

Diario di lavoro

Luogo	Trevano
Data	08.01.2019

Lavori svolti

Ho passato la mattinata a decidere quale progetto scegliere e, una volta fatto, ho dato un'occhiata al quaderno dei compiti nel dettaglio per poi parlarne con il docente Adriano Barchi, mio responsabile.

Durante il colloquio mi è stato spiegato nel dettaglio la funzionalità di questo lavoro, del perché sia nato e di come faccia da "contenitore" per 3 progetti che son stati proposti nello scorso semestre. Tuttavia, mi è anche stato detto che non mi devo preoccupare delle 3 parti, che funzionano in modo autonomo richiedendo solamente il prodotto finale che è possibile creare tramite il mio lavoro. Il sito infatti ha la possibilità di creare dei file csv dei dati che saranno salvati tramite l'applicativo che andrò a creare.

Abbiamo discusso anche riguardo al mio desiderio di utilizzare Spring Boot, con il quale già mi son trovato bene in precedenza. Purtroppo però Spring Boot necessita di un server TomCat, il quale non è sempre disponibile come opzione nei web hosting, andando a restringere il campo delle possibilità. Il mio responsabile mi ha quindi invitato a fare una panoramica dei siti che lo permettono, in modo da valutarne la fattibilità. In caso contrario, l'idea rimane di usare qualcosa di diverso rispetto a PHP, e quindi imparare un nuovo linguaggio per arricchire ancora di più le mie opzioni future.

Una volta spiegato abbastanza nel dettaglio i requisiti necessari al progetto, abbiamo definito che nonostante il quaderno dei compiti richieda una pianificazione iniziale entro il primo giorno, avendo perso metà giornata questa la farò il prossimo giorno di lavoro, completando solamente l'analisi dei requisiti per identificare al meglio il progetto.

Nel pomeriggio mi son dedicato a cercare dei servizi di web hosting, trovando offerte penso siano possibili:

<https://www.heliohost.org/#whitelist>, ancora da verificarne l'effettivo utilizzo, gratuito.

<https://www.cloudfoundry.org/how-to-try-cloud-foundry/>, ancora da capire come funziona, gratuito

<https://www.mochahost.com/it/java.php>, 5.57\$ al mese con svariate opzioni.

Ho dato inoltre un'occhiata veloce riguardo alla possibilità di creare file PDF tramite Spring Boot. Ho trovato quello che cercavo tramite questo link: <https://www.baeldung.com/java-pdf-creation>.

Per terminare, ho quasi completato l'analisi dei requisiti (non sono riuscito a terminarla, sebbene abbia ben in chiaro come completarla) e risistemato tutta la struttura, sia di quest'ultima che di cartelle e diari. Ho inoltre creato la cartella GitHub, che vedrò di tenere aggiornata affinché sia protetto da eventuali perdite di dati. <https://github.com/DyumanBulloni/FsyManager>

Ho deciso di nominare il progetto come FsyManager. Nel caso sia un problema, provvederò ad utilizzare quello definito sul quaderno dei compiti, nonostante trovi che come titolo avrebbe molto meno impatto e sarebbe più scomodo da utilizzare.

Problemi riscontrati e soluzioni adottate

Nessun problema particolare.

Punto della situazione rispetto alla pianificazione

Pianificazione ancora da definire.

Programma di massima per la prossima giornata di lavoro

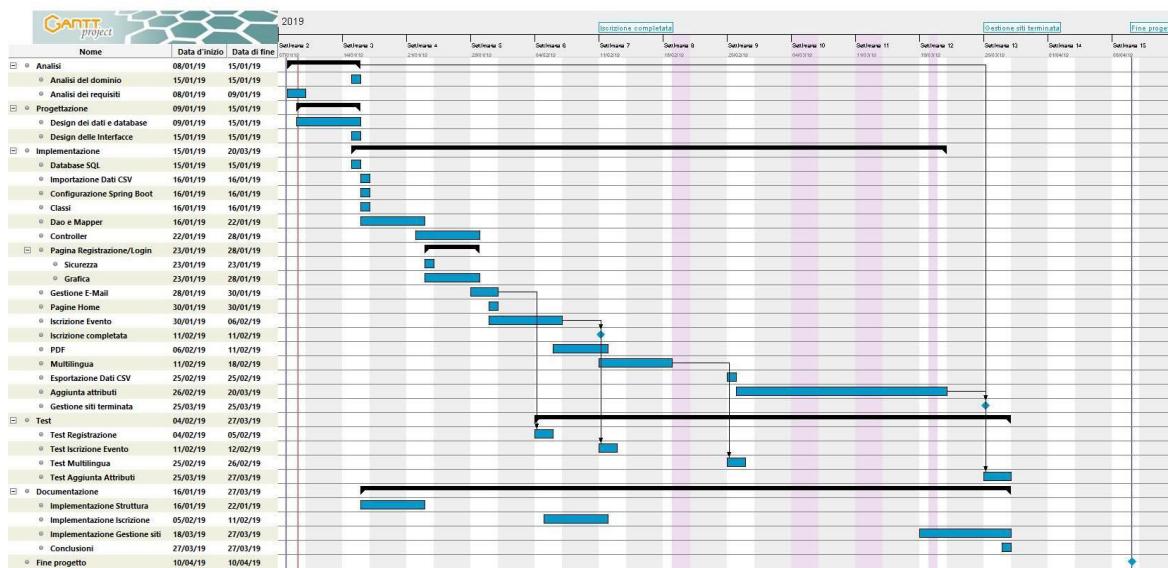
Creare il Gantt Preventivo, finire l'analisi dei requisiti una volta ottenuti gli allegati, terminare la prima parte di analisi.

Diario di lavoro

Luogo	Trevano
Data	09.01.2019

Lavori svolti

Come prima cosa ho ultimato quello che potevo fare con i requisiti. Probabilmente andranno ritoccati nel momento in cui riceverò gli allegati, ma ora almeno c'è una visione di insieme. Ho poi provveduto ad effettuare il Gantt Preventivo del progetto, il quale nonostante le varie ricerche, alla fine ho desistito all'idea del lavorare ad ore, in quanto nessun sito/applicazione trovato permetteva una funzione simile e contemporaneamente di sfruttare la cosa esportandone l'immagine. Di seguito il lavoro ultimato:



Ho poi continuato l'analisi, completando le parti di scopo, analisi del dominio e analisi dei costi e benefici.

Per quanto riguarda l'analisi del dominio, ho dato un'occhiata a ciò che già esiste, ovvero il sito ufficiale di Fsy, <https://www.fsy-europe.org/it>.

Provando a vedere come sono già state gestite le funzioni che dovrò creare nel corso del progetto, mi son reso conto di un po' di cose.

- La traduzione è a volte approssimativa
- se maggiorenne viene dato per scontato che si voglia diventare assistente o consigliere (cosa che personalmente mi pare esagerata, visti i permessi di quest'ultimi, ma dovrò poi parlarne con il mio responsabile).
- L'inserimento dei dati personali, medici e alimentari è forzato, nonostante sembra che i dati siano inseribili pian piano nell'ordine desiderato, così non è.
- Una volta iscritti ad un evento, cancellare l'iscrizione ad una qualsiasi delle fasi comporta il lasciare l'evento ben visibile (forse troppo, essendo mescolato ad altri).
- Non essendoci alcuna gestione dei vari tipi di utente (partecipante, consigliere, assistente), se non tramite età, cambiare quest'ultima anche dopo essersi iscritti come consigliere permette all'utente di avere più ruoli in contemporanea.

<p>Session : International (come consigliere)</p> <p>Completa le informazioni supplementari 0%</p> <p>Scarica il modulo</p> <p>Carica il modulo PDF firmato</p> <p>Inoltra</p> <p>Stato : Cancelled</p> <hr/> <p>Session : Alpenländische Mission (AUT/CHE/DEU) (come partecipante)</p> <p>Completa le informazioni supplementari 0%</p> <p>Scarica il modulo</p> <p>Carica il modulo PDF firmato</p> <p>Inoltra</p> <p>Stato : Mancano le informazioni personali</p>	• •
<ul style="list-style-type: none"> Iscriversi direttamente agli eventi provoca fin troppa confusione all'utente. Per ottenere informazioni a riguardo, deve per forza annullare l'iscrizione e navigare per il sito. Sarebbe quindi comodo un modo per arrivare alla pagina con più informazioni in maniera rapida ed evitando che l'utente si perda all'interno della pagina. <p>Non avendo ancora ricevuto gli allegati, non ho potuto procedere ad una creazione dello scherma E-R definitivo, sebbene abbia chiesto al responsabile e quindi quanto meno definito di come bastino 3 tabelle, una per gli utenti e due sottostanti che utilizzano identificatore esterno dell'utente, che invece rappresentano una i partecipanti e l'altra le categorie, ovvero assistenti e consiglieri.</p> <p>La pagina che ho analizzato abbastanza nel dettaglio sotto consiglio del responsabile la utilizzerò come spunto per molta della grafica e gestione del sito. In questo modo, se il mio lavoro sostituisse quello attuale, gli utenti non si vedrebbero enormi cambiamenti che possono rendere il tutto meno user-friendly, poiché meno familiare.</p>	

Problemi riscontrati e soluzioni adottate
Nessun problema particolare.

Punto della situazione rispetto alla pianificazione
In linea con la pianificazione.

Programma di massima per la prossima giornata di lavoro
Iniziare la progettazione, Schema E-R e implementazione del database.

Diario di lavoro

Luogo	Trevano
Data	15.01.2019

Lavori svolti

Oggi mi son dedicato principalmente a ridefinire alcuni dettagli del progetto tramite domande al mio responsabile, anche grazie agli allegati ricevuti (i quali, essendo pieni di dati effettivi e privati, non mi è possibile rendere pubblici assieme al resto del progetto su GitHub).

Ho avuto modo di riscontrare quanto i dati siano gestiti male all'interno del sito. Essi son infatti salvati all'interno di due file Excel, con una quantità di colonne enorme e una ridondanza di attributi tra le due "tabelle": Ho dovuto quindi analizzarli e estrapolarne la logica, per poterla riadattare alle esigenze. Con il responsabile abbiamo discusso del fatto che nome.cognome non rappresenti abbastanza unicità da essere usata come chiave primaria (come inizialmente in realtà era previsto da quaderno dei compiti), mentre l'email, essendo chi si iscrive un minorenne, spesso due della stessa famiglia utilizzano il medesimo account mail, impedendo quindi di differenziarli.

Si è quindi optato per una chiava primaria composto da nome, cognome e data di nascita.

Inoltre, dovendo il database gestire gli eventi su più anni, è stato concordato l'utilizzo di più tabelle: 1 con i dati primari che rimarranno invariati, e 1 contenente tutti i dati che potrebbero cambiare di anno in anno. Chiaramente sarà offerta la possibilità di riutilizzare i dati precedenti.

Gli utenti a quanto pare hanno la possibilità di iscriversi solamente ad un evento all'anno, e a meno di casi particolari, riguarda sempre quello più vicino (un italiano andrà ad un evento in Italia e non potrà, a meno di casi particolari, andare in uno in Germania). Essendoci tuttavia la possibilità, questo non verrà gestito dall'applicativo, lasciando il compito di decidere se accettare o meno la cosa agli amministratori.

Mi è stato detto dal responsabile di concentrarmi primariamente sull'implementazione della prima parte, riguardante la registrazione e iscrizione, tutta la gestione delle email e lo scaricamento/caricamento dei file PDF. Essendo poi non così chiaro sul quaderno dei compiti originale (il quale dopo questa discussione è stato ristampato, proprio per il procedimento non così chiaro degli eventi):

- La registrazione dell'utente viene effettuata inserendo solamente nome, cognome, data di nascita (queste 3 vengono usate come chiavi primarie all'interno del database).
- Una volta fatto, viene inviata un'email di conferma all'utente. Cliccando sul link presente, potrà attivare il suo account.
- A questo punto, gli amministratori riceveranno un'email contenente i dati dell'utente (non password ovviamente) come segnalazione della sua registrazione.
- L'utente a questo punto potrà inserire i propri dati personali, medici e alimentari. Solo una volta completati tutti i campi obbligatori potrà passare alla fase successiva.
- Gli amministratori riceveranno un'email contenente i dati completi dell'utente. A questo punto dovranno controllarne la validità, per approvarne il contenuto.
- Una volta approvati, l'utente avrà la possibilità di scaricare il/i file PDF e, dopo averli completati a mano (l'utente dovrà quindi stamparseli per conto suo per firmarli), dovrà scannerizzarli e ricaricarli sul sito.
- Ancora una volta, gli amministratori riceveranno un'email con allegati i file dell'utente. Letti e approvati, l'iscrizione dell'utente sarà finalmente completata.

Come ultima cosa, ho chiesto al mio responsabile se quindi andasse bene una delle mie proposte come web hosting. Come la volta scorsa, mi ha detto che utilizzando un web hosting

esterno (a quanto pare si riferiva comunque a infomaniak, usato dalla scuola per i progetti da sempre) TomCat non è supportata come opzione (come già sapevo). Stavolta però, a differenza della volta scorsa, ha accettato la mia volontà di lavorare con Spring Boot, lasciandomi carta bianca riguardo al web hosting alternativo (che chiaramente rimane un problema non indifferente).

Dopo qualche ricerca, ho trovato qualcosa di ben più promettente rispetto alla volta scorsa: si tratta di OpenShift, un applicativo gestito da RedHat. Ho quindi cominciato a documentarmi sull'argomento, cercando di trovare più materiale possibile in modo che, una volta che dovrò provare ad implementare il tutto (se il programma procede come previsto, sarà o la prossima giornata lavorativa o quella dopo ancora), risparmierò tempo e avrò già a disposizione soluzioni da confrontare. Link analizzati/salvati per il futuro:

<https://www.quora.com/What-is-the-most-stable-and-free-host-to-deploy-a-Spring-Boot-web-service-application-and-MySQL-database>

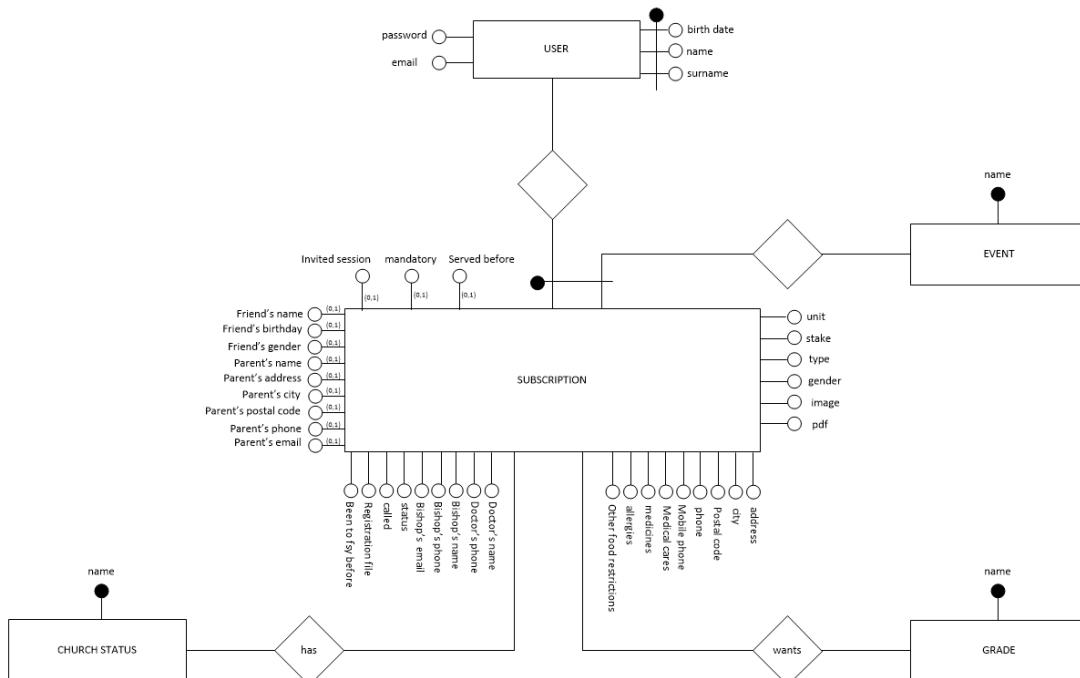
<https://blog.openshift.com/run-your-java-tomcat-application-for-free-on-openshifts-paas/>

<http://www.mastertheboss.com/jboss-frameworks/spring/deploy-your-springboot-applications-on-openshift>

<https://developers.redhat.com/blog/2018/03/27/spring-boot-mysql-openshift/>

<https://www.baeldung.com/spring-boot-deploy-openshift>

Il passo successivo è stato fare lo schema E-R del database. Questo si è rivelato abbastanza confusionario, dovendo includere una quantità enorme di dati. Mi ha preso il restante tempo della giornata, ma sono riuscito a completarlo:



Probabilmente cambierà con il tempo, ed è possibile ci siano ancora delle imperfezioni, ma per il momento direi che è sufficiente.

Problemi riscontrati e soluzioni adottate

Nessun problema particolare, se non difficoltà nel prevedere possibili problemi nella struttura del database.

Punto della situazione rispetto alla pianificazione

Database ancora da creare, ma visione più chiara sul web hosting.

Programma di massima per la prossima giornata di lavoro

Creare database iniziale (meno dati di quelli finali per velocizzare), Iniziare a creare la struttura base, vedere se OpenShift corrisponde a quello che mi serve e, nel caso, implementarlo.

Diario di lavoro

Luogo	Trevano
Data	16.01.2019

Lavori svolti

Ieri non ho notato che per terminare una volta per tutte la prima parte di analisi e progettazione, mi mancassero i mockup delle varie pagine. Per non perderci troppo tempo, li ho schematizzati, cercando di rendere il più chiaro possibile il loro utilizzo e struttura basilare più che una grafica vera e propria.

