Progetto - Ingegneria del Software



IdP con protocollo OAuth2.0 e 2FA

Corso Ingegneria del Software e Fondamenti Web

Docente Mongiello Maria

Mentore progetto Dell’Olio Giacomo Alberto

Gruppo **14**

Putignano Gianluca MAT – 588237 mail – [g.putignano11@studenti.poliba.it](mailto:g.putignano11@studenti.poliba.it)

Raimondi Federico MAT – 587660 mail – [f.raimondi@studenti.poliba.it](mailto:f.raimondi@studenti.poliba.it)

Salluce Luca MAT – 587863 mail – [l.salluce@studenti.poliba.it](mailto:l.salluce@studenti.poliba.it)

Troilo Stefano MAT – 588045 mail – [s.troilo2@studenti.poliba.it](mailto:s.troilo2@studenti.poliba.it)

Volpe Antonio MAT – 588757 mail – [a.volpe18@studenti.poliba.it](mailto:a.volpe18@studenti.poliba.it)

Sommario

[1. Introduzione al Progetto 2](#_Toc122440318)

[Descrizione 3](#_Toc1814792844)

[Obiettivo 3](#_Toc640369211)

[Ambito 3](#_Toc676751463)

[Articoli Scientifici 4](#_Toc1956977250)

[2. Analisi dei Requisiti 4](#_Toc2098288235)

[Requisiti funzionali 4](#_Toc933667980)

[Autenticazione OAuth 2.0 4](#_Toc610364935)

[Autenticazione a Due Fattori (2FA) 5](#_Toc661620368)

[Supporto per Autenticazione Federata 6](#_Toc1791426987)

[Gestione dei Token 7](#_Toc1385945185)

[Requisiti non funzionali 8](#_Toc1051392306)

[Sicurezza Elevata 8](#_Toc2096025850)

[Scalabilità e Performance 9](#_Toc144413199)

[Vincoli Tecnologici 10](#_Toc1511724206)

[Protocollo OAuth 2.0 10](#_Toc517895871)

[Compatibilità 2FA 11](#_Toc1715568686)

[3. Analisi delle Soluzioni Esistenti 12](#_Toc567399368)

[Identity Providers Affermati 12](#_Toc1944007395)

[Panoramica dei principali IdP 12](#_Toc897282893)

[Funzionalità di Sicurezza e 2FA (da controllare, rileggere e modificare) 13](#_Toc2128625827)

[Confronto di Caratteristiche 13](#_Toc1491638719)

[Usabilità e Scalabilità 13](#_Toc1468818182)

[Flessibilità e Integrazione 14](#_Toc1334798781)

[Differenziazione del Progetto 15](#_Toc1799886019)

[4. Progettazione dell’Architettura 15](#_Toc1403262010)

[Schema del Sistema 15](#_Toc510742107)

[Componenti Principali 15](#_Toc1732785012)

[Flusso di Autenticazione 15](#_Toc1288046328)

[Diagramma di Architettura 16](#_Toc930526475)

[5. Fasi di Sviluppo e Pianificazione 16](#_Toc1417994267)

[Fasi del Progetto 16](#_Toc1805737800)

[Analisi dei Requisiti 16](#_Toc2083331574)

[Progettazione 16](#_Toc23444143)

[Implementazione IdP 16](#_Toc1338768977)

[Integrazione 2FA 16](#_Toc82789366)

[Test 16](#_Toc586430626)

[Timeline 17](#_Toc1099728342)

[Risorse Necessarie 17](#_Toc1347555058)

[6. Implementazione del Sistema 17](#_Toc66849023)

[Configurazione Server IdP 17](#_Toc391339665)

[Gestione Utenti e Permessi 17](#_Toc816317692)

[Integrazione e Verifica 2FA 17](#_Toc1451333939)

[Generazione OTP 17](#_Toc351141829)

[Validazione OTP 18](#_Toc1749983719)

[Gestione Token 18](#_Toc331053048)

[7. Test e Validazione 18](#_Toc1093978348)

[Test di Sicurezza 18](#_Toc2084032799)

[Test Funzionali 18](#_Toc1818487952)

[Test di Performance 18](#_Toc774124138)

[Test con Utenti Finali (UAT) 18](#_Toc787874517)

[8. Manutenzione e Futuri Aggiornamenti 18](#_Toc677803278)

[Manutenzione Continua 19](#_Toc763484202)

[Possibili Estensioni 19](#_Toc754167753)

[Pagine web 19](#_Toc312716832)

# Introduzione al Progetto

## Descrizione

Il progetto scelto, valido come prova parziale dell’insegnamento di Ingegneria del Software, riguarda la creazione di un Identity Provider (IdP) sicuro ed efficace che si affida al protocollo OAuth 2.0 per la gestione e il controllo degli accessi, aggiungendo un ulteriore strato di sicurezza dato dal sistema di autenticazione a due fattori (2FA).

