> CommandlineTeam _

Commandline Team - Progetto "TuTourSelf"

Manuale Manutentore

Versione	1.0.0
Approvazione	Alberto Battistini
Redazione	Daniele Penazzo Alberto Battistini
Verifica	Marco Giollo
Stato	Approvato
Uso	Esterno
Destinato a	Commandline Team Prof. Tullio Vardanega Prof. Riccardo Cardin TuTourSelf S.r.l.

Descrizione
Questo documento contiene il Manuale Manutentore per il progetto TuTourSelf del gruppo CommandLine Team

commandlineteam@gmail.com

Contenuto

1 Changelog

Versione	Data	Nominativo	Ruolo	Descrizione
1.0.0	2018-05-07	Alberto Battistini	Responsabile	Approvazione del documento.
0.1.0	2018-05-06	Marco Giollo	Verificatore	Prima verifica dell'intero documento.
0.0.7	2018-05-05	Giovanni Motterle, Giulia Corò	Progettisti	Inserite descrizioni tecnologie per il client
0.0.6	2018-05-05	Giulia Corò	Progettista	Inserita descrizione delle componenti frontend
0.0.5	2018-05-04	Daniele Penazzo	Progettista	Inserite classi frontend
0.0.4	2018-05-04	Daniele Penazzo	Progettista	Aggiunta sezioni visione generale e riferimento classi
0.0.3	2018-05-03	Alberto Battistini	Progettista	Descrizione tecnologie Google Maps API e SuperAgent
0.0.2	2018-05-03	Daniele Penazzo	Progettista	Aggiunta tecnologie, intro e requisiti
0.0.1	2018-05-02	Daniele Penazzo	Progettista	Prima redazione

Tabella 1: Changelog di questo documento

2 Introduzione

2.1 Scopo del documento

Questo documento rappresenta il *Manuale del Manutentore* per l'applicazione web *TuTourSelf* sviluppata da CommandLine Team.
La finalità di questo documento è fornire al manutentore le modalità di installazione e di utilizzo del prodotto, i requisiti necessari per l'uso dello stesso, oltre ai moduli ed i *framework*$_G$ utilizzati per lo sviluppo.
Questo documento inoltre contiene la struttura dei *package*$_G$ e delle *classi*$_G$ contenuti nell'applicazione, così da rendere più semplice la ricerca e l'eventuale aggiunta di funzionalità.

2.2 Scopo del prodotto

Il capitolato si prefigge di creare una piattaforma per facilitare comunicazione ed eventuale successivo accordo$_G$ tra artisti indipendenti e locali.
In particolare, il prodotto si prefigge di:

- Creare un database centralizzato di artisti/gruppi, che possa inglobare collegamenti con eventuali altri profili di ciascun artista/gruppo (es. sito web personale, Youtube, Bandcamp, Pinterest, varie pagine social);
- Creare un database centralizzato di locali convenzionati, che puntualizzi la tipologia di artista richiesta e la strumentazione messa a disposizione dal locale stesso;
- Creare un'applicazione web (che poi in futuro possa essere facilmente trasformata in applicazione mobile) che permetta **facilmente** di mettere in contatto artisti e locali (tenendo conto della compatibilità tra le caratteristiche dell'artista e quelle del locale), consentendo di giungere ad un accordo tra le due parti. L'accordo, tra le altre cose, deve definire la data dell'esibizione, la durata dello show e il compenso concordato;
- Creare un sistema che permetta ad artisti, locali e pubblico di rilasciare feedback in base a parametri prestabiliti (comportamento, professionalità, quantità di pubblico, incasso ecc.);
- Creare un modulo che permetta al gestore del locale di pagare facilmente il cachet all'artista/gruppo.

2.3 Ambiguità e glossario

Al fine di evitare ambiguità dovute all'uso di linguaggio tecnico specifico, in appendice a questo documento è inserito un glossario che riporta le definizioni dei termini riportati con una "G" a pedice.

2.4 Assunzioni di conoscenza

Nella redazione di questo manuale, si assume che il manutentore abbia conoscenze nel settore della programmazione ad eventi e JavaScript, oltre che una conoscenza almeno basilare del linguaggio UML2.

3 Analisi tecnica delle tecnologie utilizzate

3.1 Tecnologie per client e server

3.1.1 Google Maps API

3.1.1.1 Descrizione: Google Maps è un servizio che consente la ricerca e visualizzazione di carte geografiche di buona parte della terra. Attraverso le *Google Maps API* è possibile realizzare mappe personalizzate, applicazioni mobile che le integrino, oppure creare applicazioni web che forniscono servizi di ricerca personalizzati.

3.1.1.2 Vantaggi

Semplicità d'uso: L'utilizzo di tali API è facilitato dal fatto che sfruttano servizi *REST*$_G$, ovvero un insieme di risorse sulle quali un client può chiedere le operazioni canoniche del protocollo *HTTP*$_G$.
Attraverso una serie di *endpoint*$_G$ nel nostro caso delle semplici *URL*$_G$, si recuperano i dati forniti dai sistemi di Google riguardanti i luoghi ricercati. Il tutto è possibile con pochissime righe di codice.

Documentazione dettagliata: L'apprendimento e l'utilizzo delle Google Maps API è facilitato da un'abbondante e dettagliata documentazione a supporto degli sviluppatori.

Facilità nella manipolazione dei dati: I dati ottenuti dai sistemi Google sono ritornati nel formato JSON, il quale è perfettamente compatibile con i moduli del sistema.

Integrazione ottima con React: Attraverso il modulo react-google-maps è possibile integrare la mappa e funzionalità di ricerca attraverso delle componenti React già prefabbricate.

3.1.1.3 Svantaggi

Utilizzo gratuito limitato: Dall' 11 giugno 2018 Google aggiornerà i propri termini di utilizzo delle Google Maps API nei progetti software degli sviluppatori. In particolare, la nuova piattaforma Google Maps Platform fatturerà al cliente le spese di utilizzo una volta finito un budget gratuito di 200 dollari.

3.1.1.4 Considerazioni: Le Google Maps API rappresentano una soluzione efficiente ed efficace. Esse permettono di rilasciare delle funzionalità di ricerca su mappa che soddisfano le richieste del proponente e dell'utente finale in maniera semplice e veloce.

3.2 Tecnologie per il client

3.2.1 React.js

3.2.1.1 Descrizione React è una libreria JavaScript utilizzata per costruire interfacce grafiche di single-page applications e applicazioni mobile. É open source ed è mantenuta da Facebook e da una comunità di sviluppatori. Attraverso React è possibile definire gli elementi fondamentali di una UI, ovvero i componenti, e assemblarli in modo da ottenere la pagina desiderata. I cambiamenti nello dello stato dei componenti sono automaticamente riflessi nell'interfaccia grafica, rendendo più agevole lo sviluppo di progetti medio/grandi.

3.2.1.2 Vantaggi

DOM Virtuale: React permette di rendere più veloce il rendering dei cambiamenti nella pagina astraendo il Document Object Model del browser. Tutte le modifiche vengono prima applicate ad una nuova struttura dati che permette elaborazioni più efficienti per poi applicare le differenze al DOM.

Metodi del ciclo di vita: Lo sviluppatore può utilizzare una serie di metodi che ogni componente React espone. Questi metodi sono invocati al raggiungimento di specifici stadi nel ciclo di vita del componente, come per esempio "componentWilMount", invocato prima di essere montato nel DOM, oppure "componentDidMount", invocato subito dopo.