Di seguito le prime versioni del mockup, non escludendo che si evolvano nel corso del progetto. Come indicato dal mio responsabile, mi sono ispirato alla grafica ora presente all'interno del sito ufficiale di FSY, descritto e analizzato nei giorni precedenti. La grafica non rimanendo un elemento primario del progetto, gli sarà data priorità 2.

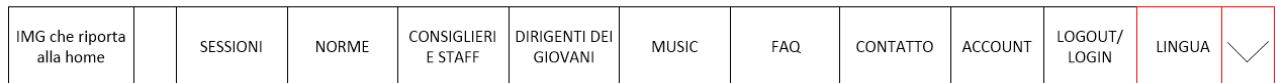


Figura 1 Header di tutte le pagine

L'header delle pagine sarà sempre lo stesso, e sarà fissato in cima per favorire la navigazione. Nel caso in cui lo spazio a schermo non sia sufficiente, tutte le opzioni verranno inserite all'interno di un sandwich menù.

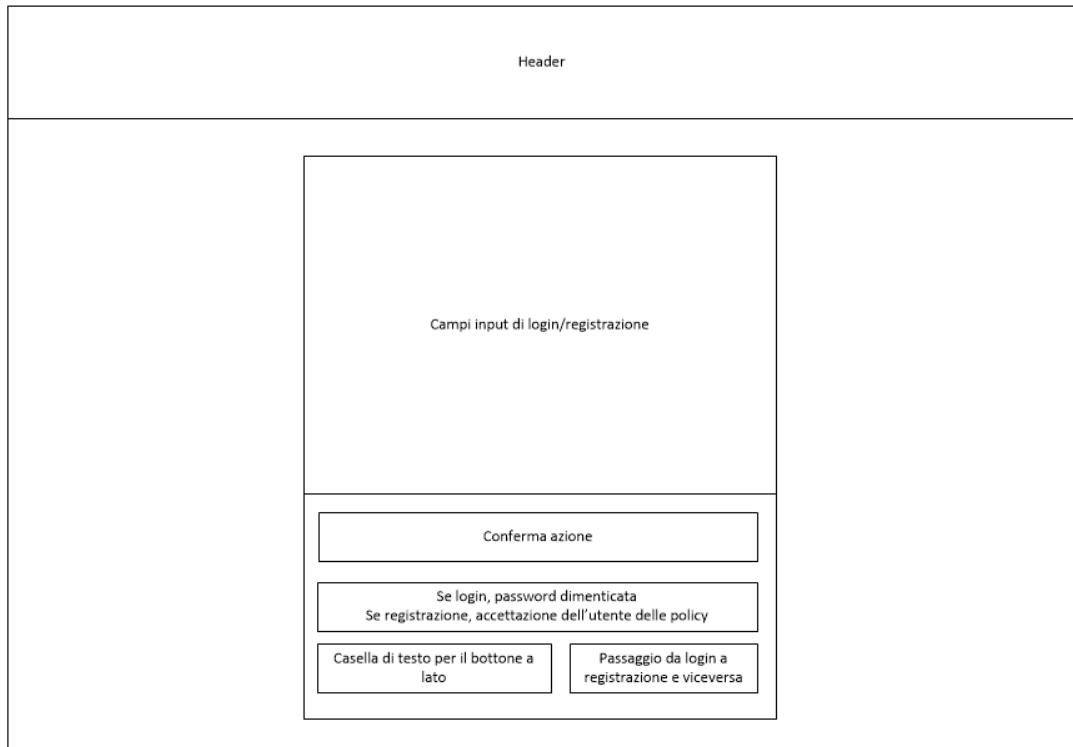


Figura 2 Pagina di login e registrazione

La pagina di login è la stessa della registrazione, cambiando solamente pochi campi e l'azione da compiere.

I dati da inserire saranno per login nome, cognome, data di nascita e password, mentre per la registrazione oltre questi saranno necessari anche l'email e la conferma della password.

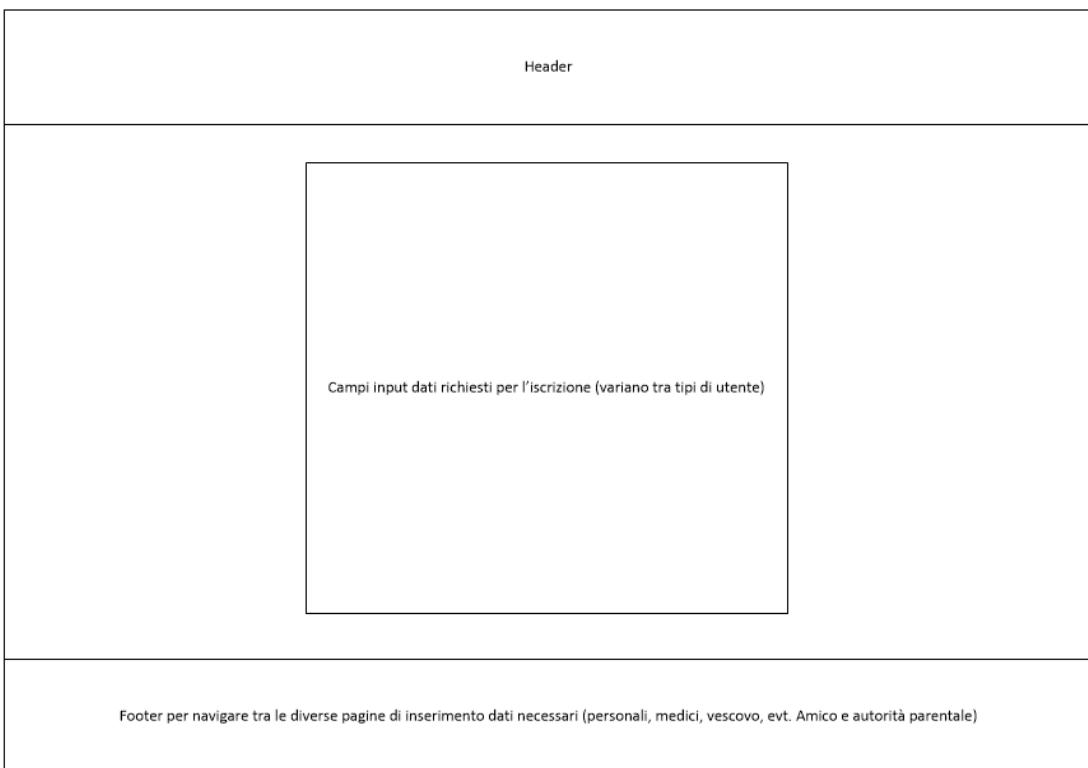


Figura 3 Pagina di inserimento dati

Questa sarà la pagina per inserire i dati dell'utente necessari all'iscrizione. I dati saranno divisi tra dati personali (genere, immagine di profilo, unit, stake, indirizzo, città, codice postale, telefono, cellulare, stato per la chiesa solo se partecipante, presenza passata ad un evento FSY), medici (cure mediche, medicine), alimentari (allergie, restrizioni sul cibo, altre restrizioni sul cibo), informazioni sul vescovo (nome, telefono e email), sul medico (nome e telefono). Nel caso in cui l'iscrizione sia di un partecipante, saranno richiesti anche i dati di un amico partecipante (nome, compleanno, genere) e dell'autorità parentale (nome, indirizzo, città, codice postale, telefono e email). Nel caso invece sia un consigliere o assistente a iscriversi, gli verranno chiesti anche le mansioni per cui si offre (grade(s) wanted), la mandatoria e se ha offerto servizio per FSY negli anni precedenti.

I campi input saranno composti semplicemente in questo modo, per tutti i dati richiesti meno il caricamento dell'immagine di profilo):



Figura 4 Struttura input per i dati

Ho poi provveduto all'effettiva creazione del database, definendo inizialmente solo gli elementi primari, ovvero la tabella utente con attributi nome, cognome, data di nascita (i 3 che formano la chiave primaria della tabella), email, genere e password. Ho poi creato una tabella iscrizione ridotta, inserendo solamente tipo, indirizzo, città, codice postale, telefono e cellulare. Per iniziare già la gestione delle lingue in modo efficace, ho creato inoltre la tabella pagina, contenente al momento solo l'id della pagina, la tabella linguaggio (come attributo solo il nome) e la tabella contenuto, tabella ponte tra le due con in più il parametro testo. Essendo elementi che avevo comunque ignorato nello schema E-R di ieri, ho dovuto risistemarlo in base a ciò che mi è venuto in mente oggi:

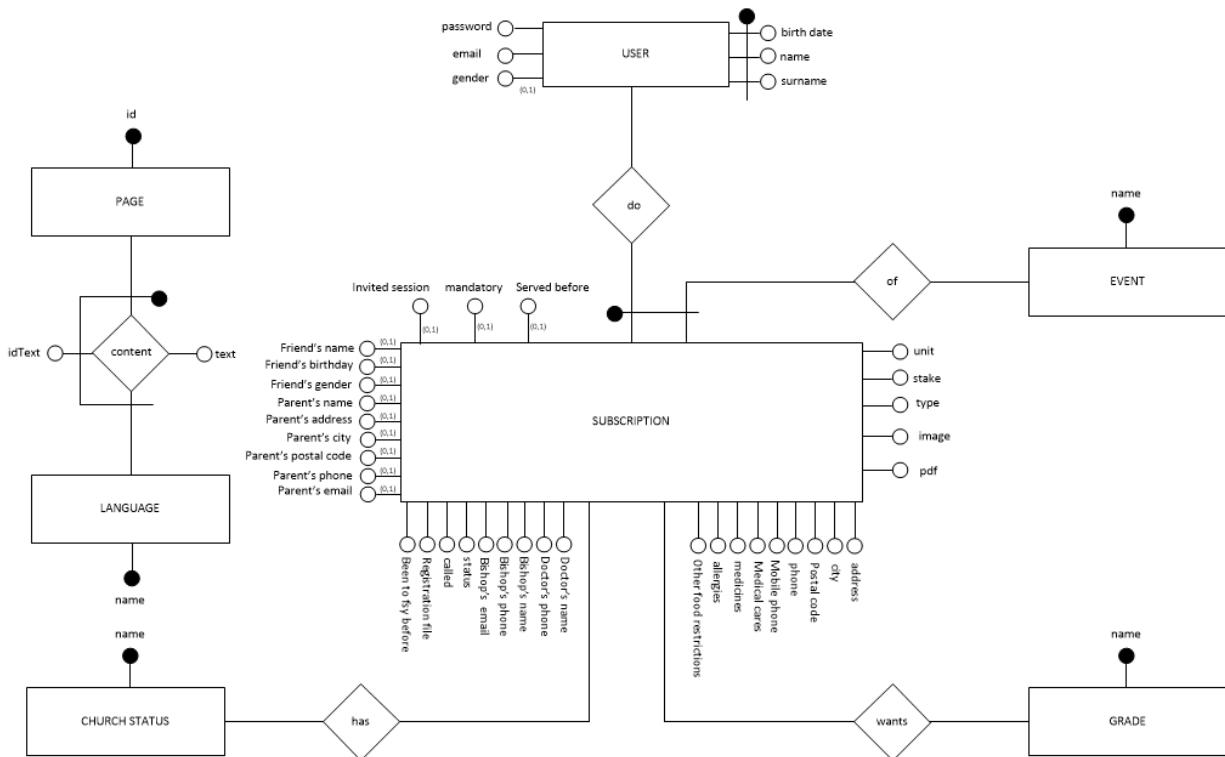


Figura 5 Schema E-R Versione 2.0

Prima di proseguire tuttavia, ho voluto dare ancora un'occhiata agli host che offrono un servizio tomcat (o jetty o undertow), per avere un'idea più chiara. Purtroppo, a livello di hosting gratuiti e non prove temporanee non ho ancora trovato nulla di effettivamente operativo. Ho analizzato meglio le possibilità trovate nei giorni precedenti, ma nessuna di esse è veramente gratuita per un tempo illimitato. Ho analizzato una moltitudine di host alla ricerca di qualcosa di un minimo ed efficiente e gratuito, ma senza successo.

- <https://www.bluehost.com/web-hosting/signup>
- <https://my.a2hosting.com/cart.php>
- <https://www.a2hosting.com/java-hosting>
- <https://www.openshift.com/>
- <https://www.cloudbees.com/products/cloudbees-codeship>
- <https://www.dAILYRAZOR.com/shared-tomcat-hosting/>
- <https://www.mochahost.com/it/java.php>
- <https://www.ktchost.com/global-special-hosting.html>
- <https://www.hostingadvice.com/reviews/cheap/>

Questi sono diciamo i primari che ho controllato al momento. Dovrò rianalizzare la situazione con il mio responsabile, per cercare di trovare una possibile soluzione o un compromesso.

Problemi riscontrati e soluzioni adottate

L'unico vero problema è la questione hosting, che si sta rivelando più ardua del previsto. Potrei quindi essere costretto a rivedere alcune idee di base, così come tutte le strutture che avevo in testa e su qualche foglio di appunti.

Punto della situazione rispetto alla pianificazione

Oggi avrei dovuto creare la struttura base, ma ho preferito dare la precedenza alla ricerca dell'hosting.

Programma di massima per la prossima giornata di lavoro

Vedere con il mio responsabile come procedere, adattarsi e iniziare a creare la struttura base dell'applicativo web.

Diario di lavoro

Luogo	Trevano
Data	22.01.2019

Lavori svolti

La giornata di oggi non è stata particolarmente produttiva, ma fondamentale. Mi ero infatti prefissato il trovare un hosting gratuito per Spring Boot, e l'ho effettivamente trovato. Si tratta di Heroku (<https://dashboard.heroku.com/account>). Negli ultimi minuti della scorsa lezione l'avevo adocchiato, ma non ero riuscito ad analizzarlo nella sua completezza. Ho comunque dato un'occhiata di nuovo al servizio di OpenShift (), per vedere se effettivamente avesse servizi che rimanevano gratuiti, avendo visto da Microsoft Azure () che alcune funzioni rimanevano attive anche senza pagare. Tuttavia, in entrambi i casi il servizio non era disponibile gratuitamente, ma solo in versione limitata (365 days Trial per Azure, OpenShift 14 days Trial).

Dopo qualche problema con la sua effettiva installazione e utilizzo tramite Git, ho capito come in realtà mi stessi complicando la vita e fosse sufficiente collegare una directory GitHub. Questa tuttavia non era sufficiente, in quanto è stato necessario anche crearne una dedicata (chiamata per semplicità FsyManagerApp, senza che questa modifica influenzi il nome del sito).

Il sito tramite il quale ora è possibile accedere al mio lavoro è il seguente: <https://fsy-manager.herokuapp.com/>.

Per caricare l'applicativo, è sufficiente fare un deploy della mia directory GitHub e il sito si aggiorna di conseguenza. È tuttavia necessario inserire la parte di codice riguardante Maven che ho trovato su questo sito <https://www.callicoder.com/deploy-host-spring-boot-apps-on-heroku/>, che permette appunto a Heroku di far avviare tramite plugin il servizio Web Runner, per permettergli di caricare Java. Come ultimo passaggio per far funzionare la cosa, è necessario inserire Java tra i BuildApps dell'account Heroku.

Al momento è presente solamente una struttura base di Spring Boot, creata tramite il sito <https://start.spring.io/>, il quale rende semplicissimo iniziare a sviluppare con questo framework. Avendo avuto tuttavia il colloquio con il mio precedente responsabile Sartori riguardo al progetto dell'anno scorso, cosa che mi ha preso circa 45 minuti, non ho creato una vera struttura.

Mi sono infatti limitato a creare un controller e una view per il login (non ancora disponibile sul sito, solo trovato il template), senza tuttavia crearmi dietro una vera logica. Per il login ho cercato tra i template gratuiti e responsive e ho deciso di utilizzare questo <https://codepen.io/emilcarlsson/pen/XbZprZ>, selezionato dalla pagina che utilizzo di solito, <https://colorlib.com/wp/free-bootstrap-registration-forms/>.

Problemi riscontrati e soluzioni adottate

Ho avuto abbastanza problemi nella mattinata riguardanti Heroku. Questo perché, come accennato, all'inizio seguivo questa guida <https://devcenter.heroku.com/articles/deploying-spring-boot-apps-to-heroku>. Tuttavia si bloccava al login, non permettendomi di collegare l'account. Tra i primi risultati su internet (<https://github.com/heroku/cli/issues/855>). Per fare quanto detto, e quindi disinstallare la versione Client per installarne una normale, ho dovuto prima di tutto capire cosa fosse npm, per poi installarlo tramite questo sito <https://www.npmjs.com/get-npm>. Per usare il comando ho comunque dovuto configurare il rispettivo proxy della scuola (<https://github.com/npm/npm/issues/17246>).

Una volta fatto tuttavia, il problema sussisteva, in quanto al login addirittura non arrivava a chiedermi le credenziali. Avrebbe dovuto aprire una scheda browser, ma ciò non succedeva. Alla fine, ho cercato un'altra soluzione solo per scoprire che era davanti ai miei occhi. Era infatti possibile collegare la cartella GitHub direttamente all'account. Ho avuto comunque qualche problema minore, ma risolto facilmente tramite i passaggi descritti nei "lavori svolti".

Punto della situazione rispetto alla pianificazione

Questa giornata era completamente fuori dalla pianificazione, ma sono comunque soddisfatto e stimo che non sarà affatto difficile recuperare il tempo impiegato nel trovare un host e a fare il colloquio con il mio precedente responsabile.

Programma di massima per la prossima giornata di lavoro

Recuperare il tempo di oggi impiegato in altri modi, quindi creare un effettivo login e registrazione e implementare le rispettive logiche.

Diario di lavoro

Luogo	Trevano
Data	23.01.2019

Lavori svolti

Oggi mi son dedicato completamente alla pagina di login e registrazione.