## Obiettivo

L’obiettivo è fornire un sistema che bilanci sicurezza e facilità d’uso per gli utenti, rispettando le *normative di sicurezza*[[1]](#endnote-2) e riducendo i rischi di accesso non autorizzato:

* Realizzare un sistema centralizzato di autenticazione (IdP)
* Implementare il protocollo OAuth 2.0
* Gestire token di accesso sicuri
* Gestire le sessioni utente in modo sicuro

## Ambito

L'autenticazione a due fattori (2FA) è un protocollo di sicurezza adottato in svariati ambiti per proteggere l'accesso alle risorse come account e servizi online. È impiegata soprattutto quando serve tutelarsi da accessi non autorizzati, aggiungendo uno strato di sicurezza in più rispetto alla tradizionale autenticazione basata solo su una singola password. Questi sono alcuni degli ambiti principali in cui viene utilizzata:

* **Accesso agli Account Online:** l’autenticazione a due fattori è utilizzata soprattutto per proteggere account online, come quelli sui social, nelle piattaforme di posta elettronica, e nei servizi di archiviazione come Google Drive o Dropbox. Questo provvedimento riduce il rischio di accesso da parte di terze persone anche se arrivano a conoscenza della password.
* **Bancario e Finanziario:** i servizi bancari e finanziari sono tra i principali ad adottare la 2FA poiché hanno il compito di proteggere e riservare informazioni sensibili, nonché il denaro dei propri clienti. Qui, la 2FA assume diverse forme, dall'uso di codici temporanei inviati via SMS all'uso di app specifiche che generano delle key temporanee valide per pochi secondi.
* **Aziendale e IT:** Le aziende utilizzano la 2FA per proteggere l'accesso alle reti e ai sistemi aziendali, specialmente nel momento in cui i dipendenti lavorano in smart working. Questa pratica contribuisce a ridurre il rischio di accessi non autorizzati a dati sensibili dell’azienda, come documenti e schede contenenti dati personali di ogni cliente
* **E-Commerce:** Anche il settore dell'e-commerce, come Amazon e tanti altri siti di vendita online, utilizza la 2FA per proteggere i pagamenti online e gli account utente. Molti siti richiedono un’ulteriore verifica prima di completare con la transazione o di salvare le credenziali della propria carta di credito per eventuali acquisti futuri.
* **Sanità:** anche in ambito sanitario la 2FA rende protetti gli accessi ai sistemi di gestione dei dati clinici e dei pazienti, mantenendo così una discreta riservatezza per quanto riguarda le condizioni dei pazienti del reparto ospedaliero.

## Articoli Scientifici

1.L'autenticazione a due fattori è considerata una misura di sicurezza estremamente efficace per proteggere account e informazioni personali, riducendo significativamente il rischio di accessi non autorizzati. Questo approccio combina due elementi distinti: qualcosa che l'utente conosce (ad esempio, una password) e qualcosa che l'utente possiede (come uno smartphone o un token) o qualcosa che l'utente è (dati biometrici come impronte digitali o riconoscimento facciale) **Fonte**: Studente Dottorando presso Western Nevada College

2. Uno studio evidenzia come la 2FA riduca l'efficacia degli attacchi di phishing e renda difficile l'accesso ai sistemi anche in caso di conoscenza della password. Infatti, grazie all'uso di app come Duo Mobile per generare codici univoci o notifiche push, rende gli account più sicuri rispetto all'utilizzo esclusivo delle password. ￼ **Fonte:** Boston University

3. Nonostante i vantaggi, l'efficacia dipende dall'implementazione e dalla configurazione corretta del sistema. Un corretto utilizzo della 2FA può proteggere dati sensibili e conformarsi a normative di sicurezza, come quelle richieste da organizzazioni governative o accademiche. **Fonte**: Western Nevada College

# Analisi dei Requisiti

## Requisiti funzionali

I requisiti funzionali sono fondamentali nella natura di un progetto perché definiscono le caratteristiche e le capacità specifiche che il sistema deve fornire per rispondere alle esigenze degli utenti e agli obiettivi del progetto. Questi guidano l’implementazione tecnica e garantiscono che il sistema svolga le funzioni essenziali in modo efficace. La loro chiarezza è cruciale per un corretto sviluppo, per una corretta fase di testing e per verificare il mantenimento del prodotto.

### Autenticazione OAuth 2.0

Questo framework di autenticazione rappresenta uno standard ampiamente riconosciuto che consente a un utente di autorizzare un sito web o un’applicazione di terze parti ad accedere alle proprie risorse protette, senza dover condividere direttamente le proprie credenziali o rivelare la propria identità.

Si impiegherà il framework OAuth 2.0 per l’**autenticazione** **dei** **client** e per la **gestione del flusso di autorizzazione**. Inoltre, si terrà traccia della **cronologia degli accessi effettuati** dagli utenti.

Questo protocollo introduce significativi vantaggi rispetto ai metodi tradizionali di autenticazione, tra cui:

* **Sicurezza**

L’utente non è più obbligato a fornire le proprie credenziali ad applicazioni di terze parti, riducendo il rischio di compromissione delle informazioni sensibili.

* **Flessibilità**

OAuth 2.0 si adatta a una vasta gamma di scenari applicativi, spaziando dalle applicazioni web tradizionali ai dispositivi IoT, garantendo soluzioni efficaci e personalizzabili.

