OMI Finder

D5: Sprint #1

Corso di Ingegneria del Software [145829]

Conti Marco	182438	[marco.conti-1@studenti.unitn.it]
Mongera Davide	209273	[davide.mongera@studenti.unitn.it]
Pugliese Simone	209036	[simone.pugliese@studenti.unitn.it]
Tomaselli Elia	209613	[elia.tomaselli@studenti.unitn.it]

University of Trento
Via Sommarive 14, 80123
Povo (TN), Italy

Indice

Project Idea

Per l'Italia, i dati delle zone territoriali omogenee (zone OMI) sono gestiti dall'Agenzia delle Entrate tramite l'Osservatorio del Mercato Immobiliare italiano. Una zona OMI è una zona delimitata da un perimetro poligonale all'interno di un comune, dove vengono impostati 2 valori di minimo e massimo, che indicano il costo in euro per unità di superficie in mq. Le quotazioni OMI, aggiornate ogni semestre, sono relative ai comuni censiti negli archivi catastali. È quindi possibile che l'elenco dei comuni presenti in Banca Dati differisca nei diversi semestri per effetto di variazioni circoscrizionali (soppressione/costituzione di nuovi comuni). Lo scopo generale di questo progetto è quello di sviluppare una piattaforma (denominata OMI Finder) che renda i dati delle zone territoriali omogenee (zone OMI) offerti dall'Osservatorio del Mercato Immobiliare dell'Agenzia delle Entrate semplici, intuitivi e rapidi. I vari attori che sfrutteranno le funzionalità e i servizi offerti dalla piattaforma potranno trovare, in base alle loro necessità, diversi dati pre-elaborati dal software proprietario, sempre pronti all'uso.

Link Utili

- GitHub Repository
- Heroku
- Apiary

Organizzazione della repository e Strategia di Branching

Per organizzare la repository abbiamo deciso di utilizzare il **Feature Branch Flow**, dato che per un progetto così piccolo il Gitflow Workflow sembrava più di intralcio che di aiuto nel nostro sviluppo. La nostra repository parte dalla creazione dei vari diagrammi preparativi alla fase di sviluppo, le cartelle *Diagrams*, *Media* e *Deliverables* contengono rispettivamente i diagrammi uml sorgente, gli screenshot e gli export in SVG per i documenti ed i pdf dei render finali per ogni deliverable. A parte i classici file README.md e .gitignore poco rimane. Tutti i source file di codice sono nella cartella *Software*, essa è divisa in 2 sottocartelle: *FrontEnd* e *BackEnd*. Queste cartelle conterranno tutti i file rilevanti per quella particolare sezione del software. La cartella *_Frontend* è un outlier, dato che contiene i file di un prototipo di frontend che non abbiamo avuto l'occasione di implementare con il resto del software, questo artefatto è stato portato da commit precedenti e non è stato ancora cambiato. Al primo momento libero i contenuti di questa cartella verranno integrati con il resto del software e migrati alla cartella *Software/FrontEnd*. I file liberi nella root della cartella non appartenenti ai sistemi di Git sono quelli necessari alla funzione di heroku, vale a dire: *Procfile* e *package.json*.