Implementato il template indicato la scorsa volta, ho avuto non pochi problemi nel fare in modo che me lo leggesse. Questo perché inizialmente avevo semplicemente copiato il codice che mostravano, senza registrarmi per scaricarlo. Dopo qualche tentativo andato a vuoto tuttavia, mi son reso conto di come il gioco non valesse la candela e che pur di evitare di perdere ulteriore tempo valeva la pena registrarsi e scaricarlo. Una volta fatto, i miei problemi, dopo aver cancellato comunque la funzione che dava problemi nel file JS, sono stati risolti. La funzione cancellata non aveva un vero scopo, o quanto meno non gliel'ho trovato. Cancellandola ho comunque verificato che tutte le funzioni primarie fossero quanto meno attive.

Mentre risistemavo il sito adattandolo alle mie esigenze, ho notato come delle icone che mi dava a disposizione mancasse quella per un calendario. Così ho deciso di fare un salto sulla pagina (<https://codyhouse.co/>) che metteva a disposizione le immagini presenti all'interno dei campi input.

Con mia sorpresa, ho scoperto un ottimo sito, che sicuramente mi sarà utilissimo da oggi in avanti. Contiene infatti una buona quantità di template e componenti (sebbene questi non siano modificabili secondo licenza) implementati con una grafica leggera e con sapiente utilizzo di JavaScript, il tutto per rendere il sito ancora più fruibile e user-friendly. È un framework, ma ogni pagina è disponibile presso il loro GitHub, e quindi accessibile e opensource.

Ho scelto i template per alcune delle prossime pagine, ovvero:

Pagina di inserimento informazioni personali: <https://codyhouse.co/demo/breadcrumbs-multi-steps-indicator/index.html#0> per i passaggi e questo (<https://codyhouse.co/demo/contact-form/index.html>) per l'inserimento dei dati.

Pagina di filtraggio dati e possibilità di stampa: <https://codyhouse.co/gem/content-filter>

Inoltre per l'amministratore, che avrà a disposizione molte più funzioni quali le modifiche alle pagine, traduzioni e aggiunta di parametri, pensavo ad un sistema simile <https://codyhouse.co/gem/stretchy-navigation> riguardo alla scelta.

Alla fine ho deciso di utilizzare FontAwesome per le icone, poiché pratico nel suo utilizzo e già conosciuto e utilizzato da me per altri siti.

Sono quindi tornato alla pagina, terminando con successo quanto concerne la grafica di questa prima parte, e implementando già l'inizio della logica, creando la classe User (parametri presenti: nome, cognome, data di nascita, email, genere e password). Il genere, dopo averci riflettuto un po', ho deciso di definirlo come booleano. Lo 0 corrisponde al genere femminile, e l'1 al maschile. Questo mi permette di avere molta più libertà, anche riguardanti l'utilizzo di lingue diverse.

Iniziando ad implementare il codice, mi son reso conto del primo vero problema: avendo una chiave primaria composta da 3 attributi, il sistema da me conosciuto per usare Spring Security diventa inefficace, quanto meno senza le dovute modifiche. Ho quindi cominciato a ricercare una possibile soluzione. Baeldung, uno dei miei siti più fidati riguardanti la programmazione in Spring, mi ha dato questa soluzione <https://www.baeldung.com/spring-security-extra-login-fields>. Tuttavia, è un sistema molto diverso rispetto a quello che uso io, e nel corso della prossima lezione andrà rivisto.

Problemi riscontrati e soluzioni adottate

Iniziale difficoltà nell'implementare correttamente il template all'interno del progetto.

Ho notato un problema di gestione, in quanto al momento Spring Security l'ho configurata in questo modo:

```

24
25
26
27     @Autowired
28     DataSource dataSource;
29
30     @Autowired
31     public void configureGlobal(AuthenticationManagerBuilder auth) throws Exception {
32         auth.jdbcAuthentication().dataSource(dataSource)
33             .authoritiesByUsernameQuery("select email, role FROM user where emailDB=?")
34             .usersByUsernameQuery("select email,password,enabled FROM user where email=?")
35             .passwordEncoder(passwordEncoder());
36     }
37
38     @Bean
39     public PasswordEncoder passwordEncoder() { return new BCryptPasswordEncoder(); }
40
41
42     @Override
43     public void configure(WebSecurity web) throws Exception {
44         web.ignoring().antMatchers("/resources/**");
45     }
46
47     @Override
48     protected void configure(HttpSecurity http) throws Exception {
49
50         http
51             .authorizeRequests() ExpressionInterceptUrlRegistry
52             .antMatchers("/account/**").access( attribute: "isAuthenticated() " ) ExpressionUrlAu
53             .antMatchers("/admin/**").access( attribute: "isFullyAuthenticated() " ) Expression
54             .and() HttpSecurity
55             .formLogin().loginPage("/login").failureUrl("/login?error") FormLoginConfigurer<HttpSecurity>
56             .usernameParameter("username").passwordParameter("password") FormLoginConfigurer<HttpSecurity>
57             .and() HttpSecurity
58             .logout().logoutSuccessUrl("/login?logout") LogoutConfigurer<HttpSecurity>
59             .and() HttpSecurity
60             .csrf().disable() HttpSecurity
61             .exceptionHandling().accessDeniedPage("/403") ExceptionHandlingConfigurer<HttpSecurity>
62             .and() HttpSecurity
63             .sessionManagement() SessionManagementConfigurer<HttpSecurity>
64             .sessionCreationPolicy(SessionCreationPolicy.IF_REQUIRED);
65     }

```

L'ultima funzione è quella principale, che controlla il sistema e gestisce le richieste. Verrà poi descritto accuratamente all'interno della documentazione.

Il problema sussiste col fatto che, come si può notare, i parametri utilizzati sia in quel metodo che nel primo sono 2, email (come username) e password. Ora invece la mia chiave primaria è composta da ben 3 parametri, e non essendoci la possibilità di utilizzare altri parametri devo valutare meglio la situazione e, eventualmente, sostituire il sistema di gestione Spring Security (esistono infatti moltissimi standard per esso, questo che uso al momento è semplicemente quello con cui mi son trovato meglio per lavorare).

Punto della situazione rispetto alla pianificazione

Dao e Mapper, non avendo ancora creato la vera logica dietro, non sono stati creati. Il resto è stato recuperato. Questa è dovuto più ad un mio errore di valutazione nella pianificazione, in quanto ha molto più senso al momento creare prima tutta la struttura per registrazione e login e solo in seguito pensare allo sviluppo delle restanti funzioni.

Programma di massima per la prossima giornata di lavoro

Finire la pagina di registrazione/login con logica implementata.

Diario di lavoro

Luogo	Trevano
Data	23.01.2019

Lavori svolti

Oggi è stata una giornata meno proficua del solito. Essendo infatti il primo giorno lavorativo composto da solamente 4 ore, mi son dovuto gestire diversamente con il diario. Inoltre, il problema che avevo notato la scorsa volta si è rivelato più impegnativo del previsto. Ero convinto di poter trovare una soluzione alternativa e cambiare il sistema di Spring Security, ma ne è risultato che tutti usino un sistema uguale se non molto simile, che comunque non prevede l'utilizzo di più campi.

Non trovando i risultati sperati, ho quindi pensato a modellare la domanda, in modo da ottenere risultati simili a quelli sperati senza dover utilizzare più parametri. Ho infatti cominciato a ricercare il modo di utilizzare, all'interno di Spring Security, un filtro.

Dopo aver esplorato un po' il web tuttavia mi son reso conto che questa strada non mi avrebbe permesso comunque di svolgere il lavoro come desiderato. Il link primario controllato è:

<https://www.baeldung.com/spring-security-custom-filter>.

Nel frattempo, ho provveduto a creare il Dao dell'utente, ovvero il mezzo tramite il quale andrà in futuro a interrogare il database tramite metodi specifici e chiusi (potendo così gestire appieno SQL Injections, errori di valori nulli e al contempo non dare la possibilità a nessuno di eseguire query non necessarie).

La pagina che ho cercato di replicare, senza tuttavia arrivare ad una soluzione, è questo:

<https://www.concretewebpage.com/spring/spring-security/spring-mvc-security-jdbc-authentication-example-with-custom-userdetailsservice-and-database-tables-using-java-configuration>

Alla fine, per evitare una perdita di efficienza lavorativa, ho deciso di ripiegare a ciò che in fondo avrei potuto pensare di implementare fin dall'inizio: un campo unico denominato id all'interno dell'utente, che farà da ridondanza unendo i campi nome, cognome e data di nascita, separandoli tramite l'utilizzo di un “.”.

Ho quindi effettuato il cambiamento a livello di database, per poi tuttavia essere costretto a rimandare il completamento e verifica di funzionamento alla prossima giornata lavorativa.

Problemi riscontrati e soluzioni adottate

Il problema principale è stato proprio la mancanza di risultati riguardo all'utilizzo di Spring Security specifico e adatto alla mia situazione. Alla fine ho risolto proprio rinunciando alla possibilità e creando una ridondanza all'interno del database, che però mi permetterà di alleggerire alcune operazioni e la tabella sottostante "subscription", quindi direi che in fondo anche rinunciando a qualcosa non c'è stata una vera e propria perdita di qualità ed efficienza del prodotto.

Punto della situazione rispetto alla pianificazione

La scorsa volta non avevo pensato al fatto che questa sarebbe stata la mia prima giornata di solamente 4 ore lavorative. Pertanto, l'implementazione del login non è ancora stata completata, sebbene abbia già creato tutta la logica quantomeno del login tralasciando la sua gestione tramite Spring Security, e abbia già predisposto i mezzi necessari al funzionamento effettivo della registrazione.

Programma di massima per la prossima giornata di lavoro

Terminare la logica di implementazione della prima parte, modificare lo schema E-R una volta completato il tutto in modo da avere una versione da poter definire quasi definitiva di quest'ultimo e strutturare l'invio effettivo delle email.

Diario di lavoro

Luogo	Trevano
Data	29.01.2019

Lavori svolti

Oggi ho finalmente terminato la parte di implementazione riguardante il login e fatto gran parte della registrazione, anche se non senza qualche problema. Infatti, inizialmente Spring Security non funzionava correttamente. Alla fine tuttavia si è rivelato un problema di template, e ho quindi provveduto a cercarne un altro, scegliendo questo:

Per le date, poiché il campo di input di tipo date non supporta i placeholder, e tenendo da conto che non in tutte le pagine funziona a dovere, ho preferito implementare un widget di Javascript che rendesse il tutto più comodo e malleabile. Parlo di datepicker (<https://jqueryui.com-datepicker/#dropdown-month-year>), che appunto grazie al fatto di essere personalizzabile, ho potuto definire un range di anni in modo che l'utente non possa fare la registrazione al sito se più grande di 50 anni (anno indicativo, più che altro per permettere il select fino a molti anni indietro) o minore di 13 anni (un anno in meno l'età da cui è disponibile iscriversi agli eventi, in modo di permettere alle persone di iscriversi già tempestivamente). Sono comunque date temporanee, potrei modificarle in qualunque momento, così come rendere la scelta direttamente a quello che poi sarà l'admin della pagina.

Il range di anni è definito tramite il campo "rangeLimit", trovato su questo link:

<https://stackoverflow.com/questions/14228151/datepicker-plugin-for-birth-date-textbox>.

Per quanto riguarda alla pagina di login, al momento è questa:

The screenshot shows a clean, modern login interface. The title 'Log In' is at the top. Below it are four input fields: 'First Name' and 'Last Name' side-by-side, followed by 'Birth Date' with a placeholder 'Birthdate', and 'Password'. Underneath these fields are two buttons: 'Remember me' (unchecked) and 'Forgot the password? Recover it here!'. At the bottom of the form is a blue 'Login' button. Below the form, a small note says 'You don't have an account?' followed by a blue 'Register' button.

Ha ancora qualche difetto a livello di grafica, dovuto principalmente ad una comprensione non ancora completa del template utilizzato, che per quanto leggermente modificato contiene fin troppe righe di css. Inoltre, i controlli sono temporaneamente ignorati, per poi essere gestiti, nel caso della registrazione, direttamente all'interno del codice. Per quanto riguarda Spring Security invece, offre un controllo completo, come ho potuto anche testare tramite il seguente test:

Log In

First Name

a; insert into user (id, name) values (1,2);

Il login funziona tramite l'utilizzo di Javascript, che scrive in un campo input nascosto e in sola scrittura la chiave primaria per eseguire il login, nel giusto formato che chiaramente non interessa all'utente.

Il codice utilizzato per eseguire il login è semplicemente l'abilitazione di Spring Security, mediante le seguenti righe:

```

20  @Configuration
21  @EnableWebMvcSecurity
22  @EnableGlobalMethodSecurity(securedEnabled = true)
23  public class SecurityConfig extends WebSecurityConfigurerAdapter {
24
25      @Autowired
26      DataSource dataSource;
27
28      @Autowired
29      public void configureGlobal(AuthenticationManagerBuilder auth) throws Exception {
30          auth.jdbcAuthentication().dataSource(dataSource)
31              .authoritiesByUsernameQuery("select id, role FROM user where id=?")
32              .usersByUsernameQuery("select id,password,enabled FROM user where id=?")
33              .passwordEncoder(passwordEncoder());
34      }
35
36      @Bean
37      public PasswordEncoder passwordEncoder() { return new BCryptPasswordEncoder(); }

```



```

44
45  @Override
46  protected void configure(HttpSecurity http) throws Exception {
47      http
48          .authorizeRequests() ExpressionInterceptUrlRegistry
49          .antMatchers("/*").access(attribute: "isAuthenticated()")
50          .and() HttpSecurity
51          .formLogin().loginPage("/login").failureUrl("/login?error") FormLoginConfigurer<HttpSecurity>
52          .usernameParameter("username").passwordParameter("password") FormLoginConfigurer<HttpSecurity>
53          .and() HttpSecurity
54          .logout().logoutSuccessUrl("/login?logout") LogoutConfigurer<HttpSecurity>
55          .and() HttpSecurity
56          .csrf().disable() HttpSecurity
57          .exceptionHandling().accessDeniedPage("/403") ExceptionHandlingConfigurer<HttpSecurity>
58          .and() HttpSecurity
59          .sessionManagement() SessionManagementConfigurer<HttpSecurity>
60          .sessionCreationPolicy(SessionCreationPolicy.IF_REQUIRED);
61

```

Queste righe di codice erano già state mostrate nel corso del diario, ma si trattava di una versione

non ancora verificata. Per questo motivo sono state reinserite e, per evitare di commettere lo stesso errore, l'implementazione della registrazione, ormai concluso se non per qualche controllo e popup di allerta per l'utente, verranno mostrate e spiegate brevemente la prossima volta.
È poi necessario creare una classe che estende , in questo modo:

```
3 import org.springframework.security.web.context.AbstractSecurityWebApplicationInitializer;
4
5 public class SecurityWebInitializer extends AbstractSecurityWebApplicationInitializer {}
```

Chiaramente, è necessario essere prima collegati al database di supporto, che al momento è in localhost. Il codice per fare ciò, implementato già nelle scorse giornate ma non segnalato su diario, è il seguente:

```
10 @Configuration
11 public class JdbcConfig {
12     @Bean
13     public DataSource mysqlDataSource() {
14         DriverManagerDataSource dataSource = new DriverManagerDataSource();
15         dataSource.setDriverClassName("com.mysql.jdbc.Driver");
16         dataSource.setUrl("jdbc:mysql://localhost:3306/fsy");
17         dataSource.setUsername("root");
18         dataSource.setPassword("");
19         return dataSource;
20     }
21     @Bean
22     public JdbcTemplate toJdbc() {
23         JdbcTemplate jdbct = new JdbcTemplate(mysqlDataSource());
24         return jdbct;
25     }
26
27
28 }
```

Nel momento in cui dovrò spostare il database su un gestore esterno, sarà sufficiente aggiornare la parte riguardante all'url e avere la certezza che la porta utilizzata per mysql sia abilitata.
La funzione remember me e password dimenticata al momento sono senza logica dietro, in quanto secondarie per il progetto (remember me lo potrei definire anche terziario, penso).
La registrazione ha un'interfaccia molto simile, ovvero:

The screenshot shows a registration form titled "REGISTER FORM". It includes fields for First Name, Last Name, Birth Date, Birthdate, Your Email, Password, and Confirm Password. There is also a checkbox for accepting terms and conditions, which is currently unchecked. A blue "Register" button is at the bottom.

I AUTHORIZE THE CHURCH OF JESUS CHRIST OF LATTER-DAY SAINTS AND ITS AFFILIATED ENTITIES (THE CHURCH) TO COLLECT, PROCESS, AND TRANSFER TO OTHER COUNTRIES MY PERSONAL INFORMATION AS IT MAY BE REQUIRED FOR CHURCH PURPOSES AND IN ACCORDANCE WITH THE CHURCH'S RECORDS MANAGEMENT AND CONFIDENTIALITY POLICIES.

Register

Questa, a differenza del login, ha una logica completamente implementata da me, della quale però non sono ancora certo del funzionamento effettivo e rimando alla prossima volta per questioni di tempistica.

Problemi riscontrati e soluzioni adottate

Il problema principale della giornata è stata proprio riguardo all'utilizzo di Spring Security, che non voleva prendere l'evento come valido (necessita di una semplice richiesta post alla pagina di login a lui segnalata). Dopo molti tentativi e ricerche, ho scoperto che il problema derivava in qualche modo dal template. Infatti, tornando ad una struttura base di solo due campi senza grafica, funzionava senza alcun problema. Per evitare di perdere altro tempo oltre l'ora dedicata al problema, ho quindi deciso di cambiare template, come indicato nello svolgimento della giornata.

Punto della situazione rispetto alla pianificazione

Oggi avrei dovuto terminare il tutto e iniziare la gestione delle email. Sono indietro, ma solo di circa mezza giornata, ed avendo ormai una base abbastanza solida riguardante le email, non dovrei metterci molto a recuperare il tempo perso in problemi vari.