JSX: JavaScript XML è un' estensione di JavaScript che permette di utilizzare una sintassi simile a quella HTML per strutturare la view dei componenti.

Proprietà: Attraverso le proprietà dei componenti è possibile far fluire le informazioni lungo una possibile gerarchia. Un componente padre può passare parte del suo stato ai componenti figli, che a loro volta potranno far risalire dati raccolti dalle interazioni con l'utente.

3.2.1.3 Svantaggi: Costruire un insieme di pagine web appoggiandosi così pesantemente a JavaScript può, in certi casi, essere uno svantaggio Per esempio in contesti dove l'interprete non è abilitato ad eseguire, l'intera applicazione non è utilizzabile.

3.2.1.4 Considerazioni: React è stato scelto sia per una richiesta esplicita da parte del proponente, sia per la sua larga diffusione e supporto da parte della comunità. Inoltre fornisce una buona documentazione con molti esempi che permettono di capire approfonditamente i meccanismi propri della libreria.

3.2.2 HTML 5

3.2.2.1 Descrizione: L'HTML è un linguaggio di markup di pubblico dominio, la cui sintassi è stabilita dal World Wide Web Consortium (*W3C*). La versione attuale, la quinta, è stata rilasciata dal W3C nell'ottobre 2014.

3.2.2.2 Vantaggi

Standard: HTML è uno standard *W3C*.

3.2.2.3 Svantaggi

Scarso supporto: HTML5 non è supportato dai browser più vecchi.

3.2.2.4 Considerazioni: E' stato adottato HTML5 in quanto è il linguaggio usato per le pagine web.

3.2.3 CSS 3

3.2.3.1 Descrizione: Il CSS è un linguaggio usato per definire la formattazione di documenti HTML, XHTML e XML. Le regole per comporre il CSS sono contenute in un insieme di direttive emanate a partire dal 1996 dal *W3C*. Le specifiche CSS3 sono costituite da sezioni separate dette "moduli". A causa di questa modularizzazione, le specifiche CSS3 hanno differenti stati di avanzamento e stabilità.

3.2.3.2 Vantaggi

Standard: CSS3 è uno standard *W3C*.

3.2.3.3 Svantaggi

Comportamento variabile: Il comportamento delle regole CSS potrebbe variare a seconda dei browser utilizzati.

3.2.3.4 Considerazioni: Il CSS3 è lo standard usato per la definizione dello stile nelle pagine web.

3.2.4 babeljs

3.2.4.1 Descrizione: Babel è un transpiler che permette di convertire codice sorgente scritto secondo lo standard ECMAScript 6_G in codice ECMAScript 5_G.

3.2.4.2 Vantaggi: Possibilità si usare la web application anche con browser meno recenti e che supportano parzialmente ES6.

3.2.4.3 Svantaggi: Non è possibile avere un controllo diretto sul codice prodotto dato che il processo di transpiling è completamente automatizzato.

3.2.4.4 Considerazioni: Utilizzando questo strumento è possibile sfruttare le funzionalità offerte dall'ultimo standard JavaScript senza dover rinunciare al supporto di alcuni browser.

3.2.5 Bootstrap

3.2.5.1 Descrizione: Bootstrap è definito un "HTML, CSS, and JS toolkit from Twitter", ovvero una raccolta di strumenti grafici, stilistici ed impaginazione che permettono di avere a disposizione una gran quantità di funzionalità e di stili modificabili e adattabili a seconda delle nostre esigenze. Bootstrap è compatibile con tutte le ultime versioni dei principali browser e la sua principale funzione è il responsive web design, ovvero il fatto che il design delle pagine web sia in grado di adattarsi dinamicamente a seconda della grandezza e delle caratteristiche del dispositivo utilizzato. Esso si pone, perciò, come una libreria multidispositivo e multipiattaforma.

3.2.5.2 Vantaggi

Uniformità: L'usare un tema di Bootstrap ci permette di avere una grafica uniforme nonostante venga sviluppata in contemporanea da più persone.

3.2.5.3 Svantaggi

jQuery: Bootstrap fa uso di jQuery, che è deprecato.

3.2.5.4 Considerazioni: La comodità di usare un tema bootstrap per le componenti HTML e la sua gradevole resa grafica ci ha portato a sceglierlo.

3.2.6 SuperAgent

3.2.6.1 Descrizione: SuperAgent è un'API ajax$_G$ leggera progettata per essere flessibile, leggibile e velocemente apprendibile a differenza di altre API disponibili.

3.2.6.2 Vantaggi

Compatibilità con Node.js: L'API SuperAgent è compatibile con Node.js, ovvero l'ambiente server utilizzato per il nostro progetto.

Facilità d'uso: SuperAgent permette di effettuare chiamate REST in semplicità grazie ai metodi flessibili disponibili all'utente.

Ottima qualità della documentazione: La documentazione a supporto dell'utente è ottima e permette quindi un veloce apprendimento.

3.2.6.3 Svantaggi: Non sono stati individuati degli svantaggi derivanti dall'utilizzo di questa tecnologia.

3.2.6.4 Considerazioni: SuperAgent rappresenta un'ottima soluzione per effettuare chiamate attraverso il protocollo HTTP necessarie per soddisfare le funzionalità del nostro prodotto software.

3.3 Tecnologie per il server

3.3.1 Amazon Web Services (AWS)

3.3.1.1 Descrizione: Amazon Web Services è un servizio di *cloud computing*$_G$ che permette la creazione di istanze di server virtuali con possibilità di scalabilità automatizzata.

3.3.1.2 Vantaggi

Semplicità: I server AWS sono semplici da configurare e sono richieste poche azioni di mantenimento;

Sicurezza: La configurazione predefinita di AWS è molto restrittiva, aumentando la sicurezza del sistema.

3.3.1.3 Svantaggi

Server virtuali: L'uso di server virtuali non garantisce le stesse prestazioni di un server, o una rete di server dedicati.

Amazon linux: Il sistema operativo Linux predefinito offerto da Amazon manca di pacchetti utili al prodotto, che hanno dovuto essere installati manualmente. Inoltre il sistema operativo stesso ha forti limitazioni sul lato della flessibilità.

3.3.1.4 Considerazioni: La semplicità di configurazione iniziale è una caratteristica positiva per un sistema cloud, ma il sistema operativo predefinito può risultare molto ostico per chi non riesce a padroneggiare Linux in modo pressochè completo.

3.3.2 Node.js

3.3.2.1 Descrizione: Node.js è un ambiente dedicato all'esecuzione di codice *JavaScript*$_G$ lato server. Questo ambiente è dotato di un'architettura ad eventi che permette input/output asincroni, allo scopo di ottimizzare *throughput*$_G$ e *scalabilità*$_G$ di un'applicazione web.

3.3.2.2 Vantaggi

Approccio ad eventi: La natura ad eventi di Node.js permette di evitare tutti i problemi derivanti dalla gestione diretta della concorrenza (ad esempio tramite *threads*$_G$), a patto di lasciare il controllo di tale gestione al sistema;

Flessibilità: Nonostante la natura ad eventi, Node.js permette di fare uso di vari tipi di stili architetturali, tra cui API-REST, di cui facciamo uso;

Modularità: Node.js permette di installare facilmente nuovi moduli tramite *npm*$_G$, consentendo così di importare nuove funzionalità dall'esterno, diminuendo il peso del mantenimento del prodotto finale, scaricandone parte sulla comunità;

Scalabilità: Data la natura ad eventi e distribuita di Node.js, è possibile creare una piattaforma facilmente scalabile che interagisce con il browser in tempo reale;

Licenza MIT: Consente senza alcun problema l'uso in prodotti commerciali *closed-source*$_G$.

