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