Programma di massima per la prossima giornata di lavoro

Testare il funzionamento della registrazione, implementare controlli e popup per l'utente, testare il pacchetto completo di login/registrazione documentando il tutto sulla documentazione, iniziare ad implementare le email e i loro passaggi.

Diario di lavoro

Luogo	Trevano
Data	30.01.2019

Lavori svolti

Oggi sono stato assente per problemi di salute, e quindi il progetto non è avanzato.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Ho perso 4 ore sulla programmazione, ma ho fiducia sul fatto di recuperarle in fretta e che questo non vada ad inficiare sulla qualità del prodotto finale.

Programma di massima per la prossima giornata di lavoro

Testare il funzionamento della registrazione, implementare controlli e popup per l'utente, testare il pacchetto completo di login/registrazione documentando il tutto sulla documentazione, iniziare ad implementare le email e i loro passaggi.

Diario di lavoro

Luogo	Trevano
Data	04.02.2019

Lavori svolti

Oggi sono stato assente per problemi di salute, e quindi il progetto non è avanzato.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Ho perso altre 4 ore sulla programmazione, ma ho fiducia sul fatto di recuperarle in fretta e che questo non vada ad inficiare sulla qualità del prodotto finale.

Programma di massima per la prossima giornata di lavoro

Testare il funzionamento della registrazione, implementare controlli e popup per l'utente, testare il pacchetto completo di login/registrazione documentando il tutto sulla documentazione, iniziare ad implementare le email e i loro passaggi.

Diario di lavoro

Luogo	Trevano
Data	05.02.2019

Lavori svolti

Oggi finalmente mi son potuto dedicare al verificare il funzionamento della registrazione. Questo ha richiesto più tempo del previsto in quanto ho provato a strutturare il tutto tramite metodi più immediati e un uso più completo delle prepared statement.

Ho inoltre rivisto ancora una volta lo schema E-R, aggiungendo un ulteriore campo all'utente: il link. Questo permette di utilizzare un link, che altro non è che una stringa casuale di 10 caratteri assegnata al momento della registrazione dopo aver confermato la sua unicità all'interno del database, per abilitare l'accaccount. In questo modo, l'utente non si ritrova la sua chiave primaria all'interno di qualcosa di vulnerabile come un url in metodo get. Questo link, essendo comunque in grado di selezionare l'utente, viene cancellato al momento della sua conferma. In questo modo, lo spazio occupato, nonché il numero di possibilità presenti (le quali comportano una maggiore difficoltà, con grandi numeri, a trovare una stringa univoca), diminuisce.

La struttura finale è la seguente:

Metodo di gestione registrazione:

Metodo di creazione utente:

```

30     public boolean createUser(User u) {
31         if(u != null) {
32             if(getUserByIds(u.getName(), u.getSurname(), u.getBirthDate()) != null) {
33                 return false;
34             }
35             String id = u.getName()+"."+u.getSurname()+"."+dateFormat(u.getBirthDate());
36             String sql = "insert into user (id, name, surname, birthDate, email, password, link) " +
37                         "values (?, ?, ?, ?, ?, ?, ?)";
38             return this.doChangeFilter(sql, id, u.getName(), u.getSurname(),
39                                         dateFormat(u.getBirthDate()), u.getEmail(), u.getPassword(), u.getLink());
40         }
41
42         return false;
43     }
44 }
```

Metodo che gestisce la richiesta sql tramite PreparedStatement:

```

100    private boolean doChangeFilter(String sql, String... params) {
101        return jdbcTemplate.execute(sql, new PreparedStatementCallback<Boolean>() {
102            @Override
103            public Boolean doInPreparedStatement(PreparedStatement ps) throws SQLException, DataAccessException {
104                for(int i = 0; i < params.length; i++) {
105                    ps.setString(parameterIndex: i+1, params[i]);
106                }
107                return ps.execute();
108            }
109        });
110    }
```

Questo metodo ancora mi ritorna false, nonostante l'operazione va a buon fine. Devo ancora trovarne il motivo, ma per ora lo trovo un problema secondario.

Metodi di abilitazione account (il primo nel controller, il secondo nel Dao):

```

89     @RequestMapping(value = "/abilitate/{link}", method = RequestMethod.GET)
90     public ModelAndView setEnabled(@PathVariable String link) {
91
92         ModelAndView model = new ModelAndView();
93         User u = userDao.getUserByLink(link);
94         if(u != null){
95             userDao.setEnabled(link);
96         }
97         model.setViewName("redirect:/login?confirmed");
98         return model;
99
100    }
118
119     public boolean setEnabled(String link){
120         String sql = "update user set enabled = 1 where link =?";
121         String sql2 = "update user set link = '' where link =?";
122         return (this.doChangeFilter(sql, link) && this.doChangeFilter(sql2, link));
    }

```

Ho poi ristrutturato allo stesso modo anche il Dao dal punto di vista delle richieste di select, con le opportune modifiche, quali:

```

70     public User getUserByIds(String name, String surname, LocalDate date) {
71         String sql = "select * from user where id = ? ";
72         String id = name+surname+dateFormat(date);
73         List<User> users = this.doSelectFilter(sql, id);
74         return getSingleUser(users);
75
76    }
77
78     @
79     private User getSingleUser(List<User> users) {
80         if(users != null){
81             if(users.size() > 0){
82                 return users.get(0);
83             }
84         }
85         return null;
    }
87
88     private List<User> doSelectFilter(String sql, String... params){
89         return jdbcTemplate.query(
90             sql, new PreparedStatementSetter() {
91                 public void setValues(PreparedStatement preparedStatement) throws SQLException, DataAccessException {
92                     for(int i = 0; i < params.length; i++){
93                         preparedStatement.setString(parameterIndex: i+1, params[i]);
94                     }
95                     preparedStatement.execute();
96                 }
97             },
98             new UserMapper());
    }

```

A questo punto ho potuto finalmente gestire i possibili errori che possono essere riscontrati. Per il login, gli unici errori possibili sono un account non registrato oppure uno non abilitato. Non avendo veramente un modo comodo di differenziare i due stadi, ho deciso di far notare

semplicemente la possibilità che la registrazione sia presente ma che l'account non sia abilitato. In questo modo, non è possibile effettuare dei controlli non permessi dello stato di registrazione di altri utenti conoscendone i dati personali. Per lo stesso motivo, non viene fatta la differenza tra errori per id e password.

Per quanto riguarda la registrazione invece, c'è solamente la possibilità che l'utente sia già registrato da gestire.

Sono poi da gestire la validità di quanto viene inserito nei campi input, di entrambe le fasi: nome e cognome sono poi controllati tramite JS con regular expression, per permettere l'inserimento di caratteri che non vengano inseriti numeri e caratteri speciali (vengono però tenuti da conto lettere con accenti, come cediglie o simili). La regular expression in questione è stata presa da il seguente sito: <https://stackoverflow.com/questions/150033/regular-expression-to-match-non-english-characters> (soluzione nei commenti della risposta).

Il campo email invece usa una regular expression tutta sua, che è stata per semplicità presa da un sito (<https://emailregex.com/>).

```

14 function checkRegex(text, type, errorBox){
15   if(type == 'email'){
16     var patt = /^(([^>0\|\|.\|;:\s@"]+(\.|[^>0\|\|.\|;:\s@"]+)*|".+"))@((\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})|(((a-zA-Z-\d)+\.)+[a-zA-Z]{2,}))$/;
17   }else{
18     var patt = /^[\u00C0-\u017E a-zA-Z]+$/;
19   }
20   var check = patt.test(text);
21   if(check || text === ""){
22     document.getElementById(errorBox).style.visibility = "hidden";
23     updateUser();
24   }else{
25     document.getElementById(errorBox).style.visibility = "visible";
26   }
27 }
28 }
```

Per finire ho effettivamente implementato la parte riguardante alle email, sebbene non abbia ancora potuto effettivamente testarla in quanto la porta smtp viene bloccata dal proxy. Questo problema, assieme alla mancanza di un effettivo spazio per il database esterno, nascono principalmente dal fatto che avrei dovuto richiedere i due elementi molto prima, ma che alla fine rimandavo. Ho provveduto solo oggi, nel mentre vedrò di verificare le email fuori dall'orario lavorativo e il database, così come di conseguenza il sito, solo in locale (cosa effettivamente fatta, sebbene non penso serva a molto creare un test-case dedicato, tenendomi quest'ultimo per quando potrò testare tutto online).

Problemi riscontrati e soluzioni adottate

Nessun problema in particolare.

Punto della situazione rispetto alla pianificazione

Reduce delle 8 ore perse, ho ancora un po' da recuperare, ma se il codice per l'email funziona come deve direi che sono praticamente in pari.

Programma di massima per la prossima giornata di lavoro

Testare email, creare home, implementare multilingua.

Diario di lavoro

Luogo	Trevano
Data	06.02.2019

Lavori svolti

Oggi ho confermato la funzionalità della funzione delle email, e proceduto con la creazione di alcuni step che vengono poi effettuati tramite l'utilizzo di missive.
Il funzionamento delle email è la seguente:

Bean di configurazione:

```

13  @Bean
14  public JavaMailSender getJavaMailSender() {
15      JavaMailSenderImpl mailSender = new JavaMailSenderImpl();
16      mailSender.setHost("smtp.gmail.com");
17      mailSender.setPort(587);
18      mailSender.setUsername("samt.fsymanager@gmail.com");
19      mailSender.setPassword("password dell'email");
20      Properties props = mailSender.getJavaMailProperties();
21      props.put("mail.transport.protocol", "smtp");
22      props.put("mail.smtp.auth", "true");
23      props.put("mail.smtp.starttls.enable", "true");
24      props.put("mail.smtp.starttls.required", "false");
25      props.put("mail.debug", "true");
26      props.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
27
28      return mailSender;
29  }

```

Classe per l'utilizzo delle email:

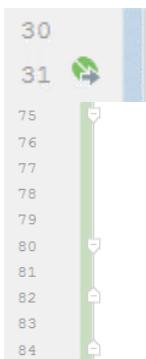
```

11  @Service
12  public class EmailSenderImpl {
13
14      @Autowired
15      private JavaMailSender javaMailSender;
16
17      public void sendingMail(String to, String subject, String body) throws MessagingException {
18          MimeMessage message=javaMailSender.createMimeMessage();
19          MimeMessageHelper helper;
20          try {
21              helper=new MimeMessageHelper(message, multipart: true);
22              helper.setTo(to);
23              helper.setSubject(subject);
24              helper.setText(body, html: true);
25              javaMailSender.send(message);
26
27          } catch (javax.mail.MessagingException e) {
28              e.printStackTrace();
29          }

```

È importante sottolineare che MimeMessage proviene dalla libreria javax.mail.internet e non da quello presente in springframework.

Utilizzo effettivo (esempio):



```

30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84

```

```

    @Autowired
    private EmailSenderImpl emailSender;
    try {
        emailSender.sendingMail(user.getEmail(), subject: "Welcome "+name,
            body: "Click this link for confirm: <a href='localhost:8080/abilitate/"+user.getLink()+">Click here to confirm your email<br>" +
            "localhost:8080/abilitate/"+user.getLink()+"</a>");

    } catch (MailException e) {
        e.printStackTrace();
    } catch (MessagingException e) {
        e.printStackTrace();
    }
}

```

Ho poi riscontrato un problema, ovvero la mancanza di Gmail di interpretare l'html nelle email, ma ho evitato di perderci dietro troppo tempo in quanto non è un problema primario e sono già indietro sulla pianificazione.

Il primo passaggio consiste nell'inviare un'email di avviso a tutti gli admin nel momento in cui un utente conferma la propria email. Chiaramente all'interno della missiva vengono indicati i dati primari dell'utente in questione.

Il secondo passaggio, sebbene non sia ancora implementata la seconda fase del progetto che farà poi scattare questo passaggio, è l'invio dei dati personali che l'utente avrà compilato una volta eseguito l'accesso. Nell'email saranno presenti due semplici versioni, ovvero l'accettazione dei dati oppure una lista dei campi da segnalare come inappropriati. Nel primo caso, l'utente potrà passare alla fase successiva (terza fase del progetto), in caso contrario dovrà sistemare i propri dati in base a quanto specificato nell'email che gli arriverà, e dovrà ripetere il processo. Qualunque sia il caso comunque, nel momento in cui un amministratore effettuerà una delle due scelte tutti gli amministratori riceveranno un'email che gli informerà che il controllo è stato gestito (specificando anche chi lo ha fatto). Non era prettamente nel quaderno dei compiti, ma mi è sembrato un passaggio fondamentale una volta ragionato sul problema. Nel caso vada gestito in modo diverso provvederò dopo averne discusso con il mio responsabile.

Problemi riscontrati e soluzioni adottate

L'unico problema di oggi l'ho riscontrato nell'interpretazione da parte dei vari sistemi di mail (ho provato con Gmail e Outlook). Il primo svuotava solamente l'attributo href, mentre il secondo mi rimuoveva l'intera parte del tag, lasciando comunque ciò che c'era tra l'apertura e la chiusura del tag. Dopo aver cercato un po', ho avuto la conferma del fatto che non fosse un problema di programmazione: è infatti presente una variabile booleana nell'impostazione della classe, che definisce esattamente l'abilitazione o meno dell'interpretazione della stringa inviata come parte html.

Ho quindi pensato ci fosse qualche regola che non conoscessi riguardo all'utilizzare html nelle missive, ma questo sito https://support.e2ma.net/Resource_Center/Account.How-to/basic-format-for-your-own-html-emails mi ha dimostrato che in realtà non era così, e che dovrebbe essere abbastanza intuitivo.

Come ultimo tentativo, ho provato a cercare possibili impostazioni o modi per risolvere questo problema nei due sistemi di mail indicati prima, ma mi son fermato a metà: questo perché se non è un'impostazione base, non posso aspettarmi che gli utenti ce l'abbiano, e quindi anche se riuscissi magari a risolverla per i miei client, lo stesso non varrebbe per gli altri utilizzatori. Ho

quindi deciso di accantonare il problema, in modo da evitare la perdita di ulteriore tempo prezioso già perso per questioni salutari.

Punto della situazione rispetto alla pianificazione

Oggi avrei dovuto completare le pagine home (sulle quali ancora non ho ancora un piano ben preciso in mente, penso rimarranno basilari) e l'iscrizione al progetto. Sono quindi indietro di 2 giorni rispetto al programma.

Programma di massima per la prossima giornata di lavoro

Iniziare la struttura e terminare la logica della creazione della pagina per l'iscrizione.

Diario di lavoro

Luogo	Trevano
Data	11.02.2019

Lavori svolti

Oggi mi son dedicato nella strutturazione dei passaggi di iscrizione consegna documento. Nello specifico, ho creato i metodi che inviano le email indicate nel corso dello scorso diario e predisposto il funzionamento delle pagine dalle quali sarà poi possibile controllare i dati e selezionare quali sono idonei e quali invece l'utente dovrà reinserire.

Per prima cosa però, ho aggiornato velocemente il mio responsabile, spiegandogli la situazione delle email al momento, dove anche lui mi ha intimato a evitare di perdere tempo con la struttura interna delle email in quanto la parte fondamentale è il loro funzionamento. Ho quindi richiesto uno spazio per inserire i dati su un sito esterno, in modo da poter finalmente aggiornare e testare le funzionalità del sito hostato su Heroku, e non solo tramite localhost.

Le suddette pagine, sebbene non ancora create, saranno semplicemente una struttura simile se non uguale al form di inserimento dei dati, con l'aggiunta di un campo input checkbox per ogni campo, che saranno selezionati di default, in modo da far deselezionare all'amministratore i campi che non sono accettabili. Questi verranno poi identificati dal codice che ho scritto oggi, in modo da poter indicare all'utente in modo preciso i dati che necessita di reinserire. Nel caso non vengano segnalate modifiche da parte dell'amministratore, l'iscrizione sarà accettata come tale e si passerà alla fase successiva, quella del file pdf, la quale utilizzerà il medesimo procedimento.

Viene poi data la possibilità di inserire delle note dell'amministratore, che possono essere richieste più specifiche o simili. L'intero procedimento di risposta verrà inviato sia all'utente che agli amministratori, che potranno quindi vedere il caso come gestito (nell'apposita tabella contenente tutti gli utenti, filtrabile) così come avvertiti tramite email, tramite la quale potranno vedere anche il modo in cui l'iscrizione è stata gestita dal collega, in modo da valutarne l'operato.

Ho poi iniziato a dividere il codice tra più Controller, per tenere il codice più pulito e ordinato. EsyController ha una mappatura standard (""), ViewController ("view") conterrà tutti i metodi per visualizzare utenti, liste e simili. Queste funzioni saranno disponibili solamente per admin, consiglieri e assistenti. E infine, AdminController, nel quale ci saranno tutti i metodi per i quali sarà necessario avere i permessi di amministratore. Al momento la separazione è questa, sebbene al momento alcuni metodi debba purtroppo riprenderli tra i vari Controller, che non possono comunicare tra loro se non tramite metodi e costanti statici (che non sono quelli di cui ho bisogno al momento).

Problemi riscontrati e soluzioni adottate

Nessun problema in particolare.

Punto della situazione rispetto alla pianificazione

Oggi avrei dovuto terminare la creazione dei file pdf, quando ancora sono fermo alle email.

Tuttavia, questo ritardo è dovuto più che altro alle assenze delle scorse settimane e il fatto di aver sottovalutato la mole di codice richiesto per le email, che mi stanno riempiendo le giornate.

Programma di massima per la prossima giornata di lavoro

Terminare finalmente la struttura delle email, iniziare la pagina per l'iscrizione al sito.