3.3.2.3 Svantaggi

Approccio asincrono: Data la natura ad eventi di Node.js, è necessario che il gruppo faccia uso di tecniche di codifica asincrone, più complesse rispetto alla classica programmazione lineare, cioè può rendere il codice complesso e poco mantenibile;

Gestione dei moduli: In alcuni casi può risultare difficile gestire i moduli di Node.js, è necessario quindi avere una visione chiara ed ampia del prodotto per non incorrere in problemi dovuti a sovrascrittura di moduli.

3.3.2.4 Considerazioni: Nonostante le difficoltà date dal modello asincrono e dall'uso del linguaggio JavaScript, nato principalmente per la programmazione lato client, Node.js risulta essere ormai lo standard de-facto per quanto riguarda la programmazione di applicazioni web lato server. Inoltre le sue caratteristiche di modularità e scalabilità lo rendono un prodotto molto appetibile per la realizzazione di questo progetto.

3.3.3 MongoDB e Mongoose

3.3.3.1 Descrizione: MongoDB è un database di tipo *NoSQL*$_G$ orientato ai documenti in stile *JSON*$_G$, mentre Mongoose è uno strumento di modellazione per oggetti in MongoDB.

3.3.3.2 Vantaggi

Scalabilità orizzontale: Tramite *Sharding*$_G$, MongoDB è in grado di rendere molto semplice l'aggiunta di nuove macchine allo scopo di distribuire il carico di lavoro.

Approccio PA/EL: In caso di partizione di rete, MongoDB preferisce mantenere disponibilità e bassa latenza a scanso della consistenza.

Funzioni di aggregazione: È possibile fare uso di modelli *MapReduce*$_G$ per processare dati in batch.

Query ad-hoc: MongoDB supporta query su campi, su campioni di dati o anche query con espressioni regolari, questo dà a MongoDB un'altissima flessibilità di lavoro.

3.3.3.3 Svantaggi

Configurazione di default poco sicura: La configurazione predefinita di MongoDB consente l'accesso al database da parte di chiunque, questo può portare a forti pericoli nel caso una macchina venga infettata da *Ransomware*$_G$.

Approccio PA/EL: In caso di partizione di rete, il database diverrebbe inconsistente: in prima analisi questo potrebbe non essere un problema, ma è sempre consigliato tenere conto della possibilità che si possano creare problemi.

Approccio NoSQL: L'approccio NoSQL di MongoDB non consente di effettuare *Join*$_G$, scaricando sul codice JavaScript questa responsabilità;

Query sugli indici non atomiche: Questo può portare ad avere dati poco aggiornati se una query viene effettuata quando un documento viene aggiornato

Licenza Complessa: MongoDB è rilasciato sotto licenza GNU Affero GPL v3, che consente l'uso commerciale, ma non è ben chiaro se l'obbligo di rilascio del sorgente riguardi solamente eventuali versioni modificate di MongoDB o anche i prodotti che ne fanno uso.

3.3.3.4 Considerazioni: Insieme a Node.js, MongoDB è ormai diventato parte di uno stack molto usato nella programmazione di applicazioni web (lo stack più usato è Node.js + MongoDB + AngularJS), e nonostante alcuni seri svantaggi dati dalla configurazione predefinita e dalle modalità di operazione decise, la forte scalabilità orizzontale e le garanzie di disponibilità date sono molto interessanti e portano molto valore aggiunto al progetto.

3.3.4 Socket.io

3.3.4.1 Descrizione: Socket.io è un modulo che consente ad un'applicazione Node.js di mettere a disposizione dell'utente una comunicazione bidirezionale basata su eventi in tempo reale.

3.3.4.2 Vantaggi

Semplicità d'uso: Per creare una chat in tempo reale in stile broadcast sono necessarie pochissime righe di codice, questo migliorerà la leggibilità del codice e ridurrà l'incidenza di errori di codifica.

Supporto browser: Questo modulo è ben supportato dai browser, essendo testato e garantito come compatibile con Firefox, Chrome, Internet Explorer (fino alla versione 8), Edge e Safari, oltre a diversi browser mobile.

Rilevazione disconnessioni Il modulo è in grado di rilevare eventuali disconnessioni da parte degli interlocutori ed agire di conseguenza, solitamente tentando automaticamente di ristabilire la connessione.

Debugging semplificato: È possibile attivare il debug di Socket.io semplicemente importando una *Variabile d'ambiente*$_G$.

Test: Questo modulo è già testata per conto proprio, il che solleva il gruppo dalla scrittura di test dedicati.

Licenza permissiva: La licenza MIT è completamente compatibile con un prodotto closed-source.

3.3.4.3 Svantaggi

Compatibilità: Essendo un modulo che risulta rinnovato e riprogrammato da poco tempo, non possiamo valutarne la compatibilità con il resto delle tecnologie usate.

3.3.4.4 Considerazioni: L'estrema semplicità di codifica, oltre alla presenza di una documentazione delle API molto nutrita e condita di esempi e di un Tutorial completo e molto dettagliato, rende questa libreria estremamente appetibile per la realizzazione a basso livello del sistema di chat in tempo reale fra le parti che vogliono portare a termine un accordo. La compatibilità poco conosciuta è una preoccupazione marginale, dato che il modulo verrà usato soprattutto lato server.

3.3.5 validate-vat

3.3.5.1 Descrizione: Validate-vat è uno script scritto in CoffeeScript atto a verificare e validare partite IVA emesse da un qualunque stato europeo.

3.3.5.2 Vantaggi

Validazione molto semplice La validazione di una partita IVA richiede solo 2 righe di codice, rendendo il codice molto più leggibile ed evitando errori derivanti da una gestione manuale della validazione.

Test: Questo modulo è già testato per conto proprio facendo uso di expect.js, sollevando il gruppo dalla necessità di scrivere test specifici per questo modulo.

Peso: Questo modulo è lungo meno di 200 righe di codice, rendendolo un modulo estremamente leggero che non dovrebbe avere alcun impatto sulle performance.

Licenza: La licenza MIT permette l'uso in prodotti commerciali closed-source.

3.3.5.3 Svantaggi

Richieste possibilmente in chiaro: Da una prima analisi del codice, sembra che le richieste di validazione e le conseguenti risposte siano trasmesse tramite HTTP, in chiaro, cosa che potrebbe risultare problematica sotto alcuni punti di vista.

Limite massimo di richieste: Il servizio con cui si interfaccia questo modulo prevede un errore dedicato specificamente se si eccede il numero massimo di richieste di validazione in contemporanea. Ciò fa presagire alcuni possibili problemi nel caso si volesse scalare il prodotto a livello europeo, in cui un "boom di registrazioni" potrebbe mandare in crisi il sistema. Il gruppo dovrà valutare se vale la pena prendersi in carico tale rischio o se è preferibile attuare una serializzazione delle richieste, rallentando però il sistema.

3.3.5.4 Considerazioni: La semplicità d'uso e la possibilità di ispezionare semplicemente questo modulo fa al caso nostro per la risoluzione del requisito di verifica delle partite IVA, il fatto che le richieste di validazione (e possibilmente anche le risposte) sembrano effettuate in HTTP senza alcun sistema di sicurezza è preoccupante, ma bisogna anche considerare il fatto che tali informazioni sono di dominio pubblico.