Product Backlog

ID	Nome	User Story	Importanza	Story Points	DoD	Task ID	Tasks	Stima
1	vantaggio_informazione	Sono un investitore nel real estate, ho bisogno dei dati su una particolare zona OMI per permettermi di fare migliori decisioni di investimento.	10	5	Questo è possibile quando il software permette di interagire con i dati tramite chiamate HTTP. I dati devono essere scorribili ma non manipolabili.	T01	Inizializzare Node.js su branch	1
						T02	Gestione profilo e webserver Heroku	5
						T03	Creazione database MongoDB	1
						T04	Collegamento tra Node.js e Atlas	3
						T05	Realizzazione servizi API con Express.js	2
						T06	Deploy Database MongoDB Online	1
						T07	Interfacciamento Js con MongoDB (Mongoose)	5
						T08	Popolazione del database con i dati da GeoPoi	8
2	dati_per_ricerca	Come ricercatore, sto facendo uno studio che cerca di correlare [fattore] alla locazione geografica, quindi ho bisogno di accesso ad un database organizzato che posso filtrare per luogo.	30	2	Questo è possibile quando le chiamate API non permettono solo di accedere a semplici aggregazioni di dati pre-fabbricate, ma è possibile filtrare le singole istanze di dati, inviando come argomenti HTTP i parametri o filtri necessari	T09	Algoritmo di identificazione zona OMI a partire da delle coordinate	8
						T02	Gestione profilo e webserver Heroku	5
						T05	Realizzazione servizi API con Express.js	2
						T04	Collegamento tra Node.js e Atlas	3
						T06	Deploy Database MongoDB Online	1
						T01	Inizializzare Node.js su branch	1
						T08	Popolazione del database con i dati da GeoPoi	8
						T10	Integrazione del servizio di pagamento Stripe	8
3	integrazione_software	Come compagnia immobiliare, il nostro software privato ha bisogno di integrare le API così da potersi interfacciare con i dati OMI.	20	3	Questo è possibile quando le API fornite dal nostro servizio sono espansive, documentate e affidabili. Non deve esserci confusione nella relazione richiesta - dati inviati in risposta.	T11	Creazione documento users	1
						T01	Inizializzare Node.js su branch	1
						T04	Collegamento tra Node.js e Atlas	3
						T02	Gestione profilo e webserver Heroku	5
						T12	Controllo tipologia utente (Freemium/Premium)	2
						T05	Realizzazione servizi API con Express.js	2
						T06	Deploy Database MongoDB Online	1
						T13	Servizio di gestione registrazione e accesso	5
						T14	Possibilità di modifica password utente	2
						T15	Gestione conflitti registrazione utente	5
4	uso_ricreativo	Come utente internet annoiato, voglio trovare fatti interessanti riguardanti il luogo in cui vivo.	50	1	Questo è possibile quando il nostro sito permette la ricercare dei valori di una Zona OMI tramite l'inserimento dei parametri attraverso un form dedicato sul sito	T16	Inizializzazione frontend con placeholder static page	1
						T02	Gestione profilo e webserver Heroku	5
						T17	Pagina web con mappa	13
5	consultare_la_documentazione	Come programmatore di [azienda immobiliare], voglio leggere documentazione estensiva per poter poi poi integrare all'interno del software dell'azienda le API di Omi Finder.	40	1	Questo è possibile quando esiste una documentazione estensiva e dettagliata dell'API all'interno del frontend disponibile in qualsiasi momento.	T16	Inizializzazione frontend con placeholder static page	1
						T19	Creazione pagina per Documentazione tramite Apiary	5
						T20	Realizzazione Documentazione con Pagina Apiary	5
						T21	Homepage Sito web	3
						T22	Pagina di registrazione e login	5

Sprint #1

Goal

L'obiettivo del primo sprint sarà quello di avere il Frontend completo (esclusa la parte relativa alla mappa), il Database MongoDB operativo tramite Heroku con i dati delle zone OMI, il servizio di registrazione e autenticazione funzionante, oltre alle API per ottenere le diverse zone OMI.