Diario di lavoro

Luogo	Trevano
Data	12.02.2019

Lavori svolti

Oggi è stata una giornata veramente poco produttiva rispetto a quanto sarebbe dovuto essere. Il motivo è l'enorme perdita di tempo dovuta, ancora una volta, all'hosting. Dopo molto tempo usato nel cercare di collegarmi direttamente al web hosting su myd messomi a disposizione, ho chiesto al mio responsabile chiarimenti. Così, ho scoperto che il 99% degli hosting presenti su internet permette l'utilizzo del proprio spazio per i database solamente da localhost, e quindi dalla piattaforma stessa (in questo caso quindi, infomaniak). Questo si è rivelato un grande imprevisto, dato che chiaramente lavorando in Java avevamo già scartato tempo fa l'utilizzo di infomaniak. La mia richiesta dello spazio si basava dunque sull'ignoranza di questo dettaglio, mentre il mio responsabile ha semplicemente fornito quanto richiesto senza porsi o pormi domande.

Il restante tempo della mattinata è stato dunque impiegato nel provare ad implementare la versione MySQL di Heroku, ovvero il web hosting che utilizzo al momento. Purtroppo, non sono andato lontano. Proseguendo nei passaggi trovati sul sito stesso, avrei dovuto installare un build-app presente, CleanDB Heroku. Per installare una qualsiasi applicazione, per quanto gratuita come in questo caso, a quanto pare è necessario inserire la propria carta di credito. Dopo aver provato ad inserirne una falsa creata tramite un sito, mi ha segnalato errore. Probabilmente si rende conto quando questa sia falsata.

Ho quindi cominciato a ricercare nuovamente un sistema di web hosting che potesse ospitare sia un applicativo Java che un database MySQL.

Ho trovato qualcosa. Il primo è stato , il quale però richiedeva una professionalità fin troppo alta, con richiesta di documenti e iscrizioni alla propria community. Ho quindi ripiegato su uno che avevo conosciuto ai tempi ma che allora era chiuso per questione di spazi limitati: HelioHost.

Una volta iscritto e provato ad aprire il cPanel tramite il quale gestire il caricamento e l'utilizzo della mia applicazione, dà una pagina d'errore, senza che io ne riesca a capire il motivo.

Al momento penso tuttavia di aver perso fin troppo tempo, quindi nel pomeriggio proseguirò con il codice in sé, possibilmente testando e sistemando quanto creato la scorsa giornata.

Ho quindi preso in considerazione un'opzione che avrei preferito evitare: inserire la mia carta di credito per il tempo necessario all'installazione dell'applicativo gratuito di Heroku per gestire i db MySQL. Nel corso delle prossime lezioni valuterò se sarà il caso o meno di attuare la cosa.

Ho quindi continuato l'implementazione del codice, strutturando la classe di Subscription (con attributi uguali a quelli della sua tabella, fatta eccezione per l'utente che include l'oggetto e lo stesso l'evento, invece di essere scomposto rispettivamente in 3 e 2 campi).

Il Mapper di Subscription non ha niente di particolare rispetto alla logica di quello User, fatta eccezione per gli attributi indicati sopra (User e Event), i quali tramite le colonne corrispondenti viene creato l'oggetto, senza tuttavia controllarne l'effettiva presenza nel database. Questo è stato fatto dopo aver testato l'effettivo controllo e aver capito che come controllo non solo era chiaramente ridondante, ma anche inutile e dispendioso, dovendo importare i rispettivi Dao all'interno di un semplice Mapper.

Per quanto riguarda il Dao, ho deciso di non crearne uno nuovo ma di inserire i metodi all'interno dello UserDao. La scelta deriva dal fatto che molti metodi dell'utente sono necessari alla struttura dell'iscrizione e ripeterli non avrebbe avuto molto senso.

Ho quindi iniziato a creare la pagina per la visualizzazione dei dati.

L'utente viene visualizzato tramite la strutturazione dell'url:

localhost:8080/{action}/{link}

Il link rappresenta l'utente sul quale si vuole eseguire l'azione, link che è chiaramente univoco e che, a differenza di quanto detto in precedenza, non verrà cancellato una volta usato per la registrazione ma verrà tenuto come valore unico dal quale ricavare gli utenti.

Action si divide in più categorie: modify è un'azione che sarà permessa solo all'utente rappresentato nel link. View è un'azione permessa solamente a consiglieri, assistenti e chiaramente amministratori, anche se questi hanno anche l'azione action, che consiste in una versione migliorata della sola lettura, potendo accettare i parametri dell'utente nel caso questi siano ancora non confermati da nessun altro amministratore.

Sulla base di queste possibilità, ho deciso di strutturare la pagina tramite due semplici variabili che rendono completamente diversa la funzione della pagina: write e checked.

```
model addObject( attributeName: "write", attributeValue: false);
model addObject( attributeName: "checked", attributeValue: true);
```

Tramite questi due semplici attributi passati con il controller alla pagina tramite Spring, indico quali funzioni deve seguire. Nel caso write sia vera, tutti i campi input sono scrivibili. In caso contrario, non lo saranno, tramite l'impostazione di una variabile scritta in ogni campo input come "disabled".

Circa allo stesso modo funziona checked. Nel caso sia vera, verranno mostrati dei campi input checkbox per permettere ad un amministratore di accettare o meno i dati inseriti. Nel caso sia false, la pagina verrà mostrata normalmente.

```
<c:set var="admin" value="${'hidden'}"></c:set>
<c:set var="wPermission" value="${'disabled'}"></c:set>
<c:choose>
    <c:when test="${checked}">
        <c:set var="admin" value="${''}"></c:set>
    </c:when>
    <c:when test="${write}">
        <c:set var="wPermission" value="${''}"></c:set>
    </c:when>
</c:choose>
```

```
<input class="user" ${wPermission} type="text" name="cd-name" value="${user.name}" id="cd-name">
```

```
<div ${admin}>
    <input type="checkbox" id="cd-checkbox-name">
    <label for="cd-checkbox-name">Checked</label>
</div>
```

Ho diviso le due variabili in quanto gli assistenti e i consiglieri avranno la possibilità di lettura ma non di accettare i campi, gli utenti potranno solo scrivere e modificare i propri dati mentre gli amministratori al momento possono solo accettare i parametri, con eventualmente la possibilità in futuro di renderli capaci anche di modificare dati (pensavo più a eventuali errori di battitura e simili, per il momento non chiudo la possibilità ma non la tengo abilitata).

Problemi riscontrati e soluzioni adottate

Ho avuto problemi per tutta la mattinata, potremmo dire.

Avendo ricevuto uno spazio esterno per inserire il mio database, il quale è sempre rimasto in locale, ho cominciato ad apportare le modifiche, le quali erano veramente semplici, a detta mia. Pensavo fosse sufficiente risolvere il collegamento tramite JDBC, in questo modo:

```

13  @Bean
14
15     public DataSource mysqlDataSourceLocalhost() {
16         DriverManagerDataSource dataSource = new DriverManagerDataSource();
17         dataSource.setDriverClassName("com.mysql.jdbc.Driver");
18         dataSource.setUrl("jdbc:mysql://localhost:3306/fsy");
19         dataSource.setUsername("root");
20         dataSource.setPassword("");
21         return dataSource;
22     }
23
24  @Bean
25     public DataSource mysqlDataSource() {
26         DriverManagerDataSource dataSource = new DriverManagerDataSource();
27         dataSource.setDriverClassName("com.mysql.jdbc.Driver");
28         dataSource.setUrl("jdbc:mysql://efof.myd.infomaniak.com:3306/efof_fsy");
29         dataSource.setUsername("efof_fsy_user");
30         dataSource.setPassword("Fsy_Db_Admin");
31         return dataSource;

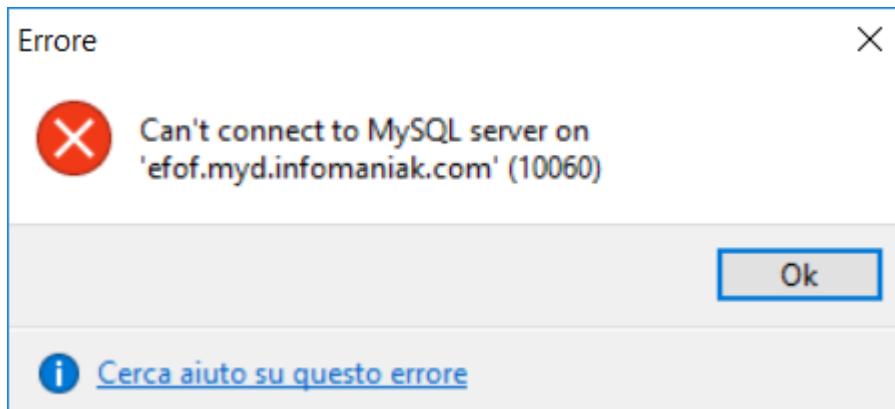
```

La prima versione è quella che utilizzavo fino ad ora, mentre la seconda è la versione corretta per il collegamento. O così credevo. Infatti, collegandomi tramite quel DataSource il risultato è il seguente:

"org.springframework.security.authentication.InternalAuthenticationServiceException: Failed to obtain JDBC Connection; nested exception is

com.mysql.jdbc.exceptions.jdbc4.CommunicationsException: Communications link failure"

Per verificare il funzionamento del collegamento del tipo di connessione tramite quei parametri, ho provato a collegarmi tramite l'utilizzo di HeidiSQL, il quale però mi risultava questo errore:



Ho quindi cominciato a cercare possibili soluzioni al problema, che sembrava collegato e mio. Ho consultato senza successo i seguenti siti:

<https://stackoverflow.com/questions/2983248/com-mysql-jdbc-exceptions-jdbc4-communicationsexception-communications-link-fai>

<https://serverfault.com/questions/89955/unable-to-connect-to-mysql-through-jdbc->

[connector-through-tomcat-or-externally](#)

<https://www.onlinetutorialspoint.com/spring-boot/spring-boot-h2-database-jdbc-example.html>

Ho quindi cominciato a cercare alternative della strutturazione, ma con scarso successo.

<https://stackoverflow.com/questions/50837960/spring-boot-connection-to-mysql-remote-database>

<https://springframework.guru/configuring-spring-boot-for-mysql/>

Alla fine, consci del fatto che il problema potesse non essere di programmazione ma di logica, ho chiesto al mio responsabile consiglio, il quale mi ha spiegato che per utilizzare il web hosting che mi aveva fornito era obbligatorio l'uso di infomaniak. Chiaramente, questo ha cambiato le carte in tavola, rendendo non solo il lavoro svolto nella mattinata inutile, ma l'intera richiesta e conseguente apertura di uno spazio per contenere il database.

Alla fine, ho provato a cercare alternative definitive, ovvero un remote hosting che funzionasse con qualsiasi applicazione esterna. Dopo qualche tentativo tuttavia che ritornava gli stessi errori che dava infomaniak, ho capito che non ne valeva la pena.

<https://remotemysql.com/>

Ho avuto anche un piccolo problema in seguito con la programmazione riguardante il metodo split delle stringhe. Era necessario per dividere l'id dell'utente e utilizzare i 3 parametri in modo separato per inserirli nell'iscrizione, ma purtroppo non funzionava a dovere.

Questo è quello che controllavo:

```
String[] id = rs.getString( columnLabel: "userId" ).split( regex: "." );
System.err.println(rs.getString( columnLabel: "userId")+"："+id[0]："+"/"+id[1]："+"/"+id[2]);
```

Questo è quello che ritornava:

```
e.f.2019-02-20://f.2019-02-20
```

Dopo qualche test e controllo, ho capito che il problema riguardava la regular expression per cui dividevo. Facendo qualche ricerca ho confermato la mia ipotesi, vedendo che per quanto riguarda il carattere '.' Va trattato come carattere speciale, e quindi scritto '\\.'. (sito con soluzione: <https://stackoverflow.com/questions/7935858/the-split-method-in-java-does-not-work-on-a-dot>)

Punto della situazione rispetto alla pianificazione

Ho perso veramente molto tempo oggi, riuscendo a recuperare solamente in parte nel pomeriggio. Tuttavia, avendo ancora due settimane di margine a fine progetto, e consci del fatto che prima o poi dovrei recuperare le ore perse per malattia in qualche modo, sono fiducioso nel farcela. Per le prossime lezioni dovrei implementare la funzione multilingua, la quale ho già ben in chiaro la struttura e penso di risolverla velocemente.

Programma di massima per la prossima giornata di lavoro

Continuare la pagina dell'account, implementando la modifica dei campi e l'accettazione di un admin di questi ultimi già inseriti.

Diario di lavoro

Luogo	Trevano
Data	13.02.2019

Lavori svolti

Oggi ho continuato la fase dell'iscrizione, portando avanti principalmente la pagina account della quale ho spiegato la scorsa volta i vari utilizzi.

Al momento l'unica pagina implementata correttamente è quella dei dati personali (città e simili). Gli amministratori vedono una schermata del genere (soggetta ancora a cambiamenti di grafica):

Come si può vedere, sono presenti i campi principali con accanto un campo checkbox, che rimane attivato di default. Se l'amministratore vuole segnalare il campo come inappropriato, è sufficiente deselezionare il checkbox corrispondente. Ho poi pensato di aggiungere delle note per ogni "parte" (quelle selezionabili tramite il menu in basso), in modo da indicare eventuali (non sono obbligatorie, chiaramente) consigli per sistemare i campi non idonei, questo per favorire la velocità del procedimento. Tutti i campi sono poi inviati tramite submit apposito, ed essendo ora sotto admin la pagina e operazione sarà disponibile solamente per questi ultimi.

Nel caso degli altri utenti, sia i campi checkbox che quello delle note spariscono senza lasciare traccia, in modo da non confonderne il fruitore. C'è comunque la differenza che la modalità di scrittura è disponibile solamente se il link dell'utente corrisponde a quello loggato al momento.

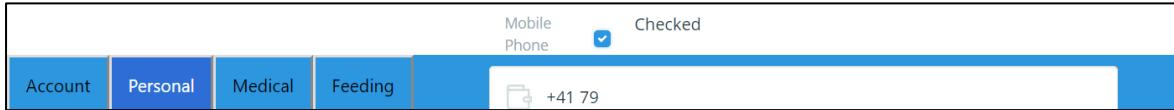
Al momento non è stato gestito il filtro per assistenti e consiglieri: la pagina in modalità view è disponibile a chiunque sia loggato, questo per il fatto di non avere ancora deciso esattamente come strutturare la raggruppamento di questi ultimi.

Ho unificato la richiesta view e quella modify nello stesso metodo, differenziandoli tramite l'ennesimo parametro nell'url che controlla quale delle due è richiesta.

La logica per ricevere tutti i dati ed estrapolare i dati necessari per usarli nel modo corretto non è ancora stata terminata e pertanto la completerò la prossima volta.

Problemi riscontrati e soluzioni adottate

Non ho avuto particolari problemi, se non nella gestione del css della pagina.
Il footer è infatti trasparente, mostrando gli elementi della pagina al di sotto.



Per quanto abbia provato a eliminare tramite la funzione “ispeziona” del browser Google Chrome tutti gli attributi, sia del footer che del corpo principale, non sono riuscito a trovare un elemento che facesse una cosa del genere.

Ho quindi isolato il codice dal resto del template, per rendermi effettivamente conto che era un conflitto tra questo e il codice del footer.

Ho provato a cercare velocemente su internet che cosa potesse creare questo effetto (che nel mio caso è non voluto), ma non sono riuscito a trovare niente se non opacity, index-z e transparent, tutti elementi non presenti nel mio file css.

In mancanza di idee e già in ritardo, ho scelto di ignorare questo difetto grafico e rimandarlo a quando avrò recuperato il tempo perso.

Punto della situazione rispetto alla pianificazione

Sono ancora indietro, ma sto cercando di guadagnare tempo dove riesco, unificando fin da subito il codice per renderlo più performante e più semplice da sistemare.

Programma di massima per la prossima giornata di lavoro

Terminare la pagina di iscrizione, almeno per quanto concerne l’uso dei valori checked e note degli amministratori (gestiti nel modo corretto, con email e simili).

Diario di lavoro

Luogo	Trevano
Data	18.02.2019

Lavori svolti

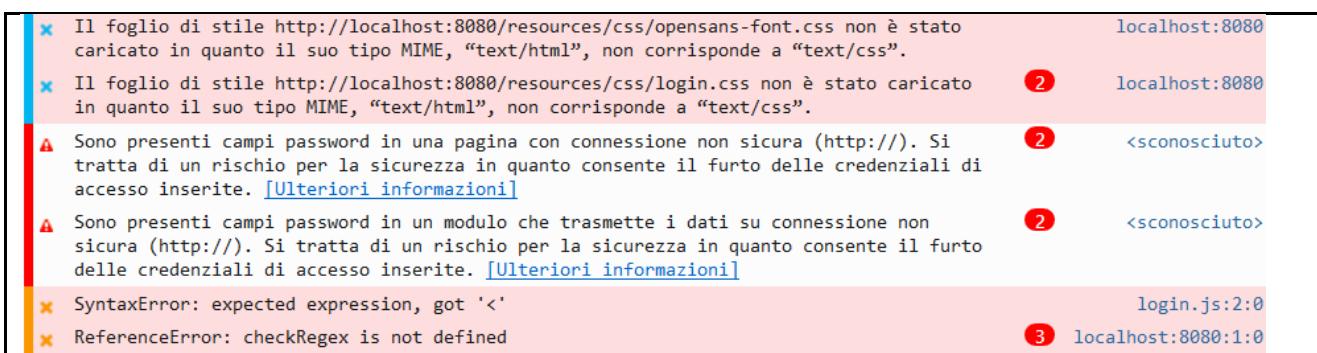
La giornata di oggi purtroppo è stata praticamente inutile. Sulle 4 ore a disposizione, due ci sono state occupate con l'interessante presentazione della scuola BFH, l'HES di Bienna. Mentre nelle prime due ore ho riscontrato un errore mai visto prima, che mi ha impedito di procedere nel lavoro e del quale ancora non ho trovato una soluzione. Per evitare di perdere ulteriore tempo prezioso, vedrò di risolvere questo problema prima della prossima giornata lavorativa, ovvero settimana prossima, visto che i prossimi due giorni sarò assente per il reclutamento di militare. Tengo però a precisare di come questi due giorni non influiscano sulla velocità del lavoro, in quanto ero a conoscenza della cosa fin da prima dell'inizio del progetto e ne ho tenuto da conto nel Gantt preventivo.