3.3.6 Passport.js

3.3.6.1 Descrizione: Passport è un modulo di autenticazione per Node.js, con supporto per login interne o tramite Facebook, Twitter e Google.

3.3.6.2 Vantaggi

Struttura a plugin: Questo modulo fa uso di un sistema a plugin, utile per implementare eventuali strategie di login personalizzate;

Supporto nativo per Facebook e Google Questo permette al gruppo di concentrarsi di più sulle nuove caratteristiche da implementare, piuttosto di dover "reinventare la ruota".

Documentazione: La documentazione di passport.js è molto ricca e riporta esempi specifici per express.js ma che possono essere comunque adattati.

Test: La libreria è molto testata, riportando una copertura del 99%, molto alta. Questo solleva il gruppo dalla stesura di test specifici per molte parti della login.

Licenza permissiva: La licenza MIT permette un uso del modulo compatibile con un prodotto commerciale closed-source.

3.3.6.3 Svantaggi

Express.js: Il modulo viene riportato come compatibile con express.js, ma non è conosciuto il grado di difficoltà posto dall'uso di implementazioni diverse. Questo potrebbe risultare un problema se non è presente documentazione specifica per il caso in esame.

3.3.6.4 Considerazioni: La prospettiva di avere una soluzione "tutto in uno" che comunque consente un certo grado di flessibilità è estremamente buona, soprattutto per una parte del progetto come la gestione degli accessi, che può essere estremamente pericolosa se non configurata a dovere. Si ritiene che il gruppo possa comunque superare eventuali ostacoli che si presenteranno da una documentazione specificamente dedicata ad express.js.

3.3.7 nodemailer

3.3.7.1 Descrizione: Nodemailer si propone come un modulo molto semplice per l'invio di email tramite node.js.

3.3.7.2 Vantaggi

Semplicità: Il codice richiesto da nodemailer è molto semplice e comprensibile, questo aumenta la pulizia del codice e la leggibilità, riducendo l'incidenza e la probabilità di errori.

Supporto STARTTLS/TLS: Il supporto alle tecnologie $STARTTLS_{\text{G}}$ e TLS_{G} permette una trasmissione sicura dei messaggi.

Nessuna dipendenza: L'assenza di dipendenze esterne permette una migliore ispezione del codice del modulo, rendendo inoltre il modulo più leggero;

Test: Il modulo è testato in modo approfondito grazie a test automatici, questo solleva il gruppo dalla redazione di test specifici per questa funzionalità.

3.3.7.3 Svantaggi

Problemi con Gmail: Sono stati riportati più volte problemi tra questo modulo e Gmail, il gruppo potrebbe doversi trovare ad affrontare problemi di compatibilità.

Problemi con suite di sicurezza: Sono stati riportati problemi con firewall ed antivirus sul server usato per l'invio di email, questo porta ad errori criptici come ad esempio errori di timeout.

3.3.7.4 Considerazioni: L'uso di una liberira semplice che permetta di creare codice comprensibile è ben vista dal gruppo che ritiene opportuno prendersi eventualmente in carico delle problematiche di compatibilità analizzate.

3.3.8 Express.js

3.3.8.1 Descrizione Express.js è un framework per applicazioni web minimale, estensibile tramite plugins e con un set di caratteristiche atte alla creazione rapida di applicazioni web sia multi-page che single-page, oltre ad applicazioni di tipo ibrido.

3.3.8.2 Vantaggi

Routing semplice: Il routing tramite Express.js è estremamente semplice ed è possibile separarlo in diversi file per un'organizzazione migliore "per sezioni".

Supporto ai middleware Express permette di fare uso di diversi middleware per aggiungere funzionalità necessarie all'applicazione web.

Supporto per sessioni: Tramite un plugin specifico è possibile integrare e gestire facilmente le sessioni.

3.3.8.3 Svantaggi: Nessuno rilevato.

3.3.8.4 Considerazioni: La semplicità e la pulizia messe a disposizione da Express.js per la creazione di un'applicazione web sono qualità molto importanti per questo progetto, dove è ottimale avere un framework non intrusivo e performante.

3.3.9 BCrypt-NodeJS

3.3.9.1 Descrizione: Questo è un'implementazione in JavaScript puro dell'algoritmo di hashing BCrypt di dimensioni molto ridotte.

3.3.9.2 Vantaggi

Leggerezza: Se l'applicazione web viene eseguita su server Windows non sarà necessario scaricare gli SDK per il funzionamento di alcune soluzioni concorrenti.

Flessibilità: Questa implementazione supporta sia hashing sincrono che asincrono in maniera semplice e pulita.

Nessuna dipendenza esterna: Per la generazione di array di byte casuali, BCrypt-NodeJS si basa sulle librerie standard di Node.js.

3.3.9.3 Svantaggi

Età: Sul repository il progetto risulta essere in versione 0.0.3 da circa 5 anni, e non è più mantenuto attivamente.

Irreperibilità diretta del sorgente: Il repository fa riferimento ad una pagina GitHub non più esistente, il codice è comunque disponibile presso `https://github.com/shaneGirish/bcrypt-nodejs`.

3.3.9.4 Considerazioni Nonostante l'età e l'abbandono del progetto (che comunque risulta essere stato aggiornato nel 2017), consideriamo questa soluzione ideale per la leggerezza e la mancanza di dipendenze.

3.3.10 Connect-Mongo

3.3.10.1 Descrizione: Connect-Mongo è un modulo che permette di memorizzare le sessioni di Express all'interno di una base di dati MongoDB.

3.3.10.2 Vantaggi

Semplicità: L'attivazione del modulo richiede due righe di codice.

Configurabilità: È possibile effettuare configurazioni avanzate del modulo in caso di necessità.

Supporto a JSON: Connect-Mongo converte automaticamente gli oggetti non riconosciuti da MongoDB tramite JSON.Stringify e JSON.parse.

3.3.10.3 Svantaggi

Configurazione di default: La configurazione di default non riusa connessioni di MongoDB già aperte.

3.3.10.4 Considerazioni: Nonostante i possibili problemi prestazionali dovuti al non riuso di connessioni al database, la possibilità di aggiungere supporto alla memorizzazione di sessioni su database, con conseguente aumento della robustezza ed affidabilità dell'applicazione, in modo semplice e pulito è molto appetibile.

3.3.11 GeoLib

3.3.11.1 Descrizione: GeoLib è una libreria che permette calcoli su coordinate geografiche, oltre a diversi tipi di conversione utili.

3.3.11.2 Vantaggi

Semplice: Il calcolo di distanze e la conversione è semplice e veloce, facendo uso degli oggetti messi a disposizione.

3.3.11.3 Svantaggi

Oggetti LatLong: Il calcolo deve fare uso di oggetti interni di tipo LatLong per nascondere la complessità dei calcoli.

3.3.11.4 Considerazioni Questa libreria è ottima per i calcoli che vengono fatti per il rilevamento di locali nelle vicinanze di un percorso tracciato e l'uso di oggetti specifici non è un grosso problema.

4 Requisiti di Sistema

TuTourSelf è un'applicazione web funzionante tramite browser, sia mobile che desktop.

4.1 Dispositivi supportati

L'applicazione è stata testata e risulta compatibile con i seguenti sistemi desktop:

- Microsoft Windows 10;
- Apple OSX Sierra;
- Gentoo Linux (Profile v17).

