**DOCUMENTAZIONE PROGETTO ESAME TECNOLOGIE INTERNET:**

Sommario

[Specifiche progetto: 3](#_Toc66919819)

[Specifiche Tecniche: 4](#_Toc66919820)

[Descrizione tecnologie: 5](#_Toc66919821)

[Riferimenti: 6](#_Toc66919822)

[Progettazione sistema: 7](#_Toc66919823)

[Header.php - Cliente 8](#_Toc66919824)

[Logout.inc.php: 8](#_Toc66919825)

[Header.php - Impiegato: 8](#_Toc66919826)

[Logoutemp.inc.php: 8](#_Toc66919827)

[Footer.php 9](#_Toc66919828)

[Index.php - Cliente: 10](#_Toc66919829)

[loadCardProducts.js: 10](#_Toc66919830)

[Index.php - Impiegato: 10](#_Toc66919831)

[Loginemp.inc.php: 10](#_Toc66919832)

[Signup.php: 11](#_Toc66919833)

[Signup.inc.php: 11](#_Toc66919834)

[Login.php: 12](#_Toc66919835)

[Login.inc.php: 12](#_Toc66919836)

[Chisiamo.php: 13](#_Toc66919837)

[Cart.php: 14](#_Toc66919838)

[Cart.js: 14](#_Toc66919839)

[Checkout.php: 15](#_Toc66919840)

[checkoutCreditCard.js 15](#_Toc66919841)

[checkoutETH.js 15](#_Toc66919842)

[metamask.js 15](#_Toc66919843)

[Search.php: 16](#_Toc66919844)

[SearchBook.js 16](#_Toc66919845)

[Profile.php: 17](#_Toc66919846)

[Profile.js: 17](#_Toc66919847)

[ProfileETH.js: 17](#_Toc66919848)

[Editdetails.inc.php: 17](#_Toc66919849)

[Dashboard.php: 18](#_Toc66919850)

[Dashboard.js: 18](#_Toc66919851)

[Funzioni incluse dagli altri files: 19](#_Toc66919852)

[Dbh.inc.php: 19](#_Toc66919853)

[getSmartContractInfo.js: 19](#_Toc66919854)

[getSmartContractInfo.inc.php: 19](#_Toc66919855)

[isLogged.inc.php: 19](#_Toc66919856)

[getAddressETH.inc.php: 19](#_Toc66919857)

[buyNewBook.inc.php: 19](#_Toc66919858)

[Function.inc.php: 20](#_Toc66919859)

[- emptyInputSignup: 20](#_Toc66919860)

[- invalidEmail: 20](#_Toc66919861)

[- emailExists: 20](#_Toc66919862)

[- passwordMatch: 20](#_Toc66919863)

[- invalidPassword: 20](#_Toc66919864)

[- createUser: 20](#_Toc66919865)

[- emptyInputLogin: 20](#_Toc66919866)

[- loginUser: 20](#_Toc66919867)

[- emailDetailsExists 20](#_Toc66919868)

[- editUserDetails 20](#_Toc66919869)

[- loginEmp 20](#_Toc66919870)

[- viewFormat 20](#_Toc66919871)

[- viewLanguage 20](#_Toc66919872)

[- buyNewBook 20](#_Toc66919873)

[- viewAddressETH 20](#_Toc66919874)

[PassShow.js 21](#_Toc66919875)

[chart.js 21](#_Toc66919876)

[routes.php 22](#_Toc66919877)

[Use Case Test: 23](#_Toc66919878)

# Specifiche progetto:

Creare una Decentralized application (DApp) per un sistema di vendita di prodotti online:

- I prodotti online in questione sono: Libri.

- Il sistema deve essere accessibile agli utenti tramite un sito web dinamico.

* Home del sito dove si possono:
  + Visualizzare i prodotti.
  + Acquistare prodotti. Metodi di pagamento:
    - Carta di Credito
    - Ether (Blockchain Ethereum)
  + Effettuare ricerche sui prodotti per:
    - Titolo.
    - Autore.
    - Editore.
    - Anno di pubblicazione.
    - Nome del genere letterario.
* Pagina di registrazione.
* Pagina di login.
* Pagina profilo dove si possono:
  + Visualizzare lo storico degli ordini.
  + Aggiornare i dettagli del proprio account.
  + Collegarsi con la blockchain Ethereum per:
    - Visualizzare il proprio bilancio (ETH)
    - Visualizzare i propri ordini effettuati con lo Smart Contract
* Pagine per effettuare l’acquisto di uno o più prodotti.

- Il sistema deve essere accessibile agli impiegati tramite un sito web dinamico.

* Pagina di login.
* Dashboard usata dagli impiegati per:
  + Aggiornare il numero di copie disponibili.
  + Acquisire i prodotti mancanti. I prodotti non devono essere duplicati. Si devono al massimo aggiornare il numero di copie esistenti. Quindi se si deve acquisire un prodotto, questo non deve essere già presente nel database.
  + Spedire gli ordini.
  + Visualizzare le statistiche delle vendite per ogni prodotto.
  + Visualizzare il bilancio del negozio (ETH).
  + Visualizzare il bilancio dello Smart Contract (