**Iterazione 1- Planning di aggiornamento**  
**Aggiornamento/Espansione degli Use Case**

**UC1: LOG IN / SIGN IN**

Non è necessario l’utilizzo di una email per il login, in quanto non viene utilizzata in alcun modo.

POSTCONDIZIONI: l’utente può eseguire nuovamente l’accesso all’app inserendo

username e password

**UC2 + UC3**

I presenti casi d’uso restano invariati, è stata presa la decisione di non implementarli nella presente versione dell’app; il ruolo di “Ente” verrà ricoperto dal team di sviluppo, che si occuperà di popolare il database dei luoghi.

**UC4 – UC7**

I presenti casi d’uso restano invariati.

**Pianificazione aggiunte software al progetto**

* Costruzione database in locale;
* Creazione server locale, con la struttura Model-Controller su Java;
* Inizializzazione progetto Flutter per la componente View;
* Creazione collegamento tra lato client e server;
* Implementazione front-end e back-end di log-in e sign-in

(con funzione di hashing password);

* Stesura codice dell’algoritmo di calcolo del percorso.

**Descrizione delle operazioni di scambio client-server**

**1-SIGN-IN**

Client invia nome e password

Server fa hash della password e, se nome utente e password non sono già presenti, salva i dati nel DB

Server invia esito dell'operazione

**2-LOG-IN**

Client invia nome e password

Server fa hash della password e, se nome utente e password sono presenti, restituisce successo

Server invia esito dell'operazione

**3-SCELTA CITTA’**

La scelta della città da parte dell'utente viene fatta tramite una query sul database dei luoghi; l'utente seleziona la città di suo interesse, il server riceve la città ed esegue un'altra query, per restituire tutti i luoghi di tale città

**4-SCELTA LUOGHI+ REQUISITI DI VIAGGIO**

L'utente seleziona una sequenza di luoghi che è interessato a visitare; dopo aver fatto ciò inserisce i vari requisiti di viaggio (giorni/luogo di partenza/soste/rapidità del passo/orari di partenza-ritorno in albergo). Tutti questi dati sono poi inviati al server.

**5-RECUPERO TABELLA DI MARCIA**

Il server riceve i dati dal client e calcola il percorso migliore per visitare più luoghi possibili dato il tempo a disposizione; la tabella di marcia è salvata su un database per riferimento futuro e viene restituita al client, per visione dell'utente.