4.1.1 Browser Supportati

Di seguito viene fornito un breve elenco con le versioni minime di tutti i browser sui quali il funzionamento del nostro prodotto è garantito:

- Google Chrome$_G$: il sistema viene supportato dalla versione **49**;
- Internet Explorer$_G$: il sistema viene supportato dalla versione **9**;
- Microsoft Edge$_G$: il sistema viene supportato dalla versione **15**;
- Firefox$_G$: il sistem viene supportato dalla versione **5**;
- Safari$_G$: il sistema viene supportato dalla versione **11**.

ATTENZIONE: Per rendere effettivo il funzionamento su tali browser, deve necessariamente essere abilitato JavaScript$_G$.

5 Configurazione dell'ambiente di lavoro

5.1 Installazione delle dipendenze

Per funzionare, è necessario installare alcune dipendenze prima di procedere alla configurazione dell'applicazione TuTourSelf.

5.2 Installazione di Node.js

Per prima cosa è necessario installare Node.js, è possibile installarlo tramite il proprio gestore di pacchetti (da root):

Gentoo Linux e derivate: `emerge nodejs`.

Debian e derivate/Ubuntu: `apt-get install nodejs`.

Fedora $\geq$ 22: `dnf install nodejs`.

Fedora $<$ 22 e derivate: `yum install nodejs`.

ArchLinux e derivate: `pacman -S nodejs`.

Alternativamente è possibile fare riferimento al sito ufficiale `https://nodejs.org`.

5.3 Installazione di NPM

Per poter installare le altre dipendenze di TuTourSelf è necessario installare *Node Package Manager* (NPM), solitamente questo applicativo è installato durante l'installazione di Node.js, è possibile controllarne la presenza tramite il comando `npm -v`, in caso positivo tale comando dovrebbe restituire la versione di NPM installata. In caso non fosse disponibile è possibile fare riferimento al sito ufficiale `https://www.npmjs.com/get-npm`.

5.4 Configurazione dell'applicazione

5.4.1 Clonazione della repository

Per prima cosa è necessario clonare la repository privata fornita. Questo può essere fatto con il comando `git clone` *[URL]* .

5.4.2 Installazione/Aggiornamento dei moduli

Successivamente è necessario spostarsi all'interno della cartella del repository ed assicurarsi che sia presente il file `package.json`.
Se il file è presente è possibile installare le dipendenze del backend tramite il comando `npm install`, se invece il file è assente si è verificato un errore durante la clonazione del repository e sarà necessario ricominciare la procedura di configurazione dalla clonazione della repository.
Successivamente è necessario installare le dipendenze del frontend, navigando in `src/frontend/src` con il comando `cd src/frontend/src` e dando nuovamente il comando `npm install`.

5.4.3 Avvio del server

Al termine dell'installazione è possibile tornare alla root del progetto, tramite `cd ../../..` ed avviare il server tramite il comando `npm start`.

6 Architettura

6.1 Formalismo

L'architettura dell'applicativo sarà descritta dal generale al particolare; quindi si descriveranno prima i package e le componenti e successivamente si vedranno in dettaglio le classi contenute, fornendone una descrizione, specificandone l'uso e le relazioni con altre classi.

6.2 Visione generale

TuTourSelf è un'applicazione web suddivisa in due parti principali che comunicano tra loro trasmettendosi oggetti JSON tramite richieste HTTP.

6.2.1 Frontend

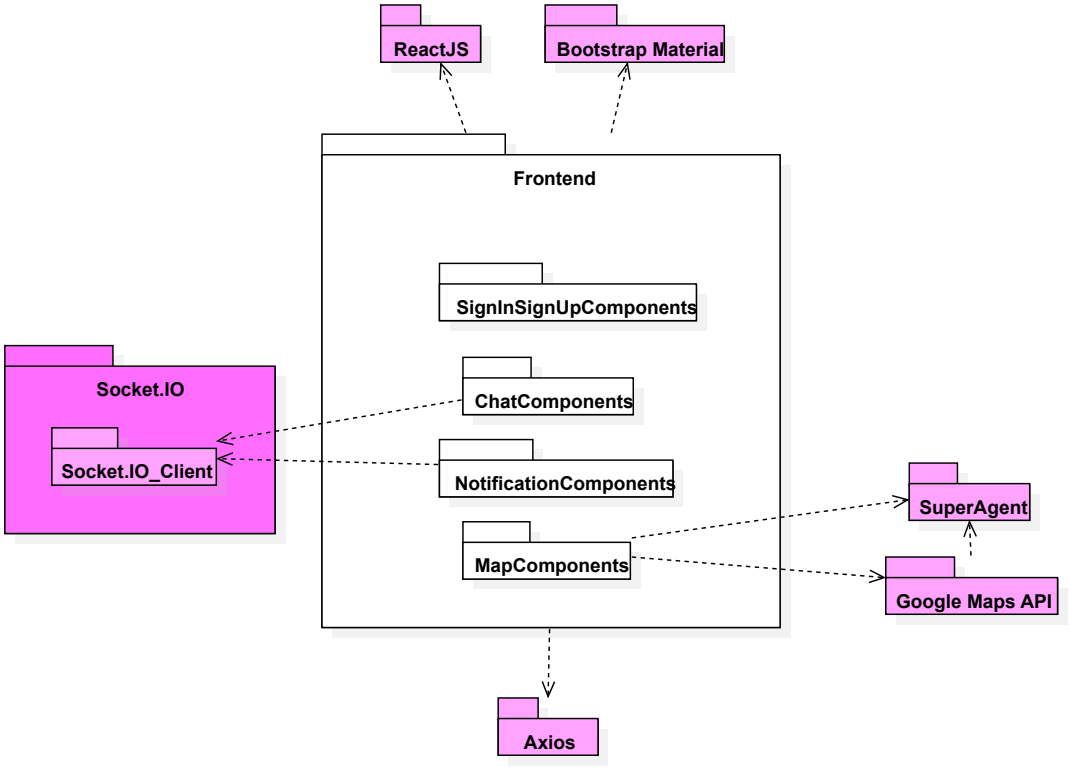

Immagine 1: Visione generale del frontend

Il frontend è una *Single Page Application* (SPA) costruita facendo uso di HTML5, CSS3, JavaScript e React. Questa parte dell'applicazione è intesa per essere semplice e leggera, scaricando la maggior parte del lavoro sul backend ed è costruita secondo un modello a componenti, incoraggiato da React. Il frontend fà uso di Axios per comunicare con il backend tramite richieste HTTP a specifiche path.