* **Standardizzazione**

Essendo un protocollo aperto, il framework favorisce l’interoperabilità tra piattaforme e applicazioni, semplificando l’integrazione tra sistemi diversi.

Queste caratteristiche fanno di OAuth 2.0 un elemento chiave per la gestione dell’accesso alle risorse protette in un ecosistema digitale moderno e in continua evoluzione.

### Autenticazione a Due Fattori (2FA)

L’autenticazione a due fattori (2FA) è una **misura di sicurezza** che richiede l’utilizzo di **due diversi fattori di identificazione** per garantire l’accesso a un sistema o servizio. In questo modo si va ad aggiungere un ulteriore livello di protezione rispetto alla semplice combinazione di nome utente e password, rendendo più difficile a persone non autorizzate di accedere a informazioni sensibili.

La 2FA si basa su una combinazione di due tra i seguenti fattori di autenticazione:

* **Fattore di conoscenza:** qualcosa che solo l’utente sa, come una password o un PIN.
* **Fattore di possesso:** qualcosa che solo l’utente possiede, come un token hardware o un dispositivo mobile.
* **Fattore di inerenza:** qualcosa che caratterizza l’utente (dato biometrico), ad esempio un’impronta digitale o il riconoscimento facciale.

Il processo di autenticazione segue degli step ben precisi per ottimizzarne il funzionamento:

1. **Accesso iniziale:** l’utente inserisce nome utente e password.
2. **Richiesta del secondo fattore**, il sistema richiede una verifica aggiuntiva, come un codice OTP (one-time password) inviato via SMS/e-mail o generato da un app.
3. **Verifica dei fattori**, se i due fattori sono corretti, l’accesso viene garantito.

Si hanno differenti metodi di applicazione del 2FA:

* **Autenticazione via e-mail o SMS**

Viene inviato un codice OTP all’utente. È un metodo semplice e accessibile ma vulnerabile a minacce come l’intercettazione degli SMS.

* **Token hardware**

L’utente dispone di un dispositivo fisico come una chiavetta o una smart card che va a generare un codice identificativo, se questo codice combacia con il codice creato dal server, basato sullo stesso metodo, allora verrà concesso l’accesso all’utente. Metodo sicuro ma meno pratico e soggetto a smarrimenti.

* **Token software**

Codici generati da app di autenticazione, un buon compromesso tra sicurezza e praticità.

* **Notifiche push**

L’utente approva l’accesso tramite un app sicura. Questo metodo riduce il rischio di *attacchi di tipo man-in-the-middle*[[2]](#endnote-3).

* **Autenticazione biometrica**

Utilizza tratti unici dell’utente come impronte digitali o il riconoscimento facciale. È una delle opzioni più sicure e all’avanguardia, oggi disponibile in gran parte dei siti web.

Considerando ora il concetto in sé di 2FA si riesce a capire ancora meglio la sua importanza, proprio perché le password da sole non sono più sufficienti a garantire la sicurezza, poiché possono essere facilmente violate tramite ingegneria sociale, attacchi di phishing o brute force. La 2FA riduce significativamente questi rischi, richiedendo un secondo livello di verifica per accedere a un account. La 2FA gioca un ruolo chiave nell’implementazione dell’*architettura* [[3]](#endnote-4)￼, garantendo che ogni accesso venga verificato rigorosamente e tenendo conto di parametri come dispositivo, comportamento e posizione.

Una buona implementazione dell’autenticazione 2FA si basa su:

* **Scegliere fattori di autenticazione adatti** al caso d’uso e alla sensibilità delle informazioni protette.
* **Ottimizzare l’esperienza utente (UX)** per garantire un flusso semplice senza compromettere la sicurezza.
* **Crittografare le comunicazioni** per proteggere i dati durante la trasmissione.
* **Considerare l’autenticazione adattiva**, che adatta il livello di verifica in base al rischio percepito.

Considerando anche il concetto di **autenticazione multifattoriale** (MFA), la generalizzazione della 2FA, si nota che questa può includere più di due fattori di verifica. La MFA offre un livello di sicurezza ancora più elevato, rendendola ideale per contesti ad alto rischio.

### Supporto per Autenticazione Federata

Nel contesto di un Identity Provider (IdP) che implementa OAuth 2.0 e autorizzazione a due fattori, il supporto per l’autenticazione federata consente l’integrazione con sistemi di identità esterni. Questo approccio permette agli utenti di autenticarsi tramite *fornitori di identità di terze parti*[[4]](#endnote-5), senza dover creare credenziali dedicate per ogni applicazione o servizio, garantendo un accesso sicuro e semplificato.

Questo tipo di autenticazione è basata su protocolli standard aperti, come OAuth 2.0, OpenID Connect e SAML (Security Assertion Markup Language), che permettono la condivisione sicura di informazioni di autenticazione tra domini diversi. Questo approccio offre significativi vantaggi in termini di interoperabilità, sicurezza e semplificazione gestionale.

