

**Indice**

1. **Introduzione1**

1.1 Scopo Del Sistema**1**

1.2 Obiettivi Di Progettazione**1**

1.3 Definizioni, Abbreviazioni e Acronimi **1**

1.4 Riferimenti **1**

1.5 Panoramica: Diagramma delle classi **1**

1. **Sistema Corrente1**
2. **Sistema Proposto1**

3.1 Panoramica **1**

3.2 Decomposizione Del Sistema **2**

3.3 HardWare/SoftWare Mapping **3**

3.4 Gestione Dei Dati Persistenti **4**

3.5 Controllo Accessi e Sicurezza **4**

3.6 Controllo Software Globale **4**

3.7 Condizione Limite **5**

1. **Servizi del sottosistema 17**
2. **Glossario 17**
3. Introduzione

1.1 Scopo Del Sistema

La gestione delle password è sempre stato un tema spesso trascurato dagli utenti d’oggi, con il risultato che molte persone riscontrano difficoltà nel proteggere in modo sicuro e pratico i propri account, utilizzando password banali e semplici da decifrare, incorrendo così sempre più spesso ad attacchi hacker. Nonostante la consapevolezza sull'importanza della sicurezza digitale, sono poche le applicazioni che permettono di gestire e proteggere le password in modo semplice e intuitivo. È per questo che nasce PassBox! Gli utenti avranno a disposizione uno strumento sicuro e facile da usare per archiviare, organizzare e recuperare le proprie password, garantendo accesso rapido e protezione completa delle credenziali.

1.2 Obiettivi di Progettazione

* **Usabilità**: il sistema deve essere facile e intuitivo da usare per l’utente, con funzionalità come opzioni di mostra/nascondi il testo della password durante la digitazione per ridurre gli errori.
* **Sicurezza**: le password devono essere memorizzate in modo sicuro tramite algoritmi di protezione. Implementare tecniche per limitare gli attacchi e offrire l'autenticazione a più fattori per una maggiore sicurezza.
* **Accessibilità**: il sistema deve essere accessibile ai lettori di schermo e rispettare le norme (universal usability), per un uso inclusivo da parte degli utenti con disabilità.
* **Prestazioni**: il sistema dovrebbe essere rapido nelle operazioni di uso, come la funzione di verifica dei requisiti delle password.
* **Portabilità**: il sistema deve supportare anche altri dispositivi (e.g., dimensioni degli schermi sia per desktop che per mobile).
* **Robustezza**: Il sistema dovrebbe gestire gli errori con messaggi specifici per le eccezioni.

**Privacy**: il sistema deve garantire la riservatezza dei dati personali, evitando di memorizzare informazioni sensibili in chiaro (non crittografate).

1.3 Definizioni, Abbreviazioni e Acronimi

Non ci sono definizioni, abbreviazioni e acronimi da aggiungere, per qualsiasi perplessità o dubbio consultare il glossario

1.4 Riferimenti

Nessuna definizione degna di nota, consultare il **glossario** per eventuali dubbi.

1.5 Panoramica: Diagramma delle classi

DA AGGIUNGERE

1. Sistema corrente

PassBox è un progetto Greenfield, ovvero la progettazione e implementazione del sistema vengono realizzati da zero, fornendo la possibilità di scegliere le tecnologie che si ritengono più adatte. Non si deve adattare il proprio progetto per integrarsi con tecnologie e design già esistenti.

1. Sistema proposto

3.1 Panoramica

Per il PassBox utilizziamo un'architettura Client-Server, che garantisce efficienza, modularità e scalabilità. Un sistema basato su Client-Server è particolarmente adatto perchè ci permette di separare l’autenticazione, sicurezza dei dati sensibili (come le password) e l’accesso ai servizi offerti. Le diverse parti coinvolte nel Client-Server sono:

**Client**  
Il client è l’interfaccia utilizzata dall’utente per interagire con il sistema. Il ruolo principale è raccogliere input, come le credenziali dell’utente, e inviare richieste al server.

*Tipologie di client*:

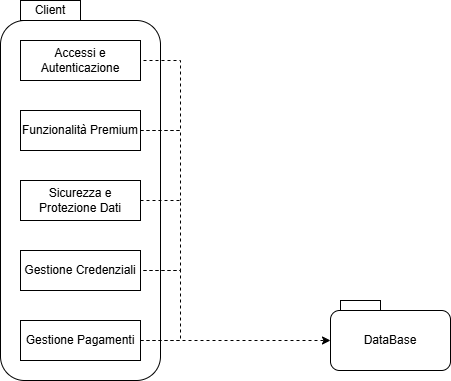
1. *Gestore*: Raccoglie e valida i dati dell’utente (ad esempio, la password) e consente operazioni come l’aggiornamento delle credenziali.
2. *User*: l’utente finale che utilizza il sistema per creare e gestire le proprie credenziali in modo sicuro.

**Server**  
Il server si occupa della gestione delle operazioni fondamentali del sistema, si occupa di:

1. *Autenticazione e sicurezza*: gestione sicura delle credenziali con crittografia e verifica gli utenti (login, 2FA).
2. *Archiviazione Sicura*: Conserva le password crittografate e gestisce le chiavi di crittografia in modo sicuro.
3. *Servizi di amministrazione*: supporta il Gestore nelle attività di aggiornamento della sicurezza e gestione degli account.

Utilizziamo questa architettura poichè ci offre ottime prestazioni e ci da un elevata affidabilità nella gestione e conservazione dei dati.

3.2 Decomposizione del sistema



1. **Accessi e Autenticazione**

* *Register*: Fase di registrazione inserendo le credenziali utente con autenticazione a 2 fattori (2FA) tramite OTP.
* *Login*: Fase di accesso con credenziali utenti, autenticazione a 2 fattori (2FA) tramite OTP.