6.2.2 Backend

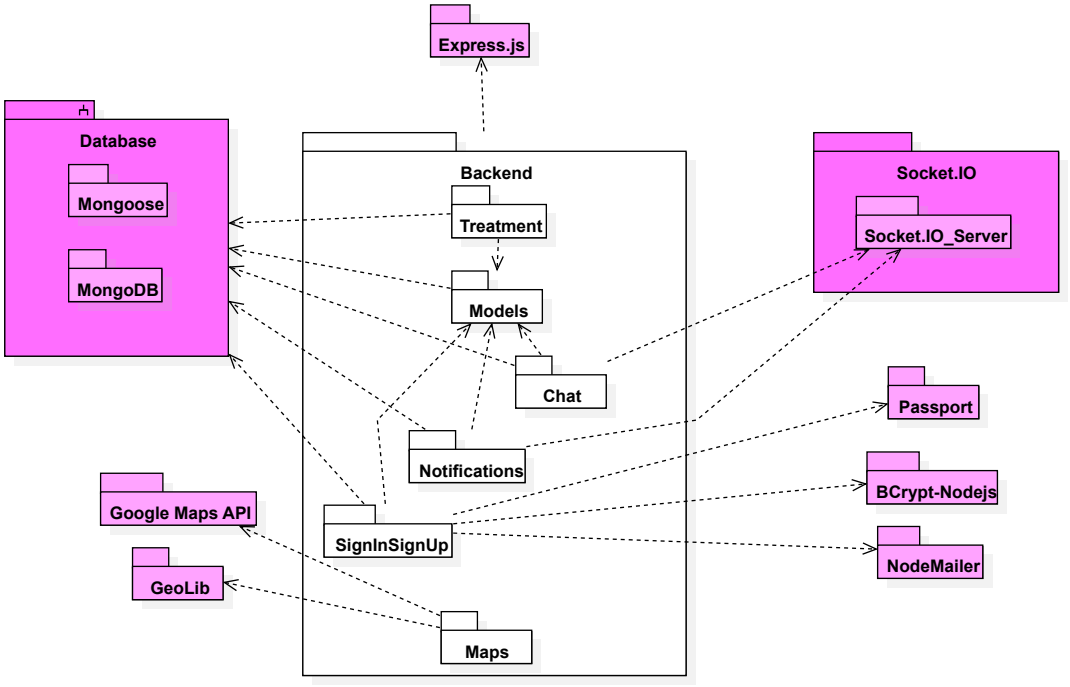

Immagine 2: Visione generale del backend

Il backend è stato sviluppato in modo modulare, facendo uso del paradigma ad eventi incoraggiato da Node.js. Il backend si occupa della conservazione dei dati e della loro elaborazione, della gestione delle sessioni, della gestione delle ricerche e dell'invio dei dati richiesti al frontend.

6.2.3 Package contenuti

TuTourSelf::Frontend Contenente le classi ed i package che vanno a comporre il frontend di TuTourSelf.

TuTourSelf::Backend Contenente le classi ed i package che vanno a comporre il backend di TuTourSelf.

6.2.4 Librerie e framework usati

Express: un framework minimale, ed estensibile tramite plugin, per la creazione di applicazioni web tramite Node.js, implementa un più pulito ed efficace sistema di routing.

React: un framework JavaScript che permette di creare Single Page Applications facendo uso di un modello a componenti, incoraggiando il riuso.

Bootstrap Material: un framework HTML/CSS/JavaScript che permette di costruire facilmente pagine in stile Material Design.

Mongoose: driver e tool di modellazione per MongoDB, mongoose permette di definire facilmente schemi e modelli per il database, imponendo una struttura ben definita ai documenti.

Socket.io: libreria per la comunicazione in tempo reale tramite websockets, socket.io permette di implementare in maniera semplice e pulita sistemi di chat e di notifica in tempo reale.

Axios: una libreria di comunicazione asincrona tramite richieste HTTP, permette di comunicare facilmente e velocemente con il backend tramite richieste poste su determinate routes.

SuperAgent: una seconda libreria di comunicazione tramite richieste HTTP, semplifica di molto le richieste verso le API di Google Maps rispetto ad Axios, rendendo il codice più pulito.

Passport: un framework di autenticazione estensibile tramite l'implementazione di uno strategy pattern interno, permette l'autorizzazione e l'autenticazione all'applicazione.

GeoLib: una libreria dedicata al calcolo su coordinate geografiche, semplica e rende pulito il calcolo di distanze e ricerche su mappe.

BCrypt-Nodejs: una libreria che fornisce un'implementazione nativa dell'algoritmo di salted hashing BCrypt, usata per confrontare e fare salted hashing di password.

NodeMailer: una libreria che consente l'invio automatizzato di email, usato per inviare le email di verifica delle registrazioni effettuate.

7 TuTourSelf::Frontend

Il pacchetto frontend contiene alcune componenti generali che non si ritiene conveniente inserire in pacchetti separati.

TuTourSelf::Frontend::Appbar La barra delle applicazioni che contiene i link necessari per la navigazione sul sito dell'utente non autenticato, ossia:

- L'homepage contenente una barra di ricerca;
- La pagina di autenticazione;
- La pagina di registrazione.

TuTourSelf::Frontend::BasicConfig Questo modulo esporta un oggetto che definisce il dominio al quale effettuare le richieste asincrone utilizzando il package *axios*.

TuTourSelf::Frontend::Dashboard La bacheca dell'utente. In questa pagina l'utente potrà visualizzare tutte le notizie e gli eventi relativi alle pagine (di artisti e locali) da lui seguiti.

TuTourSelf::Frontend::LoggedSearch In questa pagina l'utente potrà effettuare una ricerca delle pagine di artisti, locali ed eventi registrati nel sito. Per il momento è possibile ricercare soltanto tramite keywords, ma in futuro la funzionalità verrà espansa tramite una ricerca geografica.

TuTourSelf::Frontend::SecondStepSignup Questa pagina fa parte della procedura di registrazione al sito. Rappresenta appunto il "secondo step" della registrazione, che consiste nella scelta del proprio tipo utente (artista, gestore di locali oppure spettatore), che servirà poi per il reindirizzamento dell'utente ad una form specifica per il proprio tipo utente.

TuTourSelf::Frontend::Signin La pagina di autenticazione. Un utente che è già registrato al sito potrà accedervi da questa pagina in ogni momento, grazie all'inserimento della propria e-mail e della password scelta in precedenza.

TuTourSelf::Frontend::SigninSignupSearch Questa è la componente radice che tramite le route di React reindirizza le varie pagine visibili all'utente non autenticato, ossia:

- *TuTourSelf::Frontend::UnloggedSearch*;
- *TuTourSelf::Frontend::Signin*;
- *TuTourSelf::Frontend::Signup* .

TuTourSelf::Frontend::Signup La pagina di registrazione. Qui è visibile il "primo step" della registrazione. L'utente dovrà inserire e-mail, password e nuovamente la password per un'ulteriore conferma, dopodiché riceverà un'email per la conferma della registrazione.

TuTourSelf::Frontend::ThirdStepSignup Il "terzo step" della registrazione di un utente. Questa componente ha il compito di instradare le ruote alle tre diverse form per la registrazione degli utenti, che sono:

- *[TuTourSelf::Frontend::signupForms::ArtistSignup]*;
- *[TuTourSelf::Frontend::signupForms::LocalManagerSignup]*;

- *[TuTourSelf::Frontend::signupForms::SpectatorSignup].*

TuTourSelf::Frontend::UnloggedSearch Pagina che contiene l'homepage del sito, che comprende il logo TuTourSelf, una breve descrizione, e la barra di ricerca.

TuTourSelf::Frontend::UserInterface Questa è la componente radice che tramite le route di React reindirizza le varie interfacce utente, a seconda del tipo di utente che è autenticato in quel momento. Queste sono:

- *[TuTourSelf::Frontend::usersInterface::ArtistInterface]*;
- *[TuTourSelf::Frontend::usersInterface::LocalManagerInterface]*;
- *[TuTourSelf::Frontend::usersInterface::SpectatorInterface].*

7.1 TuTourSelf::Frontend:chat

Questo pacchetto contiene tutte le componenti che riguardano il sistema di chat interno.

TuTourSelf::Frontend::chat::ActiveChat Componente che rappresenta la finestra della chat attiva.

TuTourSelf::Frontend::chat::ChatHistory Componente che rappresenta l'anteprima di una chat avuta in precedenza, sulla quale si potrà cliccare per aprire la rispettiva chatroom. L'anteprima della chat consiste nella visualizzazione del solo nome dell'interlocutore.