Problemi riscontrati e soluzioni adottate

All'avvio del programma ho riscontrato un problema mai visto prima. Alcuni dei miei file css e js (contenuti all'interno delle risorse dell'app, e quindi in locale) hanno improvvisamente cominciato ad avere un contenuto diverso da quello che dovrebbero avere, ovvero quello di altre pagine, più precisamente linee html di una qualsiasi pagina. I due css qui segnalati all'avvio della pagina di login se aperti tramite browser ritornano la pagina base con solamente bootstrap caricato dell'account. Login.js invece, contiene direttamente tutta la pagina di login.jsp, che ovviamente non riesce a interpretare come codice. Ora, il problema maggiore consiste proprio nel fatto che in questi file, che vengono segnalati in questo modo dal browser, in realtà ci sia il normale contenuto, ovvero quanto creato nel corso di questo progetto. Ho provato quindi a rimuovere tutti i file dalla cartella resources, cancellare cache e cookie e riavviare il computer. Non solo non sono stati trovati problemi evidenti, ma hanno reso la situazione più strana del normale. Se l'icona della pagina e un file css non vengono trovati dando un errore 404 (volutamente), lo stesso non si può dire dei 3 file problematici, che vengono comunque visti dove non presenti e interpretati nel modo sbagliato.

- ✖ GET <http://localhost:8080/resources/fonts/line-awesome/css/line-awesome.min.css> login:11
net::ERR_ABORTED 404
- ⚠ Resource interpreted as Stylesheet but transferred with MIME type text/html: "[ht login:10 tp://localhost:8080/resources/css/opensans-font.css](http://localhost:8080/resources/css/opensans-font.css)".
- ⚠ Resource interpreted as Stylesheet but transferred with MIME type text/html: "[ht login:16 tp://localhost:8080/resources/css/login.css](http://localhost:8080/resources/css/login.css)".
- ✖ Uncaught SyntaxError: Unexpected token < login.js:2
- ✖ GET http://localhost:8080/resources/images/icons/fsy_.png 404 fsy_.png:1

Ho quindi provato a cambiare browser, passando a Mozilla. Anche questo tuttavia ha prodotto i medesimi risultati, sempre nella modalità in cui nella cartella resources non è presente alcun tipo di file.



FSY MANAGER Account

localhost:8080/resources/css/opensans-font.css

Account Personal Medical Feeding UPDATE

Account Info

Name

Surname

Birth Date

Email

Genre

- Male
- Female

Non avendo molta scelta, ho deciso di ripiegare su internet, cercando di risolvere dapprima il problema dei file css.

Dopo qualche ricerca su StackOverflow tuttavia (link visitati:

<https://stackoverflow.com/questions/22631158/resource-interpreted-as-stylesheet-but-transferred-with-mime-type-text-html-see>,

<https://stackoverflow.com/questions/49160391/firebase-resource-interpreted-as-stylesheet-but-transferred-with-mime-type-text>,

<https://stackoverflow.com/questions/6089070/resource-interpreted-as-stylesheet-but-transferred-with-mime-type-text-html-in-a>,

<https://stackoverflow.com/questions/12641168/chrome-says-resource-interpreted-as-stylesheet-but-transferred-with-mime-type-t>,

<https://stackoverflow.com/questions/41734976/resource-interpreted-as-stylesheet-but-transferred-with-mime-type-text-javascript/41851906>

), ho capito che il mio problema risiedeva altrove. Infatti non era tanto un problema di cattiva interpretazione del codice, ma proprio di trovare il codice di altri file all'interno del file. Ho quindi cambiato ricerca su internet, con risultati ancora più deludenti di prima. Dopo aver dato un'occhiata ai risultati, anche solo tramite i titoli mi era evidente di come nessuno avesse il problema che ho riscontrato io. E se qualcuno lo ha fatto, non lo ha indicato nel modo in cui verrebbe più naturale secondo me ricercare una soluzione su google (ricerche effettuate: "the content of my file css is the code html of my page problem" e "why my file css have inside the code of html page?").

Il problema non è stato risolto nel tempo a disposizione. Vedrò di arrivare con una soluzione per la

prossima volta.

Sorgente di: <http://localhost:8080/resources/js/login.js> - Mozilla Firefox

```

1 <!doctype html>
2 <html lang="en" class="no-js">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1">
6
7     <link href='http://fonts.googleapis.com/css?family=Open+Sans:400,300,700' rel='stylesheet' type='text/css'>
8
9
10    <link rel="stylesheet" href="/resources/css/account.css">
11    <link rel="stylesheet" href="/resources/vendor/bootstrap/css/bootstrap.css">
12    <script src="/resources/vendor/bootstrap/js/bootstrap.js"></script>
13    <script src="/resources/js/jquery-3.1.1.js"></script>
14    <title>Account</title>
15  </head>
16  <body>
17
18
19
20
21
22
23
24
25  <div class="navbarFooter">
26    <button id="accountB" onclick="changeFields('A')" class="active">Account</button>
27    <button id="personalB" onclick="changeFields('P')" class="">Personal</button>
28    <button id="medicalB" onclick="changeFields('M')" class="">Medical</button>
29    <button id="feedingB" onclick="changeFields('F')" class="">Feeding</button>
30    <button id="saveB" onclick="save()" class="btn-danger">UPDATE</button>
31  </div>
32  <div class="mainBody">
```

Punto della situazione rispetto alla pianificazione

La situazione oggi non è progredita, avendo a che fare con problemi di cause sconosciute e impegno scolastico. Non potrà avanzare nel corso delle due prossime giornate lavorative, ma

come già indicato quest'ultime erano già state previste nel Gantt preventivo, dato che la lettera della convocazione a militare era arrivata prima dell'inizio di questo progetto.

Programma di massima per la prossima giornata di lavoro

Terminare la pagina di iscrizione, almeno per quanto concerne l'uso dei valori checked e note degli amministratori (gestiti nel modo corretto, con email e simili).

Diario di lavoro

Luogo	Trevano
Data	19.02.2019

Lavori svolti

Assente poiché prima delle due giornate di reclutamento di militare.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

La situazione oggi non è progredita, ma come già indicato questo era già stato previsto nel Gantt preventivo, dato che la lettera della convocazione a militare era arrivata prima dell'inizio di questo progetto.

Programma di massima per la prossima giornata di lavoro

Terminare la pagina di iscrizione, almeno per quanto concerne l'uso dei valori checked e note degli amministratori (gestiti nel modo corretto, con email e simili).

Diario di lavoro

Luogo	Trevano
Data	20.02.2019

Lavori svolti

Assente poiché seconda delle due giornate di reclutamento di militare.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

La situazione oggi non è progredita, ma come già indicato questo era già stato previsto nel Gantt preventivo, dato che la lettera della convocazione a militare era arrivata prima dell'inizio di questo progetto.

Programma di massima per la prossima giornata di lavoro

Terminare la pagina di iscrizione, almeno per quanto concerne l'uso dei valori checked e note degli amministratori (gestiti nel modo corretto, con email e simili).

Diario di lavoro

Luogo	Trevano
Data	25.02.2019

Lavori svolti

Oggi ho trovato una soluzione al problema dei file interpretati nel modo sbagliato, terminato il salvataggio dei dati personali e infine cominciato a capire il funzionamento del database datomi a disposizione tramite Heroku.

Il salvataggio dei dati personali è effettuato nella stessa struttura usata per l'iscrizione, dove i parametri vengono passati tramite richiesta POST.

Ho poi fatto in modo di estrarre l'utente loggato in modo da poter verificare il suo link. Infatti, solamente l'utente in questione potrà modificare i propri dati, e tramite la stessa estrazione sarà possibile distinguere tra i vari utenti e gruppi.

Riguardo l'hosting, come già accennato, ho finito per dover inserire la mia personale carta di credito per poter utilizzare la funzione, nonostante questa sia gratuita. Ho tuttavia avuto dei problemi, che mi hanno impedito di riuscire a implementare tutto su hosting esterno come previsto. Finirò quindi questo la prossima volta, oppure continuerò a lavorare in locale per evitare di perdere tempo prezioso.

Problemi riscontrati e soluzioni adottate

Il problema riscontrato la scorsa volta non ha visto ancora una vera e propria comprensione.

Ho provato a creare altri file, a rinominarli e ad inserirli direttamente all'interno della pagina jsp.

Solo quest'ultimo ha dato risultati, ma chiaramente non è delle opzioni migliori. La soluzione finale, che ancora devo spiegarmi perché dovrebbe funzionare, è stata spostare i file css e js direttamente nella cartella resources, invece che nella sottocartella dedicata loro (css e js, che sono cartelle presenti in resources). Questo sacrifica un po' dal lato ordine, ma penso non sia un problema. Vedrò poi lavorando su hosting esterno se tornando alla normalità sorgeranno ancora questo tipo di problemi.

Ho avuto particolari problemi nel connettermi al database esterno di Heroku tramite ClearDB.

Ho consultato molti siti per capire esattamente che fare, ma quello che mi è stato di maggiore aiuto è questo: <https://selimsalihovic.github.io/2016-02-07-using-mysql-on-heroku/>

Ho provato a connettermi al sito tramite l'url e i parametri definiti all'interno del mio account, ma per ora senza successo. Una possibile soluzione che mi è venuta in mente è che il firewall della scuola mi blocca. Proverò dunque entro domani questa possibilità. In caso contrario, non so esattamente come agire e penso finirò per rimandare l'hosting esterno di nuovo per non rallentare il progetto ulteriormente.

Punto della situazione rispetto alla pianificazione

Non è stata una giornata particolarmente proficua. Sono indietro di 5 giorni, con ancora 2 settimane a disposizione di cuscinetto a fine progetto.

Programma di massima per la prossima giornata di lavoro

Terminare la struttura su hosting esterno, testare e documentare il tutto.

Diario di lavoro

Luogo	Trevano
Data	26.02.2019

Lavori svolti

Oggi mi son dedicato al rifinire quanto eseguito ieri in modo frettoloso a causa del poco tempo. Ho potuto testare il filtro dei metodi in base al ruolo dell'utente loggato, e vedere se quest'ultimo fosse lo stesso a cui si voleva accedere in modo che potesse modificare i propri dati. Queste due cose erano state abbozzate già ieri e oggi hanno visto uno svolgimento pratico più test.

Il modo per capire se l'utente loggato sia lo stesso della pagina è il seguente:

Ho poi completato e testato il salvataggio dei dati dell'iscrizione. Ho dovuto rivedere un po' il sistema, in modo da poter prevedere l'utilizzo futuro dei campi input aggiuntivi, uno dei prossimi passaggi necessari al progetto. In generale oggi ho strutturato tutto in modo da renderne l'implementazione più rapida, una volta che dovrò aggiungere la funzione.

Per la pagina JSP, sarà presente un campo extra dove verranno inseriti tutti i campi, con i rispettivi nomi assegnati loro. Per la classe, dopo qualche idea venuta in mente che non è andata in porto (come il creare oggetti appositi, o una lista di stringhe), ho optato per qualcosa che non usavo spesso ma che sicuramente si rivela utile: una mappa di stringhe e stringhe (`Map<String, String>`). In questo modo, posso assegnare il nome dell'input e il valore in maniera semplice e veloce, così come sarà semplice estrarre i valori e assegnarglieli. Per rendermi più agevole il lavoro, ho già cominciato a lavorare con mappe per l'estrazione dei parametri, che vengono richiesti all'oggetto che li contiene della richiesta POST (`HttpServletRequest`) tramite una lista di parametri, estrapolata a sua volta dal database. In questo modo, è possibile aggiungere nuovi valori che venendo inseriti nel database saranno automaticamente richiesti al momento del salvataggio dei dati dell'iscrizione.

Nel salvataggio dei dati, viene controllato dapprima che l'utente esista (e che venga modificato secondo le specifiche nuove dell'utente, anche se ci si augura che queste non varino da un giorno all'altro).

A questo punto, è possibile inserire/aggiornare i propri dati, a dipendenza del fatto che ci si stia iscrivendo ad un evento o semplicemente cambiando qualche dato. Chiaramente, in entrambi i casi gli admin verranno avvisati tramite email di osservare i dati per confermarli. Al momento non è prevista una funzione per avvisare gli admin di quali specifici campi siano stati aggiornati, ma è sicuramente qualcosa che terrò da conto e svilupperò prima della fine del progetto.

Problemi riscontrati e soluzioni adottate

Nessun problema particolare.

Ho usato del tempo nel vedere come strutturare il tutto in base a funzioni future, ma voglio pensare che sia stato più un investimento che una perdita.

Punto della situazione rispetto alla pianificazione

Considerando che nella giornata di domani dovrei ultimare il procedimento dell'iscrizione definitivamente (ma con meno campi, per ora rimarranno pochi e verranno inseriti tutti verso la fine del progetto per praticità), sono indietro esattamente di 5 giorni. Considerando i 2 di assenza, non è niente di irrecuperabile, specialmente contando il possibile lavoro a casa durante le vacanze e le due settimane cuscinetto.

Programma di massima per la prossima giornata di lavoro

Terminare iscrizione, iniziare funzione multilingua.

Diario di lavoro

Luogo	Trevano
Data	27.02.2019

Lavori svolti

Oggi mi son dedicato al completamento della funzione di iscrizione. Si è trattato quindi di testare più e più volte la funzione di salvataggio, in modo da poter utilizzare i print (out e err a dipendenza del caso) di Java per effettuare un vero e proprio debugging approfondito, per scovare le falte o errori minori di logica.

Ho trovato più errori del previsto, che hanno fatto sì che la cosa mi prendesse tutte le 4 ore.

In ogni caso, posso dire di aver completato con successo il salvataggio dei dati delle iscrizioni, aggiornando i dati nel caso sia già presente oppure creandone una nuova.

L'unica grande modifica è stata fatta nell'url. Infatti, ora i parametri richiesti sono anche nome e anno dell'evento. In questo modo, si può distinguere le pagine tra i vari eventi diversi. Per le pagine di eventi ormai conclusi chiaramente verrà inserito il divieto di modificare le informazioni, in modo da evitare cambi non desiderati.

Inoltre, ho reso più leggero l'url, dando la possibilità all'utente di scrivere meno parti per poi ridirizionarlo con le opzioni di default. Può scrivere solo il link dell'utente o i permessi (admin, view, modify) e il link. I restanti parametri verranno dati di default (per i permessi è modify, per gli eventi corrisponde all'ultimo a livello temporale, action a personal).

Problemi riscontrati e soluzioni adottate

A causa di moltissimi problemi minori, non ho potuto fare un diario ad hoc completo nel dettaglio dei procedimenti.

Punto della situazione rispetto alla pianificazione

In linea con quanto pensato la scorsa giornata.

Programma di massima per la prossima giornata di lavoro

Iniziare e possibilmente arrivare vicino alla conclusione della scelta delle lingue.

Diario di lavoro

Luogo	Trevano
Data	18.03.2019

Lavori svolti

Oggi mi son dedicato al riprendere il progetto da dove lasciato.

Dopo aver ricontrallato brevemente il codice, i diari precedenti e che le funzioni implementate (login, email, iscrizioni) funzionassero come di dovere, ho iniziato come da programma l'implementazione delle lingue.

Prima di tutto ho dovuto implementare la funzionalità nelle pagine già esistenti (al momento sto testando quella di login), dove ogni scritta deve avere più lingue disponibili. Per fare ciò è stato affidato loro l'id della pagina e della posizione in quest'ultima, con la lingua decisa e il testo nella suddetta lingua.

In questo modo tramite i Dao è per me semplicissimo estrapolare semplicemente i testi desiderati per la pagina e lingua selezionata dall'utente.

Dopo aver verificato con qualche dato inserito hard coded in italiano e inglese, ho potuto verificarne la funzionalità. A questo punto, il punto maggiore da definire è la pagina dal quale gli amministratori potranno aggiungere lingue e definirne le traduzioni.

Dopo qualche momento, ho quindi ideato una pagina divisa in 2: a sinistra saranno presenti i numeri dell'ordine dei testi, con il rispettivo testo in inglese (di default, ma modificabile) per far capire lo scopo del testo, mentre a fianco sarà possibile inserirne il valore nella lingua selezionata.

La seconda parte è ancora in fase di lavorazione, in quanto non mi è ancora perfettamente chiaro come potrei effettuarla. Tuttavia considerà in un iframe della pagina selezionata che indicherà tutti i testi (con i rispettivi numeri per ritrovarsi più facilmente) e l'idea sarebbe quella di utilizzarla come sorta di preview per la lingua che si sta andando a definire, con i valori che si aggiornano in tempo reale.

Il problema al momento è che non ho completamente chiaro come fare in modo che la pagina abbia gli stessi valori, poiché questi vengono caricati dal database sulla base della lingua.

Per fare ciò quindi, potrei essere costretto o a salvare i dati temporanei sul database, andando ad aumentare leggermente il carico su di esso, oppure a passarli in qualche modo al controller permettendogli di visualizzarli anche nella pagina dell'iframe. Quest'ultima è sicuramente la soluzione migliore, ma l'esperienza mi ha insegnato che lavorare con una cache nel controller è più controproducente che altro, quindi o trovo un altro modo o penso eviterò quest'ultima opzione.

