

# Libra FaceWallet

Realizzazione di un Web wallet per la criptovaluta Libra

**Progetto del corso di Sistemi Peer to Peer**

**Luca D'Ambrosio – [luca.dambrosio3@studio.unibo.it](mailto:luca.dambrosio3@studio.unibo.it)**

**Marco Silvestri – [marco.silvestri10@studio.unibo.it](mailto:marco.silvestri10@studio.unibo.it)**

# Indice

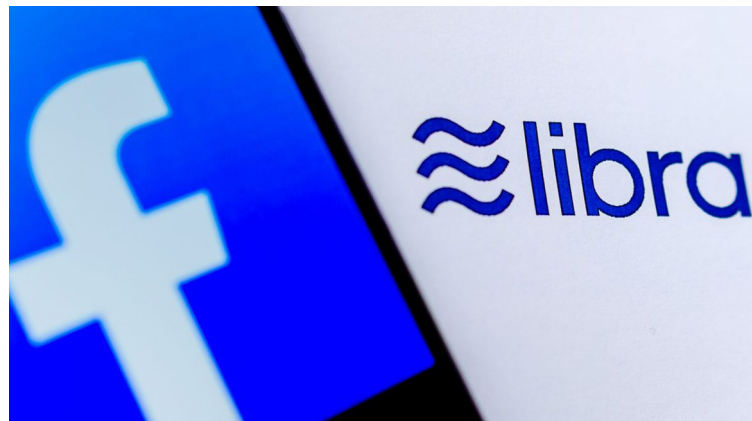
- Obiettivo del progetto
  - Implementazione del progetto
    - MoveOnLibra API
    - Operazioni implementate
  - Wallet Libra
  - Problemi riscontrati
  - Sviluppi futuri
-

# Obiettivo

Obiettivo del progetto è stato quello di creare una piattaforma web che ospiti un Wallet per la criptovaluta Libra.

Operazioni aggiuntive:

- possibilità di creare un portafoglio associandolo ad un profilo Facebook,
- condivisione del proprio wallet su Facebook.



# Implementazione

Il progetto è stato realizzato attraverso l'utilizzo delle API fornite da **MoveOnLibra (MOL)**.

Le operazioni implementate nel nostro wallet sono:

- Creazione di un nuovo wallet
  - Creazione di un nuovo wallet associato ad un profilo Facebook
- Login in un wallet esistente
- Mint
- Transfer
  - Transfer by camera
- Backup chiave mnemonica

L' API [MoveOnLibra / mol](#) viene utilizzata per sviluppare applicazioni basate su Libra e Smart Contract, con l'obiettivo di semplificare la vita come sviluppatore della blockchain Libra.

L' API è stata creata utilizzando endpoint RESTful e HTTP standard.

In particolare:

- I codici di stato HTTP vengono utilizzati per indicare lo stato della chiamata API.
- Il codice JSON viene restituito su tutte le risposte API, inclusi gli errori, con una struttura coerente per tutti i messaggi.
- I campi di testo supportano UTF-8.

Le operazioni messe a disposizione dalle API MoveOnLibra si dividono in diverse categorie in base alla loro esigenza, esse sono:

- **Address:** restituzione di informazioni relative ad un dato address. (Public)
- **Event:** restituisce informazioni di eventi generati da un dato address (Public).
- **Transaction:**
  - a. GET: restituiscono informazioni relativamente alle transazioni del sistema. (Public)
  - b. POST: operazioni che permettono di effettuare transazioni nel sistema. (Private)
- **Wallet:** operazioni che restituiscono informazioni su un dato wallet e permettono la creazione di un nuovo portafoglio.
- **Move:** permettono l'implementazione di Smart Contract (al momento non disponibili).

L'accesso alle operazioni di tipo Private presuppone una registrazione alla pagina MOL con lo scopo di ottenere un **appkey**.

Una volta ottenuta la propria appkey è necessario creare un'istanza di un oggetto LibraClient, fornendo due parametri: **network** e **appkey**. Attualmente, il parametro network ha un solo valore valido: **testnet**.

```
var client = new LibraClient ( " testnet " , appkey);
```

# Struttura del progetto

Il progetto è caratterizzato da 3 file:


- **index.php**: contiene il codice Javascript per la gestione delle operazioni del wallet e il codice html per la grafica,
- **server.php**: contiene il codice per la gestione di Login e Registrazione,
- **storeUser.json**: contiene i dati relativi agli utenti che creano un nuovo wallet nella piattaforma.



# Libra FaceWallet



Libra Wallet Logout


  
Lucas

Address d727df70...e1b0daa0

Balance 0

Id utente 550

QRCode





Copy to clipboard


Show account detail


Condividi

Activity

Mint   
Load Libra \$

Transfer   
Transfer Libra

Backup   
Backup wallet

Send by camera   
Transfer by Scan Camera

*Operazioni wallet*

*Profilo utente*

# Operazioni

- Creazione wallet

L'operazione **CREATE WALLET** è formata da due step.

Una volta che l'utente ha inserito il nome del portafoglio e una password e schiaccia il bottone "Create wallet" questo ci permette di creare un nuovo wallet nella blockchain e di salvare i dati associati al profilo utente all'interno di un file json locale.

In particolare la prima operazione è resa possibile dalla chiamata:

```
wallet = await client.walletAPI.createWallet(name)
account = await client.walletAPI.createWalletAccount(wallet)
```

Nel primo caso viene creato un wallet nella blockchain, passando come input il suo nome. La seconda chiamata associa un account al wallet prima creato.

A screenshot of a mobile application interface titled "Create new wallet" with a close button (X) in the top right corner. Below the title, there are two tabs: "Register" (selected) and "Login". A message states: "If is your first time to use this libra wallet app. You need to create a wallet of yourself first:". Below this, there is a red rectangular box highlighting the registration form. Inside the box, there is a text input field labeled "Wallet name" with a wallet icon on the right, a password input field labeled "Password" with a lock icon on the right, and a blue button labeled "Create wallet on Libra". Below the red box, there is a separator "- OR -". Below the separator, there is another password input field labeled "Password" with a lock icon on the right, and a blue button labeled "Sign in using Facebook" with a Facebook icon on the left.

# Operazioni



- Creazione nuovo wallet associato ad un profilo Facebook

L'operazione di creazione di un nuovo wallet associato ad un profilo facebook avviene nella stessa maniera descritta nella precedente slide con l'aggiunta di una funzione JavaScript che permette di effettuare la creazione di un wallet attraverso Facebook.

Per poter realizzare questa operazione è necessario creare un'applicazione nel dashboard delle API Facebook e aggiungere all'interno dell'app la funzionalità *FacebookLogin*.

I dati restituiti dalla chiamata vengono così utilizzati:

- Nome wallet = nome profilo/utente
- Immagine wallet = immagine profilo

# Operazioni



## • Creazione wallet

La risposta alla chiamata ***createWallet*** è struttura JSON del wallet che contiene le seguenti informazioni:

### WALLET:

- wallet\_id
- nome wallet
- child\_count
- appid
- rete
- created\_at

La risposta alla chiamata ***createWalletAccount*** è struttura JSON dell'account che è così composto:

### ACCOUNT:

- address
- appid
- wallet\_id
- child\_id
- rete
- created\_at

# Operazioni

- Creazione wallet

Se si possiede già un portafoglio, quest'ultimo insieme ad altre strutture legate all'utente, vengono archiviate nel localStorage browser, in modo tale che ad ogni caricamento di pagina venga ricaricato dal localStorage corrente e non venga richiesto il Login/Registrazione.

A screenshot of a mobile application interface titled "Create new wallet" with a close button (X) in the top right corner. The interface has two tabs: "Register" (selected) and "Login". Below the tabs, a message states: "If is your first time to use this libra wallet app. You need to create a wallet of yourself first:". There are two input fields: "Wallet name" with a wallet icon on the right, and "Password" with a lock icon on the right. Below these is a blue button labeled "Create wallet on Libra". A separator "- OR -" is centered below the button. There is another "Password" input field with a lock icon, followed by a blue button labeled "Sign in using Facebook" with a Facebook icon on the left.

# Operazioni

## • Login



L'operazione **LOGIN** permette di entrare in un portafoglio che è già presente nella blockchain.

Questa operazione avviene tramite 2 passaggi.

1. Inserire la chiave mnemonica e la password collegata al wallet. Questi dati vengono inviati ad un server locale che effettua le sue verifiche. Se otteniamo un esito positivo ci viene restituito l'id associato al portafoglio utente.
2. Tale "wallet\_id" viene usato dalle API per recuperare il wallet. Le chiamate effettuate sono:

```
wallet = await client.walletAPI.getWallet(id)
account = await client.walletAPI.getWalletAccounts(wallet)
```

A screenshot of a mobile application interface showing a 'Create new wallet' dialog box. The dialog has a title bar with a close button (X). Below the title bar, there are two tabs: 'Register' (with a person icon) and 'Login' (with a person and plus icon). The 'Login' tab is selected. The main content area says 'Enter your data:'. There are two input fields: 'Mnemonic key' with a key icon on the right, and 'Password' with a lock icon on the right. At the bottom, there is a large blue button labeled 'Login'.

# Operazioni

- Logout

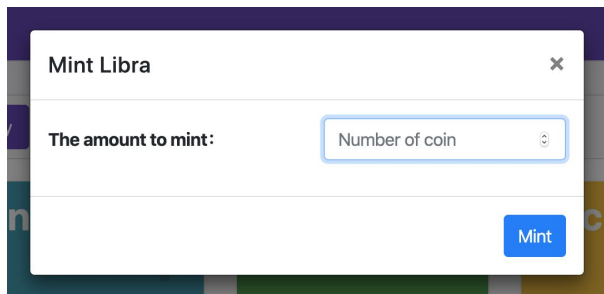
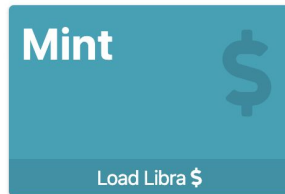
L'operazione **LOGOUT** permette di uscire dal wallet. Interrogando il server, vengono azzerati i dati di sessione, svuotando le credenziali memorizzate e vengono cancellati i dati presenti nel localStorage del Browser.

A rectangular button with a solid purple background. The word 'Logout' is written in a white, sans-serif font, followed by a white right-pointing arrow icon.

Logout ➔

# Operazioni

- Mint



L'operazione **MINT** permette di aggiungere Libra coin all'account.

In particolare l'operazione dell'API funziona in questo modo:

```
const appkey = "*****" // your app key
var client = new LibraClient("testnet", appkey);
try{
    data = await client.TransactionAPI.mint(receiver_account_address, number_of_micro_libra,
    console.log(data) //SignedTransaction
} catch (error) {
    console.log(error);
}
```

INPUT 1: Address receiver

INPUT 2: Numero di libra da caricare.



# Operazioni

## • Transfer

L'operazione **TRANSFER** permette di inviare Libra da un sender account verso un receiver account. La chiamata alle API che ci permette di realizzare questa operazione è:

```
data = await client.TransactionAPI.p2pTransfer
```

Essa prende in input:

- **wallet\_id**: id wallet loggato
- **sender\_account\_address**: address del sender
- **receiver\_account\_address**: address del receiver
- **number\_of\_micro\_libra**: libra da trasferire

A white dialog box titled 'Transfer Coin' with a close button (X) in the top right corner. It contains two input fields: 'Receiver Address in hex64 format' and 'Number of coin'. A blue 'Transfer' button is located at the bottom right of the dialog.

# Operazioni

- Download chiave mnemonica



La chiave mnemonica, come visto nelle slide precedenti, è utile durante la fase di login.

Al fine di ottenere la propria chiave il bottone Backup wallet permette di scaricare un file di testo "*NomeWallet.mnemonic*" contenente la chiave mnemonica associata al wallet. La chiave ha la seguente forma:

```
release deer wolf bottom able lazy anger damp frame shiver feed rebuild wet crucial april age pulp key;1
```

Tale operazione viene resa possibile attraverso la seguente chiamata all'API:

```
data = await client.WalletAPI.backupWallet(wallet_id);
```

INPUT 1: id wallet loggato

# Operazioni

- **Transfer by camera**

L'operazione **TRANSFER BY CAMERA** è una funzione aggiunta dell'operazione **TRANSFER**, infatti, permette di effettuare un passaggio di Libra tramite l'ausilio della Webcam.

In particolare, tramite una funzione Vue viene decodificato il qrCode del receiver e di conseguenza viene aperta la “modal” relativa a “transfer” per effettuare il passaggio di Libra da un account ad un altro.

Send by  
camera



Transfer by Scan Camera



Cancel



# Operazioni

- DEMO

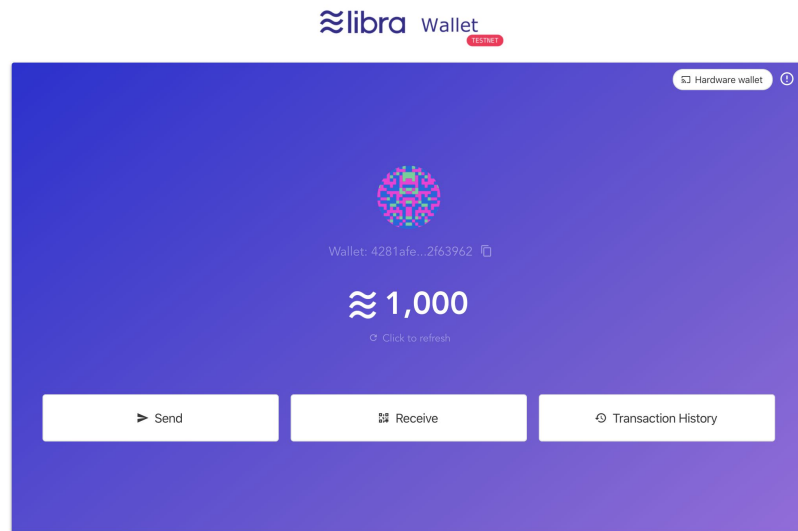


Dopo la spiegazione delle operazioni siamo pronti ad eseguire una demo del progetto.



# Wallet Libra

1. Il wallet ufficiale che dovrebbe supportare la criptovaluta Libra è **Calibra**, non ancora rilasciato da Facebook.
2. Un'altro portafoglio Kulap Libra ed è presente al seguente indirizzo <https://dev.kulap.io/libra/#/> esso implementa le principali operazioni di un wallet.



# Problemi



## 1. Utilizzo API Facebook

- *"F.R.I.E.N.D.S":*

non è stato possibile implementare la lista degli amici che possiedono un portafoglio in quanto le API di Facebook non permettono di ottenere informazioni sugli amici di un utente (in questo caso l'utente loggato). Tali informazioni non sono più disponibili dal 2018 e possono essere reperibili solo nel caso in cui, dopo accurati controlli, l'app venisse pubblicata.

- *"Condividi su Facebook":*

questa funzione seppur presente nel wallet non permette di condividere su Facebook il nostro wallet per le motivazioni date precedentemente, in particolare, in questo caso non disponiamo di un URL valido per poterlo condividere su Facebook.

## 2. API

Le API hanno smesso di funzionare per alcuni giorni a causa di un aggiornamento della Testnet Libra.

# Sviluppi futuri

Abbiamo pensato anche a dei possibili sviluppi futuri per la nostra piattaforma.

1. Migliorare funzionalità Facebook (friends, share),
2. Aggiungere grafici sull'andamento del portafoglio.