Le Caratteristiche principali per l’autenticazione federata sono:

* **Accesso centralizzato**

Gli utenti possono accedere a più applicazioni utilizzando le credenziali di un unico fornitore di identità fidato, come Google, Microsoft o Facebook, eliminando la necessità di ricordare tutte le password di accesso.

* **Interoperabilità**

Grazie ai protocolli standard, l’IdP può integrarsi con una vasta gamma di sistemi e piattaforme, migliorando la scalabilità del progetto.

* **Sicurezza migliorata**

Combinando l’autenticazione federata con il 2FA, il sistema garantisce un livello di sicurezza avanzato, riducendo il rischio di accessi non autorizzati e proteggendo le informazioni sensibili.

* **Esperienza utente ottimizzata**

Gli utenti beneficiano di un processo di accesso semplificato, che riduce i tempi e le difficoltà associati alla creazione e alla gestione di account multipli.

### Gestione dei Token

Tutto ciò che viene realizzato attraverso OAuth 2.0 si basa sull’utilizzo di token, elementi centrali nel processo di autorizzazione. I token sono stringhe firmate da un server, progettate per garantire l’integrità e l’autenticità delle informazioni relative agli utenti e alle autorizzazioni.

I token possono essere di due tipologie principali:

* **Token di Accesso (Access Token)**

Questo è il token principale utilizzato per autorizzare l’accesso alle risorse protette. Viene incluso nelle richieste inviate al server di risorse per dimostrare l’autenticità dell’utente o dell’applicazione. Gli Access Token hanno una durata limitata per garantire una maggiore sicurezza.

* **Token di Refresh (Refresh Token)**

Questo tipo di token consente di ottenere un nuovo Access Token senza dover richiedere nuovamente l’autorizzazione dell’utente. I Refresh Token hanno una durata più lunga rispetto agli Access Token e sono utilizzati per mantenere sessioni a lungo termine in modo sicuro.

Ogni token è composto da tre parti distinte che lavorano in modo sinergico:

* **Header:** include informazioni sul tipo di token e sull’algoritmo utilizzato per la sua generazione.
* **Payload:** contiene le informazioni più importanti, come quelle sull’utente, le autorizzazioni concesse e le scadenze.
* **Signature:** utilizzata per verificare l’autenticità del token. La firma viene generata applicando un algoritmo di codifica in hash, garantendo che il token non sia stato modificato dopo la sua emissione.

In OAuth 2.0 bisogna anche tener conto delle diverse tipologie di token di accesso che possono essere utilizzate per autorizzare l’interazione con risorse protette. Questi token svolgono un ruolo fondamentale nella gestione dei diritti di accesso e nell’implementazione di politiche di sicurezza, garantendo un controllo granulare e una protezione avanzata dei dati. Di seguito vengono descritte le principali categorie di token, ognuna con uno scopo specifico:

* **Token di accesso utente:** consente di autorizzare le operazioni sui dati dell’utente. La sua generazione richiede il consenso esplicito dell’utente, generalmente ottenuto tramite un modulo apposito. È un elemento chiave per garantire la sicurezza e il trattamento delle informazioni personali.
* **Token di accesso dell’app:** utilizzato per autorizzare l’accesso ai dati relativi all’applicazione stessa. La sua generazione avviene tramite una password predefinita condivisa tra l’applicazione e il servizio, come nel caso di Facebook. È ideale per operazioni di amministrazione o gestione tecnica.
* **Token di accesso della pagina:** necessario per autorizzare l’interazione con i dati associati a una pagina. La generazione di questo token richiede preventivamente un token di accesso utente, assicurando così che solo utenti autorizzati possano accedere o modificare le informazioni della pagina.
* **Token client:** identificativo specifico che viene incorporato nel codice binario delle applicazioni. Serve per l’identificazione dell’applicazione stessa, fornendo un ulteriore livello di tracciabilità e sicurezza.

## Requisiti non funzionali

I requisiti non funzionali sono essenziali per garantire che il sistema soddisfi standard di qualità come sicurezza, performance, scalabilità e usabilità. Stabiliscono il modo con cui il sistema deve operare ed assicurano che il sistema sia affidabile, efficiente e adatto a un utilizzo reale e sostenibile nel tempo.

### Sicurezza Elevata

OAuth 2.0, come framework estensibile, richiede specifiche implementazioni di sicurezza che dipendono dal contesto applicativo. Le linee guida di sicurezza affrontano vari aspetti critici:

* **Autenticazione del client:** il server di autorizzazione deve autenticare i client, evitando di fornire credenziali a client nativi o basati su user-agent. Per client non autenticati, è necessario adottare misure compensative, come la validazione dell’URI di reindirizzamento o il coinvolgimento del proprietario delle risorse.
* **Protezione delle credenziali e dei token**:

- I token di accesso e aggiornamento devono essere trasmessi e conservati in modo sicuro, utilizzando TLS per prevenire compromissioni.

- Le credenziali sensibili, come i codici di autorizzazione e le password, devono essere trattate con particolare attenzione per evitare esposizioni accidentali.

* **Impersonificazione e attacchi CSRF**:

- I server devono implementare controlli per prevenire l’impersonificazione del client e attacchi cross-site request forgery (CSRF), utilizzando il parametro “state” per verificare la legittimità delle richieste.