Problemi riscontrati e soluzioni adottate

L'unico problema degno di nota consiste nell'utilizzo dell'iframe. Questo infatti non caricava la pagina, dando un errore di X-Frame-Option che aveva come valore deny. Dopo qualche ricerca, ho scoperto a che serve e che esistono altre 2 possibilità per l'opzione, ovvero sameorigin e allowfrom. SameOrigin faceva al caso mio, dovendo richiamare una pagina del mio stesso dominio, ma cercando ho faticato a trovare il modo di impostare questa opzione.

Dopo aver cercato nello specifico per Spring Boot tuttavia la strada si è apparentemente semplificata (<https://docs.spring.io/autorepo/docs/spring-security/4.0.0.RELEASE/reference/html/headers.html>), nonostante quanto mi dicessero di fare (usare un metodo sameOrigin()) fosse impossibile, visto che non sembra esistere (<https://docs.spring.io/spring-security/site/docs/4.2.11.RELEASE/apidocs/org/springframework/security/config/annotation/web/configurers/HeadersConfigurer.html>). Ho dovuto quindi disabilitare tutta la funzione dei filtri per fare in modo funzionasse, poiché anche ad aggiungere l'opzione sameorigin (<https://stackoverflow.com/questions/26220083/h2-database-console-spring-boot-load-denied-by-x-frame-options>) come indicato qui semplicemente teneva entrambi i valori, e quindi il deny bloccava comunque tutto.

Punto della situazione rispetto alla pianificazione

In linea a quanto previsto.

Programma di massima per la prossima giornata di lavoro

Terminare la funzione del linguaggio, iniziare a definire per bene le email dei vari passaggi.

Diario di lavoro

Luogo	Trevano
Data	20.03.2019

Lavori svolti

La prima ora è stata impiegata a definire meglio alcune specifiche (come abstract, gantt e altre domande sorte tra i colleghi) dell'ormai prossimo esame LPI tramite il docente Fabrizio Valsangiacomo.

Ho poi continuato a lavorare sulla parte delle multiple lingue, arrivando ad un livello soddisfacente.

Non ho ancora terminato, ma ho sistemato l'iframe (che ora carica correttamente la pagina, la scorsa volta aveva problemi di css, risolti tramite la sistemazione di quest'ultimo nella pagina che dava problemi, utilizzando content-fluid sui div per fargli acquistare la dimensione corretta). Inoltre, ho deciso come visualizzare la preview, e ho iniziato a creare il codice per far sì che funzioni.

La funzione di preview verrà caricata nel momento in cui si premerà un bottone (questo per non appesantire eccessivamente la pagina, dato che il responsive automatico era possibile ma decisamente pesante di implementare), che andrà a salvare i valori inseriti nel database, con tuttavia una piccola differenza: è stato aggiunto un parametro (di default null) nel quale è possibile inserire il link identificativo dell'utente (in questo caso, dell'admin, poiché questa funzione è disponibile solamente a loro). In questo modo, anche interrompendo e ricaricando la pagina, avranno a disposizione le loro specifiche traduzioni, anche se ancora non implementate nel sito definitivo, che invece caricherà quelle senza utente definito.

Inoltre, ognuno potrà portare avanti traduzioni differenti. Questo per permettere agli amministratori magari di dividere le varie traduzioni, senza ostacolare i propri processi a vicenda magari sovrascrivendo dati e quindi ore di lavoro di un'altra persona.

Il sistema con cui sto salvando i dati nel database è il medesimo degli utenti, e provvederò a documentarlo non appena terminato.

Lo stesso avverrà per la pagina, al momento piena di placeholder e debugging.

Problemi riscontrati e soluzioni adottate

Nessun problema particolare.

Punto della situazione rispetto alla pianificazione

Le due settimane di cuscinetto sono ora tirate. Conto di finire la funzione multilingua entro le prossime due giornate lavorative, in modo da farmi rimanere ancora l'aggiunta/scaricamento dei file PDF, l'aggiunta di attributi non definiti come opzione per gli admin e una ridefinizione

generale delle email, operative ma non implementate al meglio in ogni loro passaggio.
Ho tre settimane per terminare tutto questo e portare avanti la documentazione, rimasta ormai
abbastanza indietro dato che ho preferito cercare di recuperare il tempo perso
nell'implementazione causa malattia e problemi riscontrati.

Programma di massima per la prossima giornata di lavoro

Terminare la funzione della multilingua, salvandone i dati e usandola in ogni pagina (al momento
sta venendo testata solamente sul login, nonostante sia implementata la possibilità per altre
pagine).

Diario di lavoro

Luogo	Trevano
Data	25.03.2019

Lavori svolti

Oggi mi son focalizzato sulla funzione del multilingua, riscontrando qualche problema fin da subito.

Infatti il metodo POST sembrava non funzionare, almeno sulla pagina dei linguaggi. Dopo un po' tuttavia mi son reso conto di come il browser indicasse che la richiesta non era stata accettata, il server reagiva comunque, facendo quello che doveva.

Ho sistemato di molto la pagina, che ora è operativa per quanto riguarda la modifica dei dati in generale. Ho inoltre inserito la funzione dei valori di default. Se una lingua non è ancora stata creata, i valori saranno vuoti, ma nel caso esista già una versione definitiva della traduzione (ovvero quella generale nonché implementata per i siti) verrà caricata quella, in modo da permettere la modifica piuttosto che l'inserimento da zero.

Per fare ciò, il programma sfrutta una differenza tra le richieste update e insert nel codice, che documenterò non appena terminata la funzione.

La funzione del salvataggio della "traduzione personale degli amministratori" ancora non funziona, e non sono ancora riuscito a trovare il problema nonostante abbia effettuato molto debugging.

Problemi riscontrati e soluzioni adottate

Nessun problema particolare, solo tanti piccole imperfezioni nel codice che mi hanno rubato tempo prezioso.

Punto della situazione rispetto alla pianificazione

In pari con quanto previsto nella scorsa settimana.

Programma di massima per la prossima giornata di lavoro

Terminare definitivamente l'inserimento della lingua, creare, scaricare e ricaricare file PDF.

Diario di lavoro

Luogo	Trevano
Data	26.03.2019

Lavori svolti

Oggi son riuscito finalmente a terminare l'implementazione delle lingue.
Il risultato finale è questo:

The screenshot shows a web interface for managing translations. On the left, a table lists 20 entries, each with a number (0-19) and two columns: the original English text and its Italian translation. On the right, a preview window shows a login form with fields for First Name, Last Name, Birth Date, Password, and Remember Me, along with links for forgot password and recover password.

0	Invalid Format
1	Login
2	First Name
3	Last Name
4	Birth Date
5	Password
6	Remember Me
7	Forgot the password?
8	Recover it here!
9	You don't have an account?
10	Register
11	REGISTER FORM
12	First Name
13	Last Name
14	Birth Date
15	Email
16	Password
17	Confirm Password
18	p7
19	p8
20	p9

Come già spiegato, la prima linea indica il numero (tramite il quale è possibile rintracciarne la posizione all'interno della pagina a lato), la seconda colonna il contenuto nella lingua più utilizzata (ovvero l'inglese, seppur sia modificabile se richiesto) e la traduzione dell'utente, la quale avrà i valori di default (se disponibili, quelli della lingua utilizzata, altrimenti quelli in inglese).

Il pulsante "see Preview" salva i dati a livello dell'utente loggato (che deve essere per forza amministratore, comunque), mentre "SAVE DATA" pubblica la traduzione rendendola disponibile sulla pagina. Le pagine ora sono impostate di default a inglese, ma con un ultimo "{lingua scelta}" è possibile caricare le versioni tradotte.

Questa è (al momento) ciò che si vede con la traduzione italiana (incompleta):

The screenshot shows a 'Log in' page with the following fields:

- Nome (Name)
- Cognome (Surname)
- Data di Nascita (Birthdate)
- Password
- Ricordami (Remember me)

Below the form:

- Forgot the password? [Recover it here!](#)
- [Login](#)
- You don't have an account?
- [Register](#)

Mancano ancora i controlli sull'utente (al momento è stato messo un link di un admin e viene considerato come se lui fosse sempre loggato) e la verifica che la lingua richiesta esista, ma sono cose molto rapide da implementare.

Il codice utilizzato non è ancora stato testato in ogni sua versione, ma lo definirei funzionante. Tutta la funzione dovrà comunque superare tutta una fase di test approfonditi, ma al momento non sono prioritari visto il poco tempo che mi rimane.

```

43  @RequestMapping(value = "/language/{language}/{page}", method = RequestMethod.GET)
44  public ModelAndView updateLanguage(@PathVariable String page, @PathVariable String language, HttpServletRequest request) {
45      ModelAndView model = new ModelAndView();
46      try{
47          int idPage = Integer.parseInt(page);
48          Page p = siteDao.getPageById(idPage);
49          if(p!=null){
50              List<Content> csEng = siteDao.getContentsPageByIdAndLanguage(idPage, language: "ENG");
51              int tot = csEng.size();
52              for(int i = 0; i < p.getContentCount() - tot; i++){
53                  System.out.println("add values");
54                  csEng.add(new Content(p.getId(), idText: csEng.size()+i, language: "ENG", text: ""));
55              }
56              List<Content> cs = siteDao.getContentsPageByIdAndLanguage(idPage, language);
57              List<Content> personalCs = siteDao.getContentsForPageAndUser(idPage, language, linkUser: "FaSW3lowMe");
58              if(csEng.isEmpty()){
59                  cs = harmonizeLists(csEng, cs);
60                  personalCs = harmonizeLists(cs, personalCs);
61              }else{
62                  return getError(model, error: "Content does not exist", pageError: true);
63              }
64              model.addObject( attributeName: "link", attributeValue: "FaSW3lowMe" );
65              model.addObject( attributeName: "page", p );
66              model.addObject( attributeName: "contents", personalCs );
67              model.addObject( attributeName: "contentsPrim", siteDao.getContentsPageByIdAndLanguage(idPage, language: "ENG") );
68              model.addObject( attributeName: "preview", attributeValue: request.getRequestURL().toString() + "/preview" );
69          }
70      }catch (NumberFormatException ex){
71          System.out.println("Number not valid");
72      }
73      model.setViewName("languages");
74  }

```

Metodo principale della visualizzazione della pagina, si occupa di estrarre i contenuti dal database tramite dei semplici select creati nel Dao. I problemi maggiori che poteva dare erano di IndexOutOfBoundsException, ma invece che gestirli con un try ho preferito impedire che tali errori si potessero verificare, aggiungendo dei valori vuoti come valori di default dove

necessari.

Inoltre, viene utilizzata il metodo harmonizeLists. Questo ha la funzione di usare una lista come base per modificare la seconda:

```

113     private List<Content> harmonizeLists(List<Content> originals, List<Content> variants) {
114         List<Content> definitives = new ArrayList<Content>();
115         definitives.addAll(originals);
116         for(int i = 0; i < originals.size(); i++) {
117             Content original = originals.get(i);
118             if(variants.size() > i){
119                 Content variant = variants.get(i);
120                 if(variant.getIdPage()==original.getIdPage() && variant.getIdText()==original.getIdText() && !variant.getText().isEmpty()){
121                     definitives.get(i).setText(variant.getText());
122                     definitives.get(i).setLanguage(variant.getLanguage());
123                 }else{
124                     if(variant.getIdText()<= originals.size()){
125                         definitives.get(variant.getIdText()).setLanguage(variant.getLanguage());
126                         definitives.get(variant.getIdText()).setText(variant.getText());
127                     }
128                 }
129             }else{
130                 break;
131             }
132         }
133         return definitives;
134     }

```

In questo modo, la nuova lista sarà della dimensione corretta e avrà i valori di default dove necessari e quelli corretti dove presenti.

Il metodo che viene richiamato per essere visualizzato nell'iframe fa qualcosa di molto simile:

```

77     @RequestMapping(value = "/language/{language}/{page}/preview", method = RequestMethod.GET)
78     public ModelAndView getPreview(
79         @PathVariable String page, @PathVariable String language, HttpServletRequest request) {
80         ModelAndView model = new ModelAndView();
81         try{
82             int idPage = Integer.parseInt(page);
83             Page p = siteDao.getPageById(idPage);
84             if(p!=null){
85                 model.setViewName(p.getTitle());
86                 String link = "";
87                 if(request.isUserInRole(ROLE_ADMIN)){
88                     User u = getLoggedUser();
89                     if(u!=null){
90                         link = u.getLink();
91                         //controlli futuri
92                     }
93                 }
94                 link = "FaSW3lowMe";
95                 List<Content> cs = siteDao.getContentsForPageAndUser(idPage, language, link);
96                 int dim = cs.size();
97                 for(int i = 0; i < p.getContentCount() - dim; i++){
98                     System.out.println("add values");
99                     cs.add(new Content(p.getId(), idText: dim+i, language, text: ""));
100                }
101                p.setContents(cs);
102                model.addObject(attributeName: "preview", attributeValue: true);
103                model.addObject(attributeName: "page", p);
104            }
105        }catch (NumberFormatException ex){
106            System.out.println("Number not valid");
107        }
108        return model;
109    }

```

Tuttavia, questo ha bisogno di caricare solamente i contenuti dell'amministratore loggato (ancora hardcoded per velocizzare il processo).

Il punto focale di tutto è stato sicuramente il sistema di salvataggio dati.

Avviene tutto tramite una richiesta post fatta Ajax nella pagina languages.jsp:

```

1  function updateDB(saved) {
2      var urlS = window.location.href + "/" + saved;
3      console.log(urlS);
4      $.ajax( url: {
5          type: "POST",
6          url: urlS,
7          data: $("#dataForm").serialize(),
8          error: function () {
9              if (saved) {
10                  console.log(self.location);
11                  location = self.location;
12              } else {
13                  document.getElementById( elementid: 'previewFrame').contentDocument.location.reload( forcedReload: true);
14              }
15          }
16      });
17
18      console.log(saved);
19 }

```

Questo utilizza il link (e quindi i parametri) della pagina corrente per poi aggiungerci un parametro, ovvero un booleano (false indica il salvataggio solo per l'admin loggato, true per tutti). Nel primo caso, necessita di essere ricaricato solo l'iframe per mostrare i risultati. Nel secondo caso va refreshata l'intera pagina, in quanto i valori modificati potrebbero essere quelli di riferimento (ovvero la seconda colonna, quella che al momento è sempre in inglese).

La richiesta viene interpretata grazie al seguente metodo:

```

150     @RequestMapping(value = "/language/{language}/{page}/{db}", method = RequestMethod.POST)
151     public void updateDataLanguage(
152         @PathVariable String page, @PathVariable String language, @PathVariable String db,
153         HttpServletRequest request){
154         try{
155             int idPage = Integer.parseInt(page);
156             Page p = siteDao.getPageById(idPage);
157             if(p!=null) {
158                 boolean save = db.equals("true");
159                 String[] contents = request.getParameterValues( s: "contents");
160                 User u = getLoggedUser();
161                 /*if(u!=null){
162                     System.out.println("POST SAVE");
163                     System.out.println(siteDao.saveLanguage(p, language, contents, save, u.getLink()));
164                 }*/
165                 boolean success = siteDao.saveLanguage(p, language, contents, save, user: "FaSW3lowMe");
166                 if(!success){
167                     System.err.println("The operation for save the traduction has failed. ");
168                 }
169                 List<Content> csEng = siteDao.getContentsPageByIdAndLanguage(idPage, language: "ENG");
170                 List<Content> cs = siteDao.getContentsPageByIdAndLanguage(idPage, language);
171                 if (csEng != null) {
172                     if (cs == null) {
173                         cs = new ArrayList<>();
174                     }
175                 }
176             }
177             }catch (NumberFormatException ex){
178                 System.out.println("POSTERRNUM");
179             }
180             System.out.println("POST END");
181         }

```

Questo richiama a sua volta il metodo presente nel Dao, che si occupa di salvare (aggiungere o modificare) i dati appena inseriti. Al momento manca ancora un controllo su quest'ultimi, sebbene si ritenga che gli amministratori siano consci che non si tratti di un'operazione da prendere alla leggera.

```

96     public boolean saveLanguage(Page p, String lan, String[] contents, boolean definitive, String user){
97         String isUser = "";
98         if(p != null) {
99             if(getPageById(p.getId()) == null){
100                 System.err.println("Page not found");
101                 return false;
102             }
103             String sql;
104             List<Content> cUser = getContentsForPageAndUser(p.getId(), lan, user);
105             for(int i = 0; i < contents.length; i++) {
106                 if(getContent(p.getId(), i, lan, user) == null){
107                     System.out.println("insert");
108                     sql = "insert into content (text, idPage, idText, language, linkUser) values (?, ?, ?, ?, ?)";
109                 }else{
110                     System.out.println("update");
111                     sql = "update content set text=? where idPage=? && idText=? && language=? && linkUser=?";
112                 }
113                 this.doChangeFilter(sql, ...params: contents[i], p.getId()+"", i+"", lan, (definitive?"":user));
114             }
115             return true;
116         }
117         System.err.println("page not passed");
118         return false;
119     }
120 }
```

Se avverranno delle modifiche provvederò a segnalarle. Più spiegazioni (riga per riga o quasi) di tutto il procedimento verranno inserite in futuro nella documentazione nella parte di implementazione.

Problemi riscontrati e soluzioni adottate

Nessun problema particolare, solo tanti piccole imperfezioni nel codice che mi hanno rubato tempo prezioso.

Punto della situazione rispetto alla pianificazione

Avrei dovuto iniziare anche la creazione dei PDF, ma purtroppo la struttura ha richiesto più tempo del previsto per essere ultimata e testata, complici errori di distrazione.

Programma di massima per la prossima giornata di lavoro

Creare, scaricare e ricaricare file PDF.

Diario di lavoro

Luogo	Trevano
Data	27.03.2019

Lavori svolti

Oggi ho cominciato a controllare come utilizzare i pdf.

