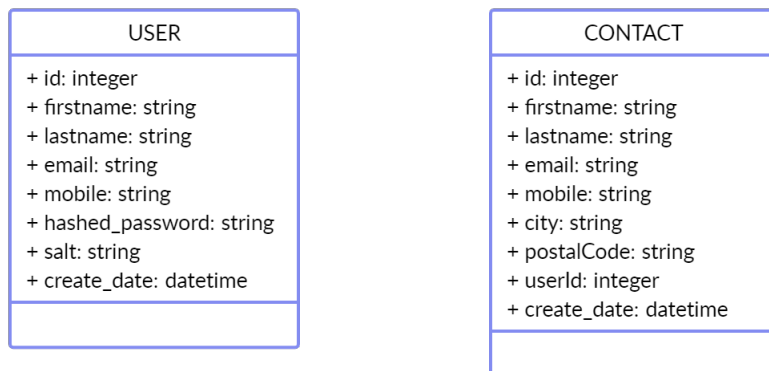


Figure 10: Entità software

l'interazione con la parte di backend. Oltre a questi fattori, viene scelta una tecnologia matura, ampiamente utilizzata e che utilizzi un linguaggio di programmazione vicino alle mie personali conoscenze tecniche pregresse. Per tutti questi motivi viene scelto **Angular** sviluppato con linguaggio **TypeScript**.

Per quanto riguarda la parte di **backend**, si vuole adottare anche qui una tecnologia matura e stabile, ampiamente utilizzata, che faccia utilizzo dello stesso linguaggio di programmazione della parte frontend così da poter sfruttare a pieno le conoscenze tecniche acquisite. Per questi motivi si sceglie **Node.js** e il framework **Express**.

Per la parte di storicizzazione dei dati si sceglie un **database** documentale quale **MongoDB** per via sia del modello applicativo sviluppato, che si presta molto bene ad essere rappresentato con tale logica, sia per le tecnologie scelte che si sposano molto bene con questo tipo di metodologia di storicizzazione dei dati.

Per soddisfare il requisito di **notificare** gli utenti in real-time, la scelta cade sull'utilizzo di **websocket** e quindi sulla libreria **socket.io**

0.5 Codice

0.5.1 Frontend

L'applicazione **Angular** di frontend viene organizzata nei seguenti moduli:

- **share**: importa e raggruppa tutte le librerie di cui fa utilizzo l'applicazione Angular (es. Bootstrap)
- **core**: contiene le logiche custom sviluppate (es. JWT interceptor), componenti software riutilizzabili (es. componente alert-messages), classi di servizio per il reperimento dei dati (es. contact service)

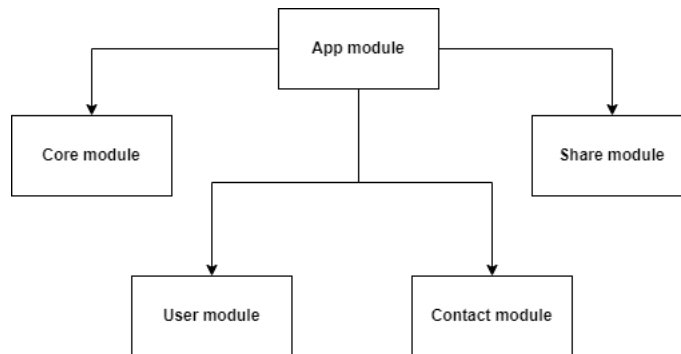


Figure 11: Diagramma moduli frontend

- **user**: rappresenta la parte applicativa dell'utente e quindi contiene i componenti di registrazione, login, profilo, dashboard
- **contact**: rappresenta la parte applicativa della rubrica e quindi contiene la i relativi componenti per mostrare la lista dei contatti, il form di creazione di un nuovo contatto e i dettagli di un contatto esistente.

Interessante è l'implementazione del **JwtInterceptor** che prima di ogni chiamata http inserisce l'header di autenticazione nella request.

Altro punto interessante è la registrazione del client agli **eventi** di creazione contatto e apertura scheda contatto tramite **websocket**. Grazie a tali eventi il client riceverà le notifiche.

0.5.2 Backend

Il backend sviluppato su framework **Express** fa uso della libreria ODM (Object Data Modelling) **Mongoose** per accedere al database MongoDB.

L'applicazione è organizzata in molto molto semplice, con i controller che espongono gli **endpoint** e le entità **User** e **Contact** che modellano le entità del dominio e sono allo stesso tempo repository per l'accesso ai dati.

L'**autenticazione** è del tipo **JWT Token** e gestita dal middleware dedicato.

La gestione degli **eventi** da notificare ai client è contenuta nel controller della parte Contact che, al salvataggio o al caricamento di un contatto, reperisce le informazioni necessarie e invia l'evento al client interessato.

0.6 Test

Per motivi di tempo i test automatici sono stati solamente abbozzati nell'applicativo frontend e mancano totalmente nella parte backend.

Si sono svolti numerosi **test manuali** sia durante lo sviluppo ma soprattutto nella fase finale, momento in cui l'applicazione è stata verificata manualmente

per controllare sia l'aderenza ai **requisiti** richiesti sia il rispetto del **Decalogo di Nielsen**.

I test sono stati eseguiti sia su dispositivi **desktop** sia su dispositivi **mobile**, privilegiando quest'ultimi visto il requisito di sviluppo **mobile first**.

Al termine di questi test preliminari, si è passati al **test di usabilità** eseguito dall'utente rappresentativo che ha accompagnato lo sviluppo dell'applicazione fin dalla stesura dei requisiti. Tale test ha confermato la correttezza di molte parti dell'applicazione e ha fatto emergere alcune criticità e dettagli da correggere in una successiva release.

0.7 Deployment

Per motivi di tempo la fase di rilascio viene posticipata a una release successiva e qui solamente abbozzata. L'idea è quella di rilasciare l'applicazione tramite piattaforma **cloud**. Questo approccio fornisce totale libertà per quanto riguarda la scalabilità dell'applicazione, oltre alla facilità di manutenzione e di deploy. Nel dettaglio, l'idea è quella di eseguire il deploy della parte client in un **bucket**: i bucket sono perfetti per distribuire file statici quali un'applicazione Angular e scalano automaticamente, inoltre sono molto economici. La parte di back-end si pensa di rilasciarla sotto forma di container **Docker** e quindi successivamente scalabile trasformandolo in uno swarm o addirittura inserendolo in un'architettura Kubernetes. Per quanto riguarda il database MongoDB, sarà sufficiente il deploy di un'immagine Docker già pronta.

0.8 Conclusioni

L'applicazione Simple Contacts è risultata essere un'ottima palestra che, grazie alla sua semplicità di concetto, ha permesso di mettere in pratica e fare propri concetti e tecnologie di questo corso. Come emerso in svariati punti della relazione, l'applicazione prodotta è in uno stato ancora **embrionale**, possiamo dire essere il prodotto della prima iterazione di sviluppo, una sorta di **MVP** che riesce già a dimostrare il funzionamento di tutte le feature pensate. Ipotizzando di proseguire con successive iterazioni di sviluppo, si potrebbero portare a termine anche le parti di **test** automatici e di **deploy**, migliorando inoltre la parte di **UI** e magari aggiungendo **nuove funzionalità**.