1. **Funzionalità Premium**

* *Autenticazione a 2 Fattori:* Possibilità di aggiungere un ulteriore controllo di sicurezza nei vari vault aggiungendo un autenticazione a 2 fattori nelle singole credenziali.
* *Scadenza Password:* Possibilità di avere un calendario che notifica la data di scadenza della password/credenziali, definita dall’utente.

1. **Sicurezza e Protezione Dati**

* *Crittografia Dati:* Tutte le password sono crittografate.
* *Importazione/Esportazione Vault:* Possibilità si importare ed esportare i vault con gli account e le credenziali utente.

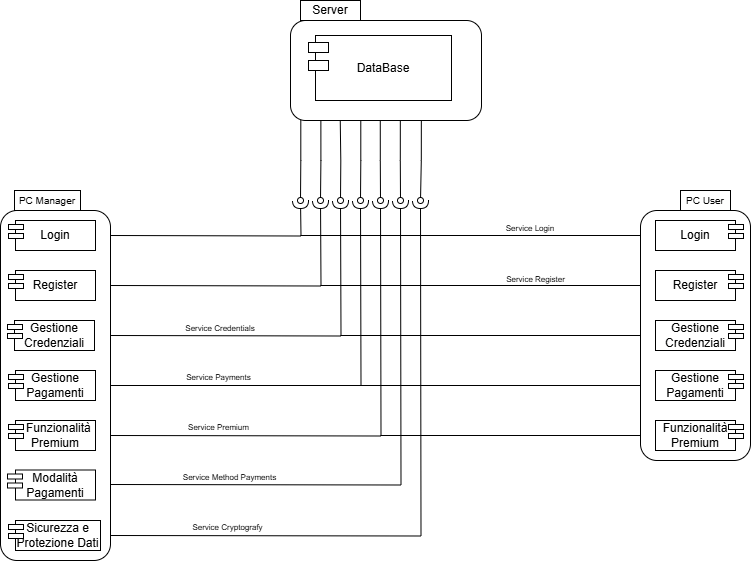
1. **Gestione Credenziali**

* *Aggiunta Credenziali:* inserimento di nuove credenziali (Esempio. Username, Password), e utilizzo di un generatore di password.
* *Modifica Credenziali:* aggiornamento delle credenziali esistenti.
* *Eliminazione Credenziali:* Rimozione delle credenziali non più sicure/necessarie.

1. **Gestione Pagamenti**

* *Aggiunta Metodo di Pagamento:* Inserimento di carte di credito/debito o altri metodi di pagamento.
* *Modifica Metodo di Pagamento:* Aggiornamento del metodo di pagamento.
* Eliminazione *Metodo di Pagamento:* Rimozione dei metodi di pagamento.
* *Gestione Abbonamenti Premium:* Upgrade ad un account premium tramite pagamento, rinnovo o cancellazione dell’abbonamento.

3.3 HardWare/SoftWare Mapping



3.4 Gestione dei dati persistenti

I dati persistenti di PassBox, si limitano ad un database NoSql gestito da MongoDB, dato che l’applicazione-web gestisce e controlla informazioni sensibili, è necessaria una crittografia, per garantire affidabilità e coerenza di questi dati.

3.5 Controllo accessi e sicurezza

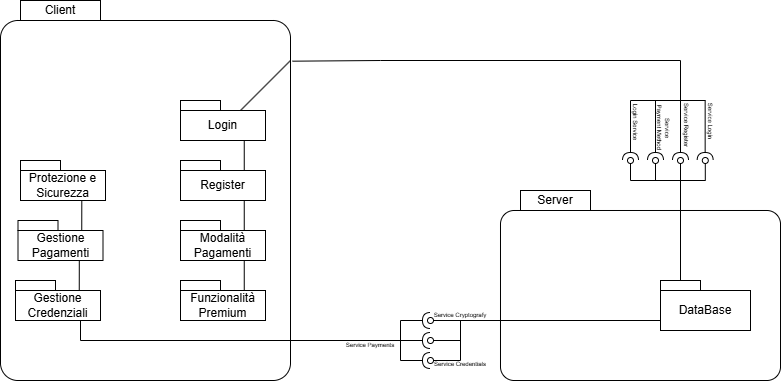
3.6 Controllo software globale

Il sistema PassBox, strutturato secondo un’architettura Client/Server, è molto probabile che debba gestire un massiccio numero di connessioni simultanee dovute alla presenza di numerosi utenti. Per affrontare questo problema si utilizza una combinazione di tecniche thread-based e multiplexing. Questo metodo ci permette di ottimizzare la gestione delle richieste contemporanee, dandoci una sincronizzazione efficace dei thread e proteggendo le risorse condivise da eventuali conflitti.

3.7 Condizioni limite

L’architettura scelta per PassBox è server-based, ovvero, tutte le funzionalità e servizi siano incentrati su un unico server, infatti se si dovesse andare incontro a qualche guasto ci sarebbe l’interruzione dell’servizio per ogni utente. La gestione del server potrebbe essere subappaltata, però nel caso di PassBox, il server viene gestito internamente per garantire un maggiore controllo sulla sicurezza, sulla privacy e sull'implementazione delle funzionalità, evitando che siano delegate a terzi. Tuttavia, questo significa che ogni configurazione, manutenzione relativa al server, così come la sua eventuale temporanea sospensione, sono responsabilità del team tecnico interno, con possibili implicazioni anche sui costi.

4. Servizi del sottosistema



5. Glossario

Nessun vocabolo da segnalare