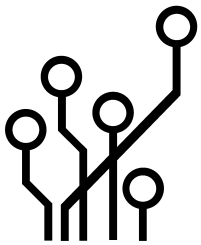

PROJECT ORIGIN

projectorigin2023@gmail.com

Specifica Tecnica

Versione	0.1.1
Responsabile	
Redattori	Corbu Teodor Mihail
Verificatori	0.1.1
Uso	Esterno
Destinatari	*ProjectOrigin* Prof. Vardanega Tullio Prof. Cardin Riccardo

Descrizione
Nel presente documento si fornisce una visione approfondita dell'architettura, del design e delle specifiche tecniche del progetto *Personal Identity Wallet*

Registro delle modifiche

Vers.	Data	Autore	Ruolo	Descrizione
0.1.1	2023-09-18	Corbu Teodor	Analista	Aggiunte parti Introduzione (capitolo Architettura), Data Scrapper, React Material UI
0.1.0	2023-08-25	Andreetto Alessio	Verificatore	Verifica$_g$del documento
0.0.1	2023-08-24	Corbu Teodor	Analista	Creazione struttura documento, aggiunta Introduzione e spiegazione dei database

Indice

1 Introduzione **3**
1.1 Scopo del documento . . . 3
1.2 Scopo del prodotto . . . 3
1.3 Note Esplicative . . . 3
1.4 Riferimenti . . . 3

2 Architettura **5**
2.1 Introduzione . . . 5
2.1.1 Container Front-end . . . 5
2.1.2 Container Back-end . . . 5
2.1.3 Container database . . . 5
2.1.4 Conteiner WaltId per strandard openID . . . 5
2.1.5 Container di supporto . . . 5
2.2 Componenti Back-end . . . 6
2.2.1 OriginiIssuerApi . . . 6
2.2.2 OriginWalletApi . . . 6
2.3 Componenti Front-end . . . 6
2.3.1 originIssuer . . . 6
2.3.2 originWallet . . . 6
2.3.3 originVerifier . . . 6
2.4 Componenti . . . 6
2.5 Diagramma delle classi . . . 6
2.6 Design pattern . . . 6

Elenco delle figure

Elenco delle tabelle

1 Introduzione

1.1 Scopo del documento

La Specifica Tecnica si pone come obiettivo di descrivere in modo esaustivo l'organizzazione della struttura del software, delle tecnologie adottate e delle scelte architetturali compiute dal gruppo durante le fasi di progettazione e di codifica del prodotto.
All'interno del documento si possono trovare gli schemi delle classi per delineare l'architettura e le funzionalità chiave del prodotto, con l'obiettivo di fornire una comprensione completa e chiara del sistema e delle interazioni interne.
Il documento contiene anche una sezione per i requisiti che vengono soddisfatti dal prodotto; questo permette al gruppo di valutare il progresso del lavoro e di tener traccia degli obiettivi imposti.

1.2 Scopo del prodotto

Lo scopo del prodotto è quello di creare una versione semplificata di un applicativo per implementare e rilasciare un "portafoglio di identità digitale" conforme a un insieme di standard, in modo che possa essere utilizzato con qualsiasi servizio, che adotti tale struttura, in qualsiasi paese dell'UE.
In particolare, si dovrà realizzare una web app$_g$ avendo queste componenti architetturali:

- Un componente back-office per consentire al dipendente dell'organizzazione emittente di verificare$_g$ manualmente la richiesta di credenziali e autorizzarne l'emissione;
- Un componente di interazione con l'utente dimostrativo per consentire all'utente (titolare) di navigare e richiedere specifiche credenziali da un emittente (ad esempio, il sito di una demo universitaria);
- Un componente di interazione con l'utente dimostrativo per consentire all'utente (titolare) di navigare un sito verificatore$_g$ e fornire le credenziali richieste;
- Un'app front-end per l'utente per archiviare e gestire le proprie credenziali;
- Un componente di comunicazione per consentire lo scambio di credenziali/presentazioni secondo un protocollo standard - il componente di comunicazione sarà implementato tre volte nei tre contesti (lato emittente, lato titolare, lato verificatore).

1.3 Note Esplicative

Alcuni termini utilizzati nel documento possono avere significati ambigui a seconda del contesto. Al fine di evitare equivoci, è stato creato un *Glossario v.1.0.0* contenente tali termini e il loro significato specifico. Per segnalare che un termine è presente nel *Glossario v.1.0.0* , sarà aggiunta una "g" a pedice accanto al termine corrispondente nel testo.