- Per evitare manipolazioni dell’URI di reindirizzamento, l’URI registrato deve essere validato e corrispondere esattamente a quello utilizzato nelle richieste.

* **Gestione di attacchi phishing e clickjacking**:

- Per gestire al meglio il rischio di attacchi phishing, è essenziale sensibilizzare gli utenti sull’importanza di riconoscere e segnalare tentativi di frode, oltre a fornire linee guida chiare per un utilizzo sicuro del sistema. Per contrastare il clickjacking, il sistema implementa misure che impediscono il caricamento delle pagine di autorizzazione all’interno di frame non autorizzati.

- Gli sviluppatori devono garantire che gli endpoint siano autenticati tramite TLS e rendere difficile l’imitazione di pagine di autorizzazione legittime.zs

* **Rischi legati a grant types**:

- L’utilizzo delle credenziali del proprietario della risorsa è fortemente sconsigliato a causa dell’elevato rischio associato.

- I flussi impliciti, che possono esporre i token tramite frammenti URI, devono essere utilizzati con cautela e solo con misure di sicurezza aggiuntive.

* **Valutazione della sicurezza dei client pubblici**: i client pubblici non devono presupporre che un token sia valido per il proprio utilizzo senza verifiche aggiuntive. Meccanismi di limitazione e autenticazione del token sono essenziali per prevenire abusi.

### Scalabilità e Performance

Il sistema di Identity Provider (IdP) sviluppato con il protocollo OAuth 2.0 e autenticazione a due fattori (2FA) è stato progettato per soddisfare rigorosi requisiti di scalabilità e performance, garantendo un’esperienza fluida e sicura anche in presenza di carichi elevati.

* **Scalabilità**

La scalabilità è un concetto pensato per rispondere alle esigenze di crescita del numero di utenti e al potenziale aumento delle richieste simultanee.

Ciò che caratterizza principalmente questo concetto è:

- **Architettura**: In questo caso il sistema è stato progettato su un’architettura modulare basata su dei microservizi. In questo modo si può applicare un concetto di scalabilità in modo orizzontale così da aggiungere ulteriori istanze, come gestione dei token e autenticazione, senza influenzare ulteriori componenti

- **Bilanciamento del carico**: Le richieste in ingresso vengono distribuite uniformemente tra i server backend per evitare colli di bottiglia, ovvero quando una risorsa specifica non riesce a gestire in modo corretto il carico di lavoro. Pertanto, in questi si casi si prediligono differenti algoritmi per la gestione dei carichi:

• **Round Robin**: per una distribuzione uniforme delle richieste.

• **Least Connections**: per inviare le richieste al server meno occupato.

• **IP Hashing**: per garantire che un utente venga sempre associato a llo stesso server quando necessario.

- **Caching distribuito**: Per ottimizzare le performance e alleggerire il carico sui database, il sistema adotta un approccio di caching distribuito. Questo meccanismo consente di memorizzare temporaneamente informazioni frequentemente richieste, come i token di autenticazione e i dettagli delle sessioni utente. Applicando questo metodo e richieste successive possono essere soddisfatte direttamente dal cache, evitando di interrogare ripetutamente il database principale. Questo approccio non solo riduce la latenza delle operazioni, ma migliora anche la capacità del sistema di gestire carichi elevati.

* **Performance**

Per garantire elevati standard di velocità ed affidabilità vengono implementate differenti strategie

- **Riduzione della latenza**: Il sistema è ottimizzato per garantire tempi di risposta inferiori molto rapidi per operazioni frequenti, come la generazione e la validazione dei token basandosi sulla gestione parallela delle richieste.

- **Monitoraggio delle prestazioni**: Vengono analizzate tutte le caratteristiche specifiche come throughput, latenza e utilizzo delle risorse. Inoltre, i log generati dal sistema vengono analizzati in tempo reale per individuare eventuali anomalie.

- **Test di carico e ottimizzazioni**: Prima di utilizzare il sistema per situazioni reali, questo viene sottoposto ad alcuni stress test per simulare scenari con elevato traffico. In questo modo possono essere ottimizzati alcuni colli di bottiglia che non permettono al sistema di performare al meglio.

## Vincoli Tecnologici

### Protocollo OAuth 2.0

L’obiettivo dell’implementazione è quello di garantire un’integrazione conforme alle specifiche standard di OAuth 2.0 per supportare flussi di autorizzazione sicuri e scalabili, adattati alle esigenze del sistema. Si prendono in analisi differenti caratteristiche:

* **Dettagli implementativi**

- **Authorization Server**:

* + Genera e gestisce i token di accesso e di refresh, convalidando client e utenti in base ai flussi specificati.
  + Configurato per gestire i parametri di sicurezza richiesti, come redirect\_uri, scope, e state.
  + Supporta estensioni per 2FA, compatibile con metodi OTP conformi agli standard RFC 6238 (TOTP) e RFC 4226 (HOTP).
* **Resource Server**:
  + Convalida i token ricevuti dall’Authorization Server utilizzando tecniche come la verifica della firma di token JWT.
  + Garantisce che i token siano validi, non scaduti e con gli opportuni privilegi (scope).
* **Client Application**:
  + Le applicazioni client seguono processi di autenticazione e autorizzazione definiti, con registrazione basata su client\_id e, se necessario, su client\_secret.
* **Vincoli tecnologici e integrazione:**
  + Utilizzo di HTTPS su tutti gli endpoint per proteggere le comunicazioni.
  + Implementazione di tecniche per mitigare attacchi CSRF e *man-in-the-middle*.
  + Conformità ai requisiti per la scalabilità, come il caching distribuito per i token e il supporto per infrastrutture containerizzate.
* **Sicurezza e monitoraggio**
  + Monitoraggio continuo delle metriche relative ai flussi OAuth, come tempi di risposta e tassi di errore.
  + Registrazione dettagliata delle richieste di autorizzazione per individuare tentativi di abuso o attacchi.

### Compatibilità 2FA

Per permettere un livello di sicurezza elevato, il sistema deve supportare differenti metodi di autenticazione a due fattori (2FA). In particolare, è essenziale che siano integrati i metodi più importanti come OAuth 2.0, fondamentale per gestire le autenticazioni in maniera sicura e flessibile.

I principali metoi di autenticazione a due fattori (2FA) sono:

* **OTP (One-Time Password):** Codici generati dinamicamente e inviati via SMS, e-mail oppure tramite app di autenticazione come Google Authenticator o Authy.
* **Notifiche push:** Un sistema pratico e veloce che consente agli utenti di confermare l’accesso tramite notifiche direttamente da un’app dedicata.
* **Token hardware:** Per chi preferisce soluzioni fisiche, sarà garantita la compatibilità con dispositivi come le chiavi di sicurezza YubiKey.

La caratteristica principale è quella di assicurare che il sistema sia pienamente compatibile con le principali piattaforme. Questo significa che dovrà funzionare senza problemi sia su desktop sia su dispositivi mobili. Allo stesso tempo è importante anche tenere conto di alcune limitazioni come l’integrazione con infrastrutture legacy o anche utilizzo di provider di terze parti per l’invio di OTP via SMS. Infine, il sistema deve essere progettato per gestire un numero elevato di richieste simultanee, mantenendo prestazioni ottimali. Al tempo stesso, deve risultare semplice e intuitivo per gli utenti, offrendo loro la possibilità di configurare metodi di autenticazione alternativi e di accedere a procedure di recupero in caso di problemi.

# Analisi delle Soluzioni Esistenti

## Identity Providers Affermati

### Panoramica dei principali IdP

Per la gestione delle identità e del controllo degli accessi si trovano nel Web varie alternative:

1. **Google Identity Platform** è una scelta ideale poiché tramite l’integrazione con altri servizi Google offre una gestione delle identità efficace e sicura, particolarmente vantaggiosa per chi già utilizza l’ecosistema Google. Inoltra Supporta OAuth 2.0.
2. **Microsoft Azure AD** da luglio 2023 fa parte della piattaforma Entra ID. È molto duttile poiché si integra con Microsoft 365 e con le altre applicazioni aziendali di Microsoft. Anche Microsoft Azure AD supporta OAuth 2.0 e consente l’implementazione di 2FA. Alcune funzionalità avanzate, però, sono disponibili solo nei piani premium.
3. **Okta** è una piattaforma molto semplice da utilizzare e molto apprezzata per le numerose integrazioni con terze parti. Inoltre, offre un portale utente personalizzabile e permette anche di configurare vari metodi di autenticazione multifattore (MFA). Anch’esso supporta OAuth 2.0. Tuttavia, seppur sia una soluzione facile da implementare, potrebbe richiedere delle personalizzazioni per soddisfare alcune esigenze aziendali più complesse e specifiche.
4. **Auth0**, oltre ad avere un'interfaccia semplice da utilizzare, è molto utile per le aziende in costante crescita che richiedono un sistema di autenticazione flessibile. Anche quest’ultima supporta OAuth 2.0 e permette di integrarci vari metodi di autenticazione multifattore (MFA), fornendo API robuste e ideali per ambienti di sviluppo dinamici sempre in sviluppo ed in crescita. Tuttavia, vi sono alcune segnalazioni per quanto riguarda la difficoltà nella configurazione avanzata e dei problemi di sincronizzazione.