TuTourSelf::Frontend::chat::ChatHistoryItem Componente che rappresenta lo spazio della chat nel quale verranno mostrate tutte le conversazioni precedenti.

TuTourSelf::Frontend::chat::ChatInterlocutorMessage Componente che rappresenta un singolo messaggio di testo che viene ricevuto.

TuTourSelf::Frontend::chat::Chat Pagina della chat. Tutte le componenti che riguardano la chat sono figlie di questa componente.

TuTourSelf::Frontend::chat::ChatMyMessage Componente che rappresenta un singolo messaggio di testo che viene inviato.

TuTourSelf::Frontend::chat::ChatProposal Componente che rappresenta lo spazio all'interno della pagina chat nel quale verrà posizionata la proposta.

TuTourSelf::Frontend::chat::SocketWrapper Componente che incapsula tutte le chiamate che avvengono tra il backend e il frontend attraverso il package *socket.io*, il quale viene utilizzato nel sistema soltanto per funzioni relative alla chat.

TuTourSelf::Frontend::chat::ViewReplyProposal Componente che visualizza la proposta ricevuta e alla quale si può rispondere.

7.2 TuTourSelf::Frontend::localManagement

Questo pacchetto contiene le componenti riguardanti il sistema di gestione dei locali da parte dell'utente registrato come gestore di locali.

TuTourSelf::Frontend::localManagement::LocalCreationForm Form per l'aggiunta di un nuovo locale alla lista dei propri locali.

TuTourSelf::Frontend::localManagement::LocalPreview Componente che rappresenta la miniatura di un locale, sulla quale il gestore può cliccare per accedere alla rispettiva pagina locale.

TuTourSelf::Frontend::localManagement::ManageLocals Pagina dove il gestore può visualizzare e gestire tutti i propri locali.

7.3 TuTourSelf::Frontend::signupForms

Questo pacchetto contiene le componenti riguardanti il "terzo step" della registrazione degli utenti, ossia la parte che riguarda i dati specifici per ogni tipo di utente.

TuTourSelf::Frontend::signupForms::ArtistSignup Form per la registrazione di un utente come artista.

TuTourSelf::Frontend::signupForms::LocalManagerSignup Form per la registrazione di un utente come gestore di locali.

TuTourSelf::Frontend::signupForms::SpectatorSignup Form per la registrazione di un utente come spettatore.

7.4 TuTourSelf::Frontend::usersAppbar

Questo pacchetto contiene le appbar specifiche per ogni categoria di utente, oltre ad una aggiuntiva mostrata durante le fasi intermedie della registrazione.

TuTourSelf::Frontend::usersAppbar::ArtistAppbar Barra delle applicazioni visibile ad un utente autenticato come artista.

TuTourSelf::Frontend::usersAppbar::LocalManagerAppbar Barra delle applicazioni visibile ad un utente autenticato come gestore di locali.

TuTourSelf::Frontend::usersAppbar::SecondStepSignup Barra delle applicazioni visibile ad un utente che sta completando la sua registrazione (ossia che sta effettuando il "secondo step" oppure il "terzo step" della registrazione).

TuTourSelf::Frontend::usersAppbar::SpectatorAppbar Barra delle applicazioni visibile ad un utente autenticato come spettatore.

7.5 TuTourSelf::Frontend::usersInterface

Questo pacchetto contiene le interfacce specifiche per ogni tipo di utente.

TuTourSelf::Frontend::usersInterface::ArtistInterface Interfaccia specifica per un utente autenticato come artista. Le varie pagine sono raggiunte grazie ai link presenti nell'*ArtistAppbar*.

TuTourSelf::Frontend::usersInterface::LocalManagerInterface Interfaccia specifica per un utente autenticato come gestore di locali. Le varie pagine sono raggiunte grazie ai link presenti nella *LocalManagerAppbar*.

TuTourSelf::Frontend::usersInterface::SpectatorInterface Interfaccia specifica per un utente autenticato come spettatore. Le varie pagine sono raggiunte grazie ai link presenti nella *SpectatorAppbar*.

7.6 TuTourSelf::Frontend::usersProfileEditor

Questo pacchetto contiene le sezioni di modifica profilo dei vari tipi d'utenza.

TuTourSelf::Frontend::usersProfileEditor::EditArtistProfile Form per la modifica delle informazioni contenute nella pagina di un artista.

TuTourSelf::Frontend::usersProfileEditor::EditLocalManagerProfile Form per la modifica delle informazioni contenute nella pagina di un gestore di locali.

TuTourSelf::Frontend::usersProfileEditor::EditSpectatorProfile Form per la modifica delle informazioni contenute nella pagina di uno spettatore.

7.7 TuTourSelf::Frontend::usersProfilePage

Questo pacchetto contiene le componenti atte alla visualizzazione delle pagine di profilo.

TuTourSelf::Frontend::usersProfilePage::ArtistPage Pagina pubblica di un utente autenticato come artista.

TuTourSelf::Frontend::usersProfilePage::LocalProfile Pagina pubblica di un locale.

TuTourSelf::Frontend::usersProfilePage::ManagerPrivateProfile Pagina profilo di un utente autenticato come gestore di locali.

TuTourSelf::Frontend::usersProfilePage::SpectatorPrivateProfile Pagina profilo di un utente autenticato come spettatore.

8 TuTourSelf::Backend

Nel pacchetto backend sono presenti alcune classi e moduli che non si ritiene opportuno raggruppare in specifici pacchetti, per motivi di funzionalità.

TuTourSelf::Backend::Server contiene la configurazione e le istruzioni di avvio del server;

TuTourSelf::Backend::Routes contiene la dichiarazione e la configurazione delle routes accettate dal server.

8.1 TuTourSelf::Backend::models

Questo pacchetto contiene tutti i modelli mongoose (e le classi che li estendono), usati per la conservazione dei dati usati dall'applicazione.

TuTourSelf::Backend::models::emailConfirmation questo modello rappresenta le conferme di registrazione in attesa.

TuTourSelf::Backend::models::local questo modello rappresenta un locale all'interno del sistema;

TuTourSelf::Backend::models::message rappresenta un messaggio di chat, memorizzato all'interno del database.

TuTourSelf::Backend::models::pendingNotifDB rappresenta una notifica destinata ad un utente al momento offline.

TuTourSelf::Backend::models::proposal rappresenta una proposta memorizzata in database;

TuTourSelf::Backend::models::room rappresenta una stanza di chat, memorizzata nel database.

TuTourSelf::Backend::models::treatment rappresenta un'istanza di accordo, memorizzata nel database.

TuTourSelf::Backend::models::user rappresenta un utente registrato presso l'applicazione.

8.2 TuTourSelf::Backend::signup

Questo pacchetto contiene la business logic per la registrazione di un nuovo utente e di locali nel sistema.

TuTourSelf::Backend::signup::register questo modulo definisce la business logic e la configurazione di Passport per la registrazione di nuovi utenti tramite sistema interno, Facebook e Google SignIn.

TuTourSelf::Backend::signup::localCreation definisce la business logic riguardante la creazione di un nuovo locale.

TuTourSelf::Backend::signup::secondStepSignup definisce le business logic riguardante il completamento della registrazione, con scelta della categoria di utenza di cui entrare a far parte e l'inserimento dei dati personali.

8.3 TuTourSelf::Backend::signIn

Questo pacchetto contiene la business logic e le configurazioni necessarie per l'autenticazione di un utente al sistema.