1.4 Riferimenti

1. Normativi:

- ***Norme di Progetto v.1.0.0*** : contengono le norme e gli strumenti per gli analisti;
- **Capitolato d'appalto C3**: https://www.math.unipd.it/~tullio/IS-1/2022/Progetto/C3.pdf;
- **Regolamento del progetto didattico:**: https://www.math.unipd.it/~tullio/IS-1/2022/Dispense/PD02.pdf.

2. Informativi:

- **Analisi dei Requisiti v1.0.0**;

- **Qualità di prodotto – slide T8 di Ingegneria del Software:** : https://www.math.unipd.it/~tullio/IS-1/2022/Dispense/T08.pdf;
- **Qualità di processo – slide T9 di Ingegneria del Software:** : https://www.math.unipd.it/~tullio/IS-1/2022/Dispense/T09.pdf;
- **Verifica e Validazione: introduzione – slide T10 di Ingeneria del Software:**: https://www.math.unipd.it/~tullio/IS-1/2022/Dispense/T10.pdf;
- **Verifica e Validazione: introduzione – slide T11 di Ingeneria del Software:**: https://www.math.unipd.it/~tullio/IS-1/2022/Dispense/T11.pdf;
- **Verifica e Validazione: introduzione – slide T12 di Ingeneria del Software:**: https://www.math.unipd.it/~tullio/IS-1/2022/Dispense/T12.pdf.

2 Architettura

2.1 Introduzione

Per la Realizzazione delle 3 webapp è stata adottata un architettura a microservizi, separando le funzioni di back-end da quelle del front-end. I vari microservizi di ciascuna webapp sono stati sviluppati su container docker differenti.

2.1.1 Container Front-end

- **originIssuer:** È la componete di front-end della webapp Issuer. Essa si interfaccia con l'utente per inviare segnali e ricevere dati dal Back-end *originIssuerApi*. Per la realizzazione del codice è stato adottato il pattern architetturale MVVM (Model-View-ViewModel).
- **originWallet:** È la componete di front-end della webapp Wallet. Essa si interfaccia con l'utente per inviare segnali e ricevere dati dal Back-end *originWalletApi*. Per la realizzazione del codice è stato adottato il pattern architetturale MVVM (Model-View-ViewModel).
- **originVerifier:**È la componete di front-end della webapp Verifier. Essa si interfaccia con l'utente per inviare segnali e ricevere dati dal Back-end *originVerfierApi*. Per la realizzazione del codice è stato adottato il pattern architetturale MVVM (Model-View-ViewModel).

2.1.2 Container Back-end

- **originIssuerApi:** È la componente di back-end della webapp Issuer che si occupa di gestire le richieste provenienti dal front-end e di comunicare con il database *originIssuerDB* per la memorizzazione dei dati.
- **originWalletApi:** È la componente di back-end della webapp Wallet che si occupa di gestire le richieste provenienti dal front-end e di comunicare con il database *originWalletDB* per la memorizzazione dei dati.

2.1.3 Container database

- **originIssuerDB:** È la componente della webapp Issuer che va a eseguire operazioni sul database per la richiesta e la memorizzazione di dati.
- **originWalletDB:** È la componente della webapp Wallet che va a eseguire operazioni sul database per la richiesta e la memorizzazione di dati.

2.1.4 Conteiner WaltId per strandard openID

- **openIdIssuer:** È una componente della libreria WaltID per mantenere lo standard openId di comunicazione tra Issuer e le altre webapp, al fine di rispettare le richieste del capitolato.
- **openIdWallet:** È una componente della libreria WaltID per mantenere lo standard openId di comunicazione tra Wallet e le altre webapp, al fine di rispettare le richieste del capitolato.
- **openIdVerifier:**È una componente della libreria WaltID per mantenere lo standard openId di comunicazione tra Verifier e le altre webapp, al fine di rispettare le richieste del capitolato.

2.1.5 Container di supporto

- **adminer:** È un container che permette agli sviluppatori di gestire il database tramite interfaccia web.
- **nginx:** Lo utilizziamo come server proxy per gestire il reindirezzamento del traffico http tramite domini verso i container interni della rete docker

2.2 Componenti Back-end

2.2.1 OriginiIssuerApi

2.2.2 OriginWalletApi

2.3 Componenti Front-end

2.3.1 originIssuer

2.3.2 originWallet

2.3.3 originVerifier

2.4 Componenti

2.5 Diagramma delle classi

2.6 Design pattern