Sprint Planning

Tasks	Story Points	Story Points	Complete	Estimated Hours	Effective Hours
T01	Inizializzare Node.js su branch	1	YES	1	1
T02	Gestione profilo e webserver Heroku	5	YES	2	3
T03	Creazione database MongoDB	1	YES	1	0,666
T04	Collegamento tra Node.js e Atlas	3	YES	1	5
T05	Realizzazione servizi API con Express.js (v1)	2	YES	4	4
T06	Deploy Database MongoDB Online	2	YES	1	0,666
T07	Interfacciamento Js con MongoDB (Mongoose)	5	YES	2	3
T08	Popolazione del database con i dati da GeoPoi	8	ALMOST	8	15
T11	Creazione documento users (v1)	3	YES	0,25	2
T12	Controllo tipologia utente (Freemium/Premium)	3	YES	2	6
T13	Servizio di gestione registrazione e accesso	3	ALMOST	5	0
T14	Possibilità di modifica password utente	3	NO	2	0
T15	Gestione conflitti registrazione utente	3	NO	5	0
T16	Inizializzazione frontend con placeholder static page	1	YES	0,25	0,25
T19	Creazione pagina per Documentazione tramite Apiary	5	NO	1	0
T21	Homepage Sito web	3	YES	1,5	1,5
T22	Pagina di registrazione e login	5	YES	2,5	3

Il progetto è suddivisibile in diverse tematiche che verranno prese in considerazione singolarmente durante lo sprint. Le tematiche sono principalmente il Database, il Frontend, il servizio di registrazione e di autenticazione e l'API vera e propria. Inizialmente, sarà necessaria la creazione del database e il setup con Heroku, per permettere a tutto il team di lavorare in autonomia alle proprie task. Successivamente, si procederà con le restanti attività. In questo meeting abbiamo deciso su quali task lavorare durante questo sprint, nonostante la decisione di prendere in considerazione la maggior parte delle task nel backlog originale, con alcuni di noi nuovi a queste infrastrutture ci danno l'idea di essere abbastanza generiche, con il pensiero di aggiungerne se necessario per il prossimo sprint ne ne assegniamo comunque in questa quantità. Miglioreremo la precisione nella selezione dei nostri budget nel prossimo sprint.

Testing

Number	Description	Test case data	Preconditions	Dependencies	Expected result	Actual result	Notes
0	Registrazione di un nuovo utente avvenuta con successo	Naviga alla pagina del frontend e inserisci nome utente e email non registrati, e una password			Viene effettuata la registrazione e il frontend notifica l'utente della registrazione avvenuta con successo		
1	Login con utente già registrato	Naviga alla pagina del frontend e inserisci nome utente o email, e password corretti			Viene effettuato il login e il frontend notifica l'utente del login avvenuto con successo		
1.1	Login con password errata	Naviga alla pagina del frontend e inserisci nome utente o email registrati e una password errata			Il frontend notifica l'utente di avere sbagliato le credenziali di accesso		
1.2	Login con nome utente o email non registrati e una password qualsiasi	Naviga alla pagina del frontend e inserisci nome utente o email non registrati e una password qualsiasi			Il frontend notifica l'utente di avere sbagliato le credenziali di accesso		
2.1	Api - lista di possibilità rispetto ad un fattore	Manda la richiesta http per la lista di possibilità rispetto ad un determinato parametro, tramite heroku			La richiesta è risposta con un oggetto JSON contenente una lista di tutti i valori che corrispondono a quel parametro, se il parametro non è valido la lista è vuota		
2.2	Api - dati sull'utilizzo di una zona	Manda la richiesta http per il campo "valori" con la possibilità di specificare quale sezione del campo, tramite heroku			La richiesta è risposta con un oggetto JSON contenente una lista di tutti i valori che corrispondono a quel parametro, se il parametro non è valido la lista è vuota		

Sprint Review

Nel seguente sprint ci siamo concentrati sul realizzare uno scheletro funzionante dell'applicazione. Per il momento ci sono ancora delle piccole parti che non sono state rifinite (autenticazione, comunicazione tra Frontend e il resto del progetto, riempimento dati Database...) e che hanno bisogno di altro tempo per venire realizzate. Tutto questo verrà fatto nel secondo sprint, assieme a tutte le altre feature mancanti del progetto che si possono vedere qua sotto. In ogni caso l'obiettivo dello sprint è stato raggiunto ragionevolmente e si può proseguire a migliorare il progetto nel secondo sprint. Nonostante i vari frammenti non siano collegati tra loro, individualmente sono molto avanti verso il loro completamento. In una applicazione lineare come questa ci è sembrata la procedura migliore. Non ha fruttato al livello che volevamo ma non ci ha deluso, vedendo il progresso fatto.