TuTourSelf::Backend::signIn::loginMethods definisce la configurazione di Passport per la login tramite autenticazione interna, Facebook o Google SignIn.

8.4 TuTourSelf::Backend::chat

Questo pacchetto contiene la business logic della chat.

TuTourSelf::Backend::chat::chat_HistoryManager questa classe è utilizzata per la gestione dello storico di una stanza di chat, comprende funzionalità per la memorizzazione di messaggi e per il recupero dello storico;

TuTourSelf::Backend::chat::chat_MessageManager questa classe serve alla gestione dei messaggi in tempo reale, richiamando il gestore dello storico per la memorizzazione e facendo uso di Socket.io per la trasmissione in tempo reale e la notifica.

TuTourSelf::Backend::chat::chat_RoomManager questa classe è usata per la creazione, il recupero e la gestione delle stanze di chat;

TuTourSelf::Backend::chat::chatHelper questo modulo si occupa della configurazione e l'inizializzazione della chat e dei moduli connessi.

8.5 TuTourSelf::Backend::helpers

Questo pacchetto contiene alcune componenti di utilità che servono ad altre componenti per funzionare correttamente.

TuTourSelf::Backend::helpers::userSocketArray questa classe copre una lacuna esposta da Passport-Socket.io, tale lacuna impedisce di collegare un certo utente alla lista di socket aperti. Questa classe risolve il problema offrendo funzionalità di inserimento, recupero ed eliminazione di socket, in maniera simile ad una multimappa.

8.6 TuTourSelf::Backend::notifications

Questo pacchetto contiene le componenti e la business logic del sistema di notifiche in tempo reale.

TuTourSelf::Backend::notifications::notificationManager questa classe si occupa della gestione e dell'invio tramite socket.io di notifiche in tempo reale.

TuTourSelf::Backend::notifications::pendingNotifications questa classe rappresenta un'estensione del modello di mongoose dedicato alla memorizzazione delle notifiche in sospeso, offrendo funzionalità quali recupero delle notifiche in sospeso dato un utente.

8.7 TuTourSelf::Backend::search

Questo pacchetto contiene la business logic e le componenti atte al funzionamento del sistema di ricerca.

TuTourSelf::Backend::search::search_Keywords questo modulo si occupa della ricerca tramite keywords e restituzione dei risultati ottenuti dalla ricerca nel database.

8.8 TuTourSelf::Backend::routes

Questo pacchetto contiene le routes di Express.js e le annesse configurazioni.

TuTourSelf::Backend::routes::auth questo modulo contiene la configurazione di tutte le routes che fanno capo al sistema di registrazione ed autenticazione, sia di utenti che di locali.

A Glossario

A

Apple Safari: *Browser*$_G$ creato dalla compagnia informatica Apple Corp.

B

Browser: Applicativo usato per la navigazione sul web. **AJAX**: Acronimo di Asynchronous JavaScript and XML, è una tecnica di sviluppo software per la realizzazione di applicazioni web interattive

C

Cloud Computing: Paradigma di erogazione di risorse informatiche caratterizzato dalla disponibilità su richiesta di tali risorse tramite internet.
Closed-Source: *Vedi Software Proprietario.*
Classi: Un costrutto di un linguaggio di programmazione usato come modello per creare oggetti.

E

ECMAScript5: Linguaggio di programmazione standardizzato e mantenuto da Ecma International. Aggiunge lo "strict mode" inteso a provvedere un più completo controllo di errori ed a scoraggiare gli sviluppatori ad incappare in errori.
ECMAScript6: Apporta modifiche sintattiche significative che aprono la porta ad applicazioni più complesse, includendo classi e moduli definendoli semanticamente come lo "strict mode" in ECMAScript 5.
Edge: *Vedi Microsoft Edge.*

F

Framework: Insieme integrato di componenti software prefabbricate, usato in congiunzione a nuovo codice scritto per soddisfare dei requisiti specifici.
Firefox: *Vedi Mozilla Firefox.*

G

Google Chrome: Browser sviluppato da un team di Google LLC.

I

Internet Explorer: Browser sviluppato da Microsoft Corp., l'ultima versione rilasciata è stata Internet Explorer 11. Ora è stato sostituito dal più moderno *Microsoft Edge*$_G$.

J

JavaScript: Linguaggio di programmazione interpretato, solitamente usato per lo sviluppo frontend ma di recente introdotto anche per lo sviluppo e la programmazione di Backend.
JSON: JavaScript Object Notation, è una notazione testuale che permette di descrivere degli oggetti in JavaScript.

M

Malware: Contrazione di "Malicious Software" (letteralmente "Software Malevolo"), fa riferimento a tutti quei software che hanno come obiettivo (primario o collaterale), il danneggiamento di un sistema informatico, il furto di dati o qualunque attività atta a danneggiare un'eventuale vittima.
Microsoft Edge: Browser sviluppato da Microsoft Corp. ed attualmente supportato.
MapReduce: Funzione in due passaggi, composta da:

- Map: Transformazione di un input in un certo output;

- Reduce: Combinazione di una tupla in input in una certa tupla di output di cardinalità inferiore.

Mozilla Firefox: Browser web sviluppato da Mozilla.

N

Node.js: Ambiente di esecuzione JavaScript *server-side*$_G$ basato sul motore V8 di Google Chrome.
NPM: Acronimo di Node Package Manager, è il gestore pacchetti predefinito di Node.js.

O

Opportunistic TLS: Estensione di protocolli per la comunicazione testuale, Opportunistic TLS offre la possibilità di passare da una connessione in chiaro ad una criptata usando una porta separata per la comunicazione crittografata. Per far partire tale connessione criptata si fa solitamente uso del comando STARTTLS.

R

REST: Abbreviativo di REpresentational State Transfer, è un'architettura per sistemi software distribuiti basata sull'uso di una struttura di URL ben precisa e l'uso di verbi HTTP specifici (GET, POST, DELETE, ...).
Ransomware: Contrazione di Ransom Software (letteralmente "Software che chiede il riscatto"), un particolare tipo di Malware il cui comportamento più diffuso è la crittografia di dati sensibili in modo che il proprietario non possa più accedervi. A crittografia completata, il software chiede un riscatto affinchè il proprietario possa accedere nuovamente ai propri dati.

S

Safari: *Vedi Apple Safari.*
Server-side: Lato Server, solitamente inteso come "esecuzione di codice da parte del server".
Sharding: Sistema di partizione orizzontale dei dati in un database, in cui ciascuna "shard" (pezzo di database) è contenuto all'interno di una diversa macchina, così da alleggerire il carico sulle singole macchine.
Software Proprietario: Il software proprietario, chiamato anche closed source, è un software la cui licenza consente al beneficiario il suo utilizzo sotto particolari condizioni ed impedendone altre come la modifica, la condivisione, lo studio, la ridistribuzione o l'ingegneria inversa.
STARTTLS: *Vedi Opportunistic TLS.*

T

Threads: Sottoprocessi, adatti all'esecuzione concorrente su sistemi di elaborazione multiprocessore o multicore.
TLS: Acronimo di Transport Layer Security, TLS è una famiglia di protocolli crittografici di presentazione.

U

URL: Acronimo di Uniform Resource Locator, un URL è una stringa che identifica univocamente l'indirizzo di una risorsa internet.

V

Variabile d'ambiente: Variabile presente nell'ambiente globale di un interprete dei comandi, accessibile dai processi tramite meccanismi indipendenti dal sistema operativo in uso.