Security & Privacy by Design: Progetto Finale

Il presente progetto ha quale obiettivo la concezione (analisi e design) e la realizzazione (sviluppo, test, preparazione al deployment) di un sistema IT che rispetti i criteri di security e privacy by design trattati nel corso del modulo.

Al fine di mantenere il focus sugli obiettivi menzionati, il progetto fornisce un database già prestabilito sulla base del quale verrà sviluppato il sistema, strutturato in un front-end, un back-end e un sistema di autenticazione e tracciabilità degli eventi, nel rispetto delle norme basilari di privacy.

Database sul quale basare il lavoro: https://github Chinook Database

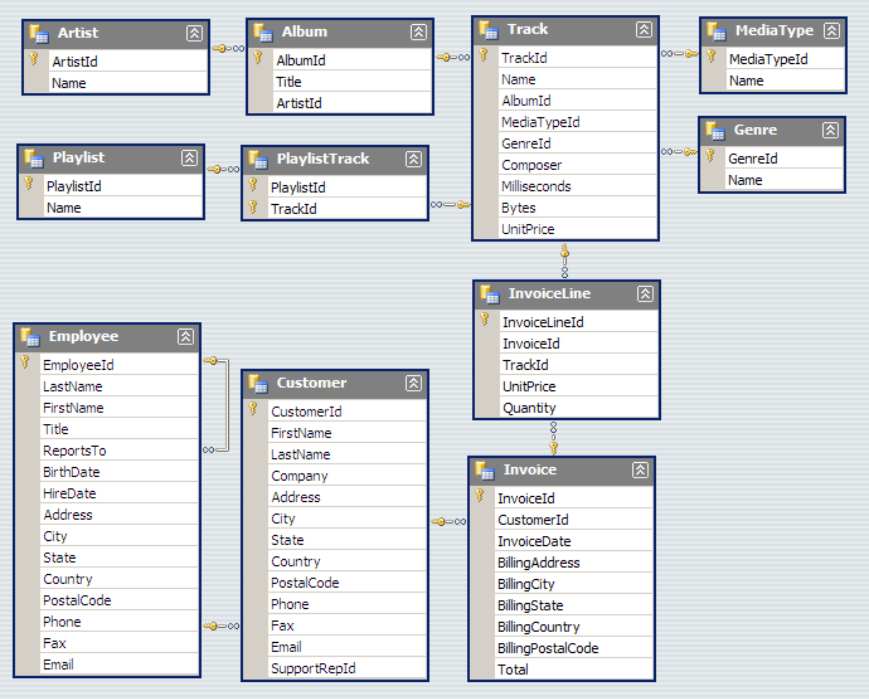
Il database Chinook è un sample database che viene proposto in vari formati, trea i quali:

- SQLite
- PostgreSQL
- MySQL
- Oracle
- MS SQL Server

Il database ha le seguenti caratteristiche:

“The Chinook data model represents a digital media store, including tables for artists, albums, media tracks, invoices and customers.

Media related data was created using real data from an iTunes Library. It is possible for you to use your own iTunes Library to generate the SQL scripts, see instructions below. Customer and employee information was manually created using fictitious names, addresses that can be located on Google maps, and other well formatted data (phone, fax, email, etc.). Sales information is auto generated using random data for a four-year period.”



1. Installare il database nel formato preferito e prendere visione dei dati dell'entità Customer ed Employee: quanti record contengono? Quale è l'impiegato con piú clienti? Quale il cliente con il maggior numero di fatture (Invoice)?

2. Sviluppare una soluzione web-based che possa essere facilmente convertita a una app android, oppure essere perlomeno "mobile friendly".

   Il database potrà essere esteso con nuove entità o attributi di entità esistenti a dipendenza delle necessità del progetto.

3. Specifiche del sistema:

   Sistema: composto da una interfaccia front-end e un sistema back-end.

   Connessione: chiamate API REST.

   Autenticazione: tramite JWT o tecnologia equivalente.

Uso del sistema:

Login come Employee: ogni impiegato deve potersi loggare al sistema, dapprima con una default password “Jo5hu4!” proposta dal sistema.
Qualora necessario adattare il database per contenere le informazioni relative alle password necessarie ad autenticare gli utenti del sistema.
Security by design: proporre una tecnica adeguata per lo stocaggio sicuro delle password.

Deve essere possibile cambiare la propria password, con una password che rispetti requisiti minimi di complessità a libera scelta; suggerimento in caso di dubbi:

- deve esser lunga almeno 6 e massimo 14 caratteri,
- deve contemplare almeno tre delle quattro categorie seguenti:
  - lettere maiuscole (A ... Z)
  - lettere minuscole (a ... z)
  - numeri (0 ... 9)
  - segni di interpunzione e caratteri non alfabetici (ad es. -, !, $, #, %)

Non è richiesto che venga mantenuto lo storico di vecchie password, ma che si verifichi che non venga reinserita la stessa password.

Una volta loggato, l’utente deve poter visualizzare un elenco dei propri clienti (Customers). Deve essere possibile filtrare l’elenco con criteri a scelta.

Logout: l’utente deve potersi sloggare dal sistema. Gli utenti verranno sloggati dal sistema dopo inattività di 2 minuti.

Authorization: gli utenti di tipo manager (i manager sono gli impiegati che hanno la parola “manager” nell’atrtributo “Title” della tabella “Employee”) possono vedere tutti i clienti, mentri gli utenti semplici vedono solo i propri clienti.
Nota: qualora potesse risultare utile è possibile inserire un nuovo attributo “Manager” di tipo booleano all’entità relativa.

Session management: Impostare i token in modo da scadere ogni 5 minuti ed effettuare un refresh automatico per gli utenti manager, mentre gli utenti normali verranno sloggati dal sistema.

Logging e traceability: inserire nel back-end dei logging che permettano di tracciare l'utilizzo del sistema e quindi le attività dell'utente

Data and Input validation: i campi programmati per effettuare query al database devono effettuare controlli di coeerenza dei criteri di ricerca e impedire ricerche non idonee (a ognuno la libertà di scegliere le opzioni adeguate a dimistrare l'implementazione di tali controlli).

Web security: il sistema deve impedire possibilità di attacchi di tipo injection (SQL, Code).

Code analysis: Dimostrare di aver effettuato una piccola analisi con un tool a scelta per identificare rischi a livello di codice.

Error handling: gestire le exception in modo che il sistema non generi messaggi strani e difficili da interpretare per un utente comune.

Risk assessment:

- Identificare 3 rischi del sistema sviluppato e classificarli per gravità (bassa, media o alta).
- Proporre dei miglioramenti

Privacy (domanda opzionale):

- Identificare le informazioni che risultano sensibili a problemi di privacy, anche in base al set di casi di data breach management analizzati in classe.
- Classificare tali informazioni in base alle nmecessità di trattamento e spiegare le scelte.
- Proporre delle soluzioni che permettano di migliorare il sistema e rendelo piú resiliente a problemi di data breach.