Product Backlog Refinement

Nel secondo sprint verranno considerati i task fondamentali per raggiungere l'obiettivo del progetto che avevamo già delineato alla prima stesura del backlog. Data l’esperienza del primo sprint, è chiaro che però molti di essi sono troppo generici e devono essere spezzati, ora che abbiamo più esperienza con l’infrastruttura su cui stiamo lavorando dovremo essere in grado di farlo. Questo però richiede rivisitare le sezioni di ogni membro del team da parte del gruppo, per avere feedback su cosa sarà necessario per collegare tutti i componenti. Tale processo sarà parte integrante del planning del secondo sprint. Tra questi elementi, alcuni di nota sono lo streamlining dell’implementazione di mongoose, che al momento è, seppur funzionante, disordinata e convoluta.

Retrospective

Nel seguente sprint ci siamo concentrati sul realizzare uno scheletro funzionante dell'applicazione, mentre per il successivo sprint verranno raffinate le parti che sono state realizzate nello sprint corrente. Se avanza tempo verrà introdotta anche la sezione relativa alla mappa sul sito web. Probabilmente viste le diverse tematiche del progetto (Database, Frontend, Backend e login/registrazione) è conveniente che diverse persone si occupino di diverse tematiche.

- Commento di Davide: Io ho lavorato principalmente sulla backend, in particolare il setup di heroku, il collegamento con Atlas e l'utilizzo di express per esporre le api a richieste http. Sono partito velocemente riuscendo a fare il setup sia di node.js che della repository git nei tempi previsti. Si è rilevato un problema interfacciarmi con nuovi sistemi quali heroku, Atlas e mongoose. In particolare voglio esporre un aneddoto di queste settimane di sviluppo: a mia insaputa mongoose opera in maniera non bene descritta nella documentazione, in particolare quando da node js si fa la richiesta per una collection e si utilizza un nome (nel mio esempio "zone-omi") mongoose automaticamente trasforma ogni stringa in Plurale dato che questa sembra essere la sintassi standard (nel mio caso diventando "zone-omis"). Questo faceva in modo che tutte le mie richieste per la cellection specifica ritornassero nulle, e gli errori rilevati a runtime non davano luce alla situazione. Ho speso una intera giornata cercando di risolvere questo particolare problema, e sono contento di aver scoperto che non sono solo. Questo evento mi ha aperto gli occhi alle catastrofi che può creare un sistema non ben documentato. La maggior parte dei miei problemi girano intorno a questo concetto. Sento che solo verso gli ultimi giorni sono veramente entrato nel mindset dello sprint. Ricalibrerò le mie aspettative ed il lavoro di cui prendere carico nel prossimo.
- Commento di Elia: ho lavorato sul setup del Database MongoDB, Heroku, e sulla realizzazione del Frontend. Il frontend è funzionante, è stato usato React.js con TypeScript e Sass, manca solamente il collegamento con il backend. L'unica sezione parte mancante è la mappa.
- Commento di Marco: mi sono concentrato sull'inserimento delle 27245 zone all'interno del Database MongoDB. Il programma di conversione dei dati è quasi completo ma necessita ancora di qualche ritocco.
- Commento di Simone: mi sono occupato della creazione del documento Users in MongoDB, oltre alla stesura del codice per la ricerca di tutti i permessi utenti (Freemium/Premium) e dei permessi di un singolo utente. Purtroppo, ho perso molto tempo nello scrivere codice in file inesistenti, oppure usando metodologie errate, per cui non sono riuscito a integrare anche le API per il Login utente e la registrazione dello stesso (due funzionalità che avrei voluto terminare prima del secondo Sprint).