Come avevo già trovato in precedenza (si parla di inizio progetto fase di analisi), esiste un tutorial funzionale per strutturare i pdf tramite Java, ovvero <https://www.baeldung.com/java-pdf-creation>.

Ho quindi implementato la prima versione, vedendo le varie possibilità con tabelle e immagini, le quali basteranno per tutto quello che mi serve.

Ho quindi iniziato ad analizzare le pagine PDF che andrò a creare, strutturando il tutto in tabelle e celle a brutta copia ma utile a me per capire la struttura da fare.

Ho poi cercato velocemente come importare un file su html, non avendolo mai fatto prima. Ho trovato facilmente che esiste il tipo "file" per gli input.

A questo punto l'unico tassello mancante era il salvataggio nel database e visualizzazione tramite pagina web. Per la prima questione è semplice, ma la seconda potrebbe rivelarsi complicata. Ho trovato questo sito a riguardo <https://aboullaite.me/spring-boot-excel-csv-and-pdf-view-example/>, ma non ho ancora avuto il tempo di testare se funzioni o meno come sistema.

Nel frattempo, mi è stato fatto notare da un collega (Gairo Mauro) di come molti record blob possano pesare sul database.

Ho quindi provato a pensare ad una soluzione alternativa, magari implementando uno storage di Heroku, come <https://elements.heroku.com/addons/xplenty>.

Cercando su Heroku inoltre, ho scoperto che esiste un add-on () che si occupa esattamente di creare file PDF, che addirittura ci riesce tramite semplice codice html, css e javascript.

Dopo averci dato un'occhiata tuttavia, ho notato che probabilmente andrei a complicarmi la vita più che altro, dati i numerosi dati che devo estrapolare dal database per la creazione dei file.

Inoltre, usare nello specifico un add-on di Heroku avrebbe bloccato l'intero applicativo a questo host, impedendo eventuali cambi futuri per ottenere più controllo o performance.

Per finire, ho cominciato ad implementare la pagina che avrà la possibilità di importare i file pdf all'interno degli utenti (solo quest'ultimi hanno il permesso di farlo) e di visualizzarli (disponibile a tutti e 3 i permessi, view-admin-utente stesso), riscontrando tuttavia un problema che non avevo considerato: il fatto che non ho mai lavorato con file blob, e che quindi non so inserirli nel database. Ho cominciato quindi a documentarmi a riguardo, ma il tempo a mia disposizione della giornata è finito prima che potessi trovare la soluzione definitiva.

Problemi riscontrati e soluzioni adottate

Ho avuto dei problemi principalmente riguardo ad elementi non sufficientemente ragionati in corso di analisi, che mi hanno portato a dover strutturare per la prima volta alcuni dettagli riguardanti i pdf.

Punto della situazione rispetto alla pianificazione

Indietro rispetto a quanto sperassi, ma recuperando il tempo perso per malattia in queste settimane dovrei finire tranquillamente.

Programma di massima per la prossima giornata di lavoro

Terminare una volta per tutte la gestione dei PDF.

Diario di lavoro

Luogo	Trevano
Data	01.04.2019

Lavori svolti

Oggi sono stato assente per problemi di salute, e quindi il progetto non è avanzato.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Ho perso altre 4 ore sulla programmazione, ma ho fiducia sul fatto di recuperarle in fretta e che questo non vada ad inficiare sulla qualità del prodotto finale.

Programma di massima per la prossima giornata di lavoro

Terminare una volta per tutte la gestione dei PDF.

Diario di lavoro

Luogo	Trevano
Data	02.04.2019

Lavori svolti

Oggi, dopo aver ripreso sottomano il codice riguardante i PDF, mi son reso conto che a causa del tempo che ormai scarseggia, non ha alcun senso provare a sbattere la testa sul database di Heroku e il salvataggio di dati in qualche add-on sconosciuta.

Per questo motivo, ho intenzione di recuperare, probabilmente a casa visto l'ammontare delle ore perse per problemi di salute, un'idea data dal mio responsabile, che ai tempi non mi pareva chissà cosa da implementare, ma al momento è l'unica via possibile (o quanto meno, che vedo).

L'idea in questione è utilizzare parti di php e parti di spring boot assieme. Più precisamente, Spring Boot si occupa di tutta la gestione dell'applicativo, mandando poi i dati da salvare in formato Json alle pagine php, che si occuperanno solamente di scriverlo all'interno del database.

Recupererò le parti necessarie per fare ciò, così come il salvataggio dei pdf in uno storage apposito, tramite vecchi progetti ed esercizi effettuati nel corso di questi anni.

Capito ormai come procedere, ho deciso di rimandare ancora la questione pdf, per concentrarmi su altro.

Ho infatti dato un'occhiata alla funzione di remember-me, parte che avevo fino ad ora ignorato in quanto ritenevo fosse relativamente semplice.

Internet mi ha dato ragione in fretta, poiché è bastato seguire questa pagina (<https://rokonez.com/spring-framework/spring-security/configure-persistent-token-remember-me-authentication-persistent-token-approach-spring-boot>) per ottenere il risultato sperato.

una delle due parti ancora primarie che mi mancano: l'aggiunta degli attributi (l'altra è la gestione dei file csv).

Le seguenti righe nel file SecurityConfig.java (metodo configure) definiscono tutto quello che serve per la funzione.

```

61           .and() HttpSecurity
62             .rememberMe() RememberMeConfigurer<HttpSecurity>
63               .tokenValiditySeconds(24 * 60 * 60) // expired time = 1 day
64             .tokenRepository(persistentTokenRepository()) RememberMeConfigurer<HttpSecurity>

```

È necessaria comunque una nuova tabella, che registri i token di login dei vari utenti per poi riproporglieli.

```

2 create table persistent_logins (
3   username varchar(64) not null,
4   series varchar(64) primary key,
5   token varchar(64) not null,
6   last_used timestamp not null
7 );

```

Questa funzione devo dire che, sebbene sulla carta sembrasse tra le più ostiche che mi avrebbero preso molto tempo, è proseguita senza troppi problemi. In una giornata posso dire di aver pressoché finito, sebbene sia ancora necessario qualche ritocco e soprattutto gestione di errori e

test.

Problemi riscontrati e soluzioni adottate

Ho avuto qualche problema nel revisionare il codice scritto tempo fa, rivedendo come estrapolare i dati dalla tabella subscription.

Alla fine, dopo qualche ipotesi finita male, ho trovato come ottenere il numero di colonne (<http://www.alessandrolacava.com/blog/how-to-get-the-number-of-columns-in-a-resultset-in-java/>). Tramite esso, ho potuto tranquillamente far ciclare tutti le colonne della tabella, filtrando senza problemi quelli già presenti da quelli no (infatti, quelli che verranno aggiunti dagli admin avranno un “_” davanti al nome, che chiaramente non sarà visibile all'interno della pagina). Questo mi consente di evitare l'aggiunta di un campo inutile (e di una classe rispettiva) gestendo così i parametri come semplici stringhe.

Punto della situazione rispetto alla pianificazione

Indietro, dovrò sicuramente recuperare quanto meno le 20 ore perse per malattia. Con quelle tuttavia, dovrei essere tranquillo.

Programma di massima per la prossima giornata di lavoro

Terminare aggiunta attributi e test. Inizio pagina filtro utenti.

Diario di lavoro

Luogo	Trevano
Data	03.04.2019

Lavori svolti

La prima ora è stata dedicata all'ennesima spiegazione riguardante il LPI, con il chiarimento di alcuni punti e il sottolinearne di altri molto importanti.

Prima di riprendere il progetto, ho notato una mancanza nel diario di ieri riguardante la funzione per ricordare l'utente tramite token contenuto nel cookie dedicato.

Mancava infatti l'immagine della funzione per gestire il token, ovviamente presente all'interno del codice in quanto fondamentale per il corretto funzionamento.

```
78     @Bean
79     public PersistentTokenRepository persistentTokenRepository() {
80         JdbcTokenRepositoryImpl tokenRepository = new JdbcTokenRepositoryImpl();
81         tokenRepository.setDataSource(dataSource);
82         return tokenRepository;
83     }
```

Ho finalmente terminato la funzione dell'aggiunta di attributi. Invece di testarla nella sua interezza tuttavia, ho preferito passare alla fase successiva. Testerò e sistemerò tutto nel momento in cui avrò tutti i pezzi a disposizione e potrò quindi risistemare l'intero sito.

Al momento è presente solamente una funzione di aggiunta per questi attributi extra. Nel caso si voglia in futuro aggiungere le funzionalità di modificare i testi oppure eliminarne, provvederò a inserirle, ma al momento non le ho ritenute così importanti o richieste da dedicarci del tempo.

Metodo di gestione attributi extra:

```
31     List<Map<String, String>> values = new ArrayList<>();
32     int tot = rs.getMetaData().getColumnCount();
33     Map<String, String> m = new HashMap<>();
34     for(int i = 1; i <= tot; i++) {
35         System.out.println(i+": "+rs.getString(i));
36         if(rs.getMetaData().getColumnName(i) != null) {
37             if(rs.getMetaData().getColumnName(i).startsWith("_")){
38                 m.put(rs.getMetaData().getColumnName(i), rs.getString(i));
39                 values.add(m);
40             }
41         }
42     }
43     s.setExtra(values);
```

Questo semplice pezzo di codice è stato inserito nel Mapper della classe Subscription. Permette di ottenere tutti i nomi delle colonne e di estrarre solamente quelli che iniziano con il valore definito la scorsa volta per gli attributi extra, ovvero “_”.

```

144     public boolean addSubscriptionColumn(String content) {
145         String c = "_" + content;
146         String sql2 = "insert into parameter(name) values (?)";
147         this.doChangeFilter(sql2, c);
148         String sql = "ALTER TABLE subscription add " + c + " varchar(250)";
149         return this.doChangeFilter(sql);
150     }

```

Come si può vedere, al momento è previsto uno spazio di 250 caratteri massimo per il campo. Questo è stato deciso arbitrariamente da me, in quanto stimo sia un valore massimo accettabile. Può essere modificato tranquillamente nel caso mi venga detto diversamente.

Come già detto, il valore viene aggiunto sia come attributo alla tabella subscription che alla lista vera e propria degli attributi, che viene utilizzata per estrarre i dati corretti e mostrarli nell'ordine preimpostato, facilitando inoltre molte dei passaggi altrimenti sarebbero troppo macchinosi.

```

45     public ModelAndView getAttribute(@RequestParam(required = false) String error) {
46         ModelAndView model = new ModelAndView();
47         model.setViewName("settings");
48         List<String> ps = userDao.getParameterNames();
49         model.addObject(attributeName: "parameters", ps);
50         System.err.println(error);
51         if(error!=null){
52             model.addObject(attributeName: "error", error);
53         }
54         return model;
55     }
56
57     @RequestMapping(value = "/parameters/{name}", method = RequestMethod.GET)
58     public String getAttribute(@PathVariable String name, RedirectAttributes ra) {
59         if(!userDao.isAParameter(name)){
60             userDao.addSubscriptionColumn(name);
61         }else{
62             System.err.println("uguale");
63             ra.addAttribute(s: "error", o: "the column already exists");
64         }
65         return "redirect:/admin/parameters";
66     }

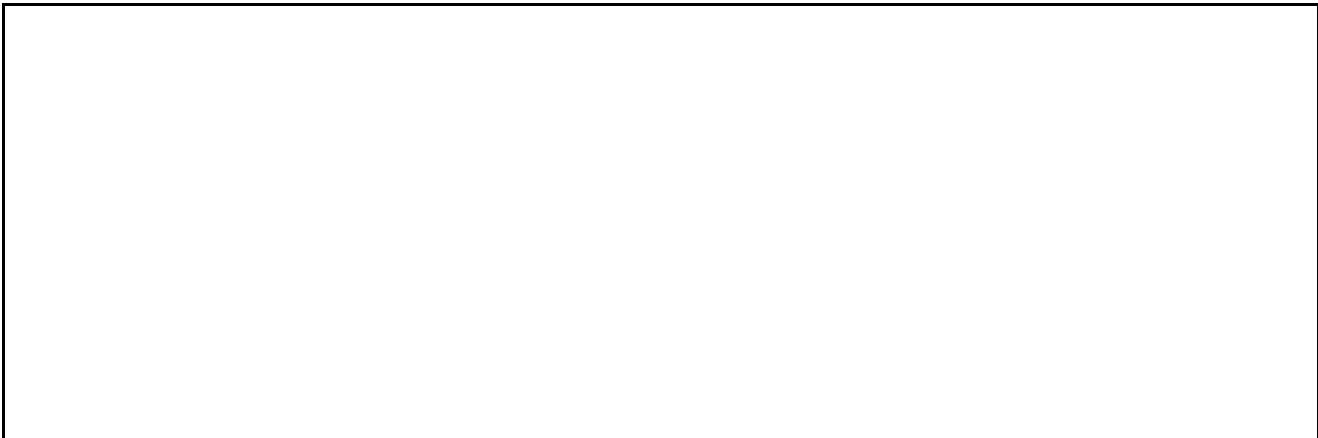
```

La gestione tramite controller va rivista, in quanto al momento il messaggio d'errore non viene visualizzato nel modo che vorrei. Ho tuttavia preferito documentare ora il tutto, in modo da poter passare ad un'altra funzionalità per velocizzare i tempi.

Sono infatti passato all'inserimento di una pagina che presenta tutti gli utenti e iscrizioni, in modo da gestire il tutto più velocemente. In futuro, i file csv verranno creati proprio in base a questa pagina, grazie ai filtri presenti.

La pagina, come indicato molti nel diario del 23 gennaio, è questa <https://codyhouse.co/demo/content-filter/index.html>.

Ho quindi passato il tempo restante a capirne il funzionamento e vedere di sfruttarlo per i miei scopi. Grazie alle classi e all'utilizzo degli attributi personalizzati (data-[parola]) è in grado di effettuare un primo filtro agli elementi. La struttura che ho pensato è dunque differenziare tra utenti e iscrizioni, e tra quest'ultime anche il filtro per evento (nome e anno) e avanzamento dell'iscrizione (per il quale è stata creata una nuova tabella apposita). È ancora in fase abbastanza primitiva, ma penso mi ci vorrà ancora poco per dargli la forma che desidero. Per filtri più specifici come per nome o simili, la pagina non offre nulla. Dovrò quindi pensare da solo al sistema da utilizzare oppure, viste le tempistiche, rinunciarci in favore di una fruizione maggiore del resto delle funzionalità.



Problemi riscontrati e soluzioni adottate

Nessun problema particolare.

Punto della situazione rispetto alla pianificazione

Ho fatto molto più di quanto mi aspettassi oggi, ma chiaramente rimango ancora indietro sul programma. Entro la prossima giornata dovrò sicuramente portarmi avanti per recuperare le ore perse.

Programma di massima per la prossima giornata di lavoro

Terminare filtro utenti, importare file csv ed esportarli.

Diario di lavoro

Luogo	Trevano
Data	08.04.2019

Lavori svolti

Purtroppo questo weekend ho recuperato meno tempo del previsto, riuscendo così a completare solamente la parte riguardante il filtro e il rispettivo salvataggio (degli utenti o iscrizioni) all'interno di un file csv.

Ho inoltre controllato riguardante Heroku, e con mia sorpresa ho scoperto che il mancato collegamento al database era dovuto al proxy e a qualche piccolo cambio di parametri, come il passaggio da DataSource a BasicDataSource (verrà spiegato nella documentazione la differenza tra i due).

Visti i tempi ormai stretti, provvederò a spiegarlo direttamente nella parte dedicata all'interno della documentazione.

Sempre per questioni di tempo, ho deciso di rinunciare alla gestione dei file PDF, sia per quanto riguarda le liste da scaricare sia download e upload da parte dell'utente, andando di fatto a rimuovere un tassello dall'intero passaggio, che è quindi saltato anche a livello di programmazione (sebbene il codice delle email per gli amministratori sia comunque presente).

Problemi riscontrati e soluzioni adottate

Il tempo stringe sempre di più, e questo porta a sacrifici che verranno ormai come mancanze e limitazioni che rimarranno tali.

Punto della situazione rispetto alla pianificazione

Non avendo recuperato nemmeno la metà del tempo perso per malattia, mi ritrovo a dover tirare il massimo in questi ultimi giorni per avere qualcosa di presentabile, specialmente riguardo la documentazione.

Programma di massima per la prossima giornata di lavoro

Ritoccare il codice facendo in modo che tutte le funzioni primarie vengano eseguite nel modo corretto, terminare la documentazione.

Diario di lavoro

Luogo	Trevano
Data	09.04.2019

Lavori svolti

Oggi ho portato avanti praticamente solo la documentazione, arrivando a quasi terminare i test e avendo una buona metà di implementazione fatta. cercando di aggiungere dove possibile alcune funzioni, come la home fino ad ora snobbata. Questa deve avere il minimo indispensabile per permettere all'utente di iscriversi agli eventi e di gestire questi ultimi, tenendo conto dei suoi progressi. Verrà spiegata per quanto possibile nella documentazione.

Problemi riscontrati e soluzioni adottate

Il tempo stringe sempre di più, e questo porta a sacrifici che verranno ormai come mancanze e limitazioni che rimarranno tali.

Punto della situazione rispetto alla pianificazione

Con 4 ore rimanenti, ormai mi è chiaro come il tempo non mi basti. Farò tuttavia il possibile per consegnare una documentazione utilizzabile e un lavoro che, per quanto limitato e ancora grezzo, funzioni correttamente.

Programma di massima per la prossima giornata di lavoro

Terminare la documentazione.

Diario di lavoro

Luogo	Trevano
Data	10.04.2019

Lavori svolti

Oggi ho terminato il progetto, completando la documentazione e il progetto per quanto possibile col tempo che mi rimaneva.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

-

Programma di massima per la prossima giornata di lavoro

-