### Funzionalità di Sicurezza e 2FA (da controllare, rileggere e modificare)

Ogni provider offre delle soluzioni di autenticazione a due fattori (2FA):

1. **Google Identity Platform**: tramite Google Authenticator, Google permette l’autenticazione a due fattori generando codici TOTP, SMS, e-mail o prompt push sui dispositivi Android collegati. Supporta anche l’autenticazione biometrica attraverso il sistema integrato sui dispositivi Android.
2. **Microsoft Azure AD**: permette l’autenticazione a due fattori tramite gli SMS, le chiamate vocali e l'utilizzo di Microsoft Authenticator per TOTP. Per i dispositivi abilitati, inoltre, è anche disponibile l’autenticazione biometrica tramite FIDO2 e WebAuthn. Infine, consente l'autenticazione adattiva, la quale applica metodi autenticazione a due fattori basati su posizione, comportamento, ecc.
3. **Okta**: permette l’autenticazione a due fattori tramite SMS ed Okta Verify (utilizzando codici TOTP e notifiche push). Inoltre, supporta metodi di autenticazione biometrica, come ad esempio l’autenticazione basata su WebAuthn per chiavi FIDO2. Infine, tramite l’utilizzo di Adaptive MFA, consente di personalizzare le politiche di accesso in base a vari fattori, come dispositivo, rischio, ecc.
4. **Auth0**: permette l’autenticazione a due fattori tramite SMS, e-mail, notifiche push e applicazioni TOTP come Google Authenticator. Inoltre, la piattaforma supporta anche WebAuthn per l’autenticazione biometrica, rendendola adatta soprattutto per le applicazioni scalabili e dinamiche.

## Confronto di Caratteristiche

### Usabilità e Scalabilità

Per quanto riguarda la capacità di gestione di utenti su larga scala e su opzioni di personalizzazione disponibili:

1. **Google Identity Platform**: adatto soprattutto per coloro che possiedono già un ecosistema Google. Per gestire i profili utenti si utilizza la Console di Google Cloud, la quale permette di personalizzarne i criteri di sicurezza. Le opzioni di personalizzazione potrebbero, però, essere limitate, qualora si ricercassero delle esigenze molto specifiche, poiché sono ottimizzate per scenari standardizzati.
2. **Microsoft Azure AD**: adatto per la gestione utenti su larga scala, grazie a numerose funzionalità avanzate. Le opzioni di personalizzazione sono numerose, però per alcune più avanzate potrebbero essere necessarie delle licenze Premium
3. **Okta**: ottimo per quanto riguarda la gestione di grandi quantità di utenti. La sua flessibilità e semplicità è dovuta al suo Universal Directory che permette di unificare le politiche di sicurezza con i criteri di accesso. Inoltre, è ideale per scenari multi-cloud grazie alla grande facilità d’integrazione con applicazioni di terze parti. Alcune personalizzazioni, però, hanno bisogno di molto tempo e supporto tecnico.
4. **Auth0**: grazie alla sua flessibilità e scalabilità permette di gestire utenti su larga scala. Inoltre, è particolarmente adatta per le aziende in crescita o per gli ambienti dinamici. È molto utilizzata per la sua integrazione personalizzabile, che permette agli sviluppatori di adattare l'autenticazione alle esigenze di ogni applicazione. Tuttavia, per delle configurazioni avanzate ed estremamente specifiche c’è bisogno di competenze tecniche elevate per una gestione ottimale.

### Flessibilità e Integrazione

Per quanto riguarda la capacità di integrazione con sistemi preesistenti e di supporto dell’autenticazione federata, i provider esistenti:

1. **Google Identity Platform**:offre delle API flessibili per integrare la gestione delle identità con applicazioni e sistemi esistenti. Supporta perfettamente il protocollo OAuth2.0 poiché compatibile con molte piattaforme moderne, facilitandone la connessione con le app da telefono e con gli accessi web da computer. La GIP pecca di flessibilità per quanto riguarda i sistemi legacy (poiché i software e le infrastrutture sono basati su tecnologie vecchie) e per scenari di tipo aziendale. Supporta provider federati come Google e Facebook, offrendo come funzionalità nativa il social login, ovvero un accesso rapido che evita la creazione di account estemporanei o di dover ricordare password tra gli infiniti account personali.
2. **Microsoft Azure AD**:noto per la sua capacità di integrarsi con spazi operativi come Microsoft e Windows, supporta una vasta gamma di protocolli standard, tra cui ovviamente OAuth2.0, che gli permettono di integrare applicazioni moderne e soprattutto quelle datate. Consente connessioni personalizzate sia per ambienti aziendali complessi sia per singoli consumatori. Offre funzionalità di trust reciproco con aziende, e ciò consente a due o più utenti di riconoscere reciprocamente le identità degli utenti autenticati nei rispettivi ambienti. Inoltre, gestisce gli ambienti ibridi, ovvero infrastrutture che hanno accesso completo ai data server dell’azienda, dove risiedono la maggior parte delle informazioni di un’azienda, mentre una restante parte viene archiviata nei cloud.
3. **Okta**:ottimo per la sua capacità di integrare rapidamente applicazioni cloud e locali grazie alla vasta libreria di connettori predefiniti. Questi connettori collegano applicazioni a Okta per centralizzare gli accessi tramite login o single sign-on. Inoltre, ha il compito di creare e rimuovere account in modo automatico nelle applicazioni collegate in base alle modifiche di una directory predefinita. Okta supporta la federazione bidirezionale, ovvero può collaborare con altri sistemi di gestione delle identità per condividere e accettare autenticazioni. Integra il login tramite provider come Google, Facebook e tanto altro, utile per consumer-oriented e scenari aziendali più complessi.
4. **Auth0**: estremamente flessibile, offre molteplici opzioni di integrazione grazie a rule engines e hooks, strumenti avanzati che consentono di personalizzare e controllare i flussi di autenticazione e altre operazioni in modo dinamico e specifico. Questi script (“Rules”) vengono eseguiti dopo l’autenticazione dell’utente ma prima che il token venga assegnato. Gli hooks sono punti di estensione che consentono di eseguire codice personalizzato in momenti precisi del ciclo di vita di un utente. A differenza delle rules, gli Hooks sono progettati per eventi più ampi e non sono legati all’autenticazione. Auth0 inoltre è progettato per ambienti altamente eterogenei e globali; infatti, è estremamente capace di adattarsi a contesti con esigenze totalmente diversi.

## Differenziazione del Progetto

*Definizione delle aree di miglioramento o personalizzazione che differenziano il nostro IdP dai provider esistenti, come sicurezza avanzata, maggiore configurabilità, o controllo centralizzato.*

# Pagine web

1 – Scheda home

La scheda iniziale contiene un **form** con i campi delle credenziali username e password e bottoni per la conferma delle credenziali, la registrazione utente ed il recupero credenziali.

In oltre ci dovrebbe essere altre modalità d’accesso con Google identity platform, Azure AD, Apple?

In caso di **accesso negato**, comparsa di un messaggio d’errore ed azzeramento del parametro password

* 1. – Credenziali corrette

Reindirizzamento alla scheda 2FA in cui processare OTP

* 1. – Registrazione utente

Reindirizzamento alla scheda Registrazione utente in cui ci sarà un form da compilare con i dati utili alla registrazione: nome, cognome, email, password, conferma password con bottoni di conferma, annullamento e navigazione generale

* 1. – Recupero credenziali

Reindirizzamento alla scheda Recupero credenziali in cui ci sarà un sistema di recupero pass.

Il metodo Agile è un approccio alla gestione dei progetti che privilegia la flessibilità, il lavoro iterativo e il coinvolgimento continuo degli stakeholder. È particolarmente utile per progetti complessi e dinamici, dove i requisiti possono evolvere durante lo sviluppo. È il più adatto per un progetto riguardante l’autenticazione a due fattori (2FA) poiché coinvolge:

* **Componenti complessi**: implementazione di protocolli di sicurezza e integrazione con sistemi esistenti;
* **Adattabilità**: è possibile che emergano nuove esigenze o vulnerabilità durante lo sviluppo;
* **Validazione continua**: una serie di test e miglioramenti iterativi per garantire la robustezza e l’usabilità

Questo metodo è il più affine al nostro progetto poiché:

* Consente di sviluppare e **testare rapidamente ogni componente** (ad esempio la generazione del codice OTP, l’invio tramite SMS o l’autenticazione biometrica)
* Permette di **raccogliere feedback** dagli utenti e migliorare l’esperienza individuale

1. Le normative di sicurezza rilevanti per un progetto di Identity Provider con OAuth 2.0 e autenticazione a due fattori (2FA) possono variare a seconda della regione e del settore, le più comuni e riconosciute includono:

   **GDPR (General Data Protection Regulation)** - **Regolamento (UE) 2016/679**: Regolamento europeo che stabilisce standard rigorosi per la protezione dei dati personali.

   **ISO/IEC 27001**: Standard internazionale per la gestione della sicurezza delle informazioni, che richiede un **Sistema di Gestione della Sicurezza delle Informazioni (ISMS)** ben strutturato e basato sui rischi.

   **NIST 800-63B (Digital Identity Guidelines)**: Linee guida del National Institute of Standards and Technology per la gestione delle identità digitali, particolarmente utili per implementare protocolli di autenticazione sicura.

   **CIS Controls (Center for Internet Security Controls)**: Una serie di best practice e controlli essenziali per migliorare la sicurezza delle informazioni.

   **PCI-DSS (Payment Card Industry Data Security Standard)** (per sistemi che trattano dati di carte di pagamento): Se il progetto IdP viene utilizzato in ambiti che richiedono transazioni finanziarie o accesso a dati di pagamento. [↑](#endnote-ref-2)
2. **Gli attacchi di tipo man-in-the-middle** sono un tipo di attacco informatico in cui un'entità malintenzionata si inserisce segretamente nella comunicazione tra due parti (es. un utente e un server), intercettando, modificando o rubando i dati trasmessi senza che le vittime se ne accorgano. [↑](#endnote-ref-3)
3. **Architettura Zero Trust** è un approccio alla sicurezza informatica basato sul principio del **"non fidarti mai, verifica sempre"**. Invece di presumere che tutto ciò che ci sia all'interno della rete aziendale sia sicuro, il protocollo Zero Trust richiede una verifica continua di tutti gli utenti, dispositivi e applicazioni, sia all'interno che all'esterno della rete aziendale. [↑](#endnote-ref-4)
4. **Fornitore di identità di terze parti:** sono entità o servizi che gestiscono e verificano l'identità degli utenti e forniscono funzionalità di autenticazione per conto di altre applicazioni o servizi. Alcuni esempi di fornitori sono Google, Microsoft, Apple, Facebook ecc. [↑](#endnote-ref-5)