Progetto Password Strength Meter
File B – Descrizione dei flussi (Diagrammi di Sequenza)

Indice

1 Scopo

Questo documento accompagna i diagrammi di sequenza (raccolti nel **File A**) e descrive i flussi in modo operativo: ordine delle chiamate, dati scambiati, iterazioni e casi alternativi. L'obiettivo e' rendere evidente **quale comportamento implementeremo** e come i componenti si coordinano tra loro.

2 DS1 – Compilazione credenziali con valutazione in tempo reale

Riferimento: File A – DS1.

2.1 Scenario

Durante la compilazione del form di accesso/registrazione, la password viene valutata in tempo reale e l'interfaccia aggiorna livello di forza e suggerimenti. Al momento della conferma viene eseguita una validazione finale (requisiti minimi).

2.2 Flusso principale

1. L'interfaccia carica il form e rende disponibili gli elementi di feedback (indicatore livello, area suggerimenti).
2. Finche' l'utente modifica il campo password, l'interfaccia invia la password corrente al motore di valutazione.
3. Il motore calcola score/livello e rileva eventuali pattern tramite il modello/regole.
4. Il risultato viene restituito all'interfaccia; se previsto, viene generato anche il feedback testuale (messaggi e suggerimenti).
5. L'interfaccia aggiorna immediatamente gli elementi a schermo (forza + suggerimenti).

2.3 Casi alternativi

- **Conferma form (password accettabile):** viene richiesta la validazione finale sui requisiti minimi; l'esito e' OK e il flusso prosegue verso l'invio delle credenziali.
- **Conferma form (password non accettabile):** validazione finale KO; l'interfaccia mostra un errore e l'utente resta sul form, continuando a modificare la password.
- **Abbandono:** l'utente annulla/chiude la pagina e non viene inviato nulla.

2.4 Implicazioni implementative

- Funzione veloce e idempotente `evaluate(password)` che ritorna almeno `score`, `level`, `patterns`.
- Funzione separata `validateFinal(password)` che controlla requisiti minimi e ritorna `OK/KO` con motivazione.

- Generazione feedback come step distinto (riutilizzabile anche altrove): `generateFeedback(evaluation)`.

3 DS2 – Valutazione password tramite API

Riferimento: File A – DS2.

3.1 Scenario

Un sistema esterno invoca un endpoint HTTP per ottenere la valutazione di una password. La risposta contiene score e livello; su richiesta puo' includere anche suggerimenti.

3.2 Flusso principale

1. Il client esterno invia una richiesta HTTP (es. `POST /evaluatePassword`) con la password e le opzioni.
2. Il servizio valida la richiesta (campi minimi, formato, limiti).
3. Il motore esegue la valutazione e produce `score/level/patterns`.
4. Il modulo API costruisce la risposta JSON e la restituisce al client.

3.3 Varianti e gestione esiti

- **Suggerimenti richiesti:** se nelle opzioni e' presente un flag (es. `includeFeedback=true`), il servizio aggiunge messaggi e suggerimenti nella risposta.
- **Richiesta non valida:** risposta di errore coerente (es. `400`) con motivazione (campo mancante/formato errato).
- **Autorizzazione (se prevista):** risposta `401/403` quando il client non e' abilitato.

3.4 Implicazioni implementative

- Endpoint REST `/evaluatePassword` con DTO di input/output (JSON).
- Riutilizzo della stessa funzione centrale di valutazione usata in DS1 e DS3.
- Risposta standardizzata: sempre score/livello, dettagli aggiunti solo quando richiesti.

4 DS3 – Esecuzione test comparativo tra PSM e baseline

Riferimento: File A – DS3.

4.1 Scenario

Viene avviato un esperimento su un dataset: per ogni record si calcola il punteggio con il PSM e con un baseline esterno. I risultati vengono salvati e, a fine esecuzione, viene generato un report comparativo.

4.2 Flusso principale

1. L'amministratore seleziona dataset e parametri del test (inclusa la configurazione del baseline) e avvia l'esecuzione.
2. Il sistema prepara l'esperimento (apertura dataset, inizializzazione risorse, creazione metadati esperimento).
3. Per ogni record del dataset:
   (a) viene letta la password (o stringa) corrente;
   (b) viene calcolato il punteggio del PSM;
   (c) viene richiesto il punteggio al baseline tramite un adapter;
   (d) il sistema salva il confronto per quel record (input + output PSM + output baseline o errore baseline).
4. Terminata l'iterazione, il sistema aggrega i risultati e costruisce un report (statistiche e confronto complessivo).

4.3 Casi alternativi

- **Baseline OK:** il record viene salvato con entrambe le valutazioni (PSM e baseline).
- **Baseline KO (timeout/errore/output non valido):** il record viene comunque salvato con valutazione PSM e un campo errore per il baseline; l'esecuzione prosegue sul dataset.

4.4 Implicazioni implementative

- Un `ExperimentRunner` che orchestri lettura dataset, scoring e persistenza record-per-record.
- Un `BaselineAdapter` isolato (chiamate esterne, parsing output, gestione errori).
- Persistenza dei risultati e generazione report a fine esperimento (aggregazioni su dati salvati).

5 DS4 – Visualizzazione risultati e statistiche degli esperimenti

Riferimento: File A – DS4.

5.1 Scenario

L'amministratore consulta gli esperimenti eseguiti: vede una lista, applica filtri/ordinamenti e apre il dettaglio con risultati e statistiche.

5.2 Flusso principale

1. L'interfaccia apre la sezione "Risultati esperimenti" e richiede la lista degli esperimenti disponibili.
2. Il sistema recupera i metadati degli esperimenti e restituisce la lista (iniziale).
3. L'amministratore applica criteri di ricerca (filtri/ordinamenti) e la lista viene aggiornata.
4. L'amministratore seleziona un esperimento e richiede il dettaglio.
5. Il sistema recupera i risultati associati a quell'esperimento e costruisce la vista di dettaglio (statistiche e grafici comparativi).

5.3 Casi alternativi

- **Lista vuota:** viene mostrato "nessun esperimento disponibile" (o equivalente) senza errori tecnici.
- **Esperimento non trovato / senza risultati:** viene mostrato un messaggio informativo e non viene costruita la vista statistica.

5.4 Implicazioni implementative

- Query per lista esperimenti con filtri/ordinamenti e query per dettaglio esperimento (per id).
- Funzioni di aggregazione statistiche basate sui risultati salvati (riutilizzabili anche per export).
- Gestione esplicita degli stati vuoti e degli id non validi.

6 DS5 – Esportazione dei risultati di un esperimento

Riferimento: File A – DS5.

6.1 Scenario

Dalla vista di un esperimento, l'amministratore richiede l'esportazione dei risultati in un formato scaricabile (es. CSV/JSON). Il sistema genera il file e avvia il download (o restituisce un allegato HTTP).

6.2 Flusso principale

1. L'amministratore seleziona "Esporta" e sceglie il formato.
2. Il sistema recupera i risultati dell'esperimento dal repository.
3. I risultati vengono trasformati nel formato richiesto e viene generato un file di export.
4. Il file viene restituito all'interfaccia, che avvia il download (o riceve la risposta HTTP con allegato).

6.3 Casi alternativi

- **Nessun risultato disponibile:** viene restituito un errore informativo (non c'e' nulla da esportare).
- **Formato non supportato:** viene restituito un errore coerente (es. richiesta non valida) e non viene generato alcun file.

6.4 Implicazioni implementative

- Funzione `exportExperiment(id, format)` che legge dati, serializza e produce `bytes + filename + content-type`.
- Supporto minimo a due formati (CSV e JSON) con controlli espliciti sul formato richiesto.
- Riutilizzo delle stesse query di DS4 (dettaglio esperimento) per evitare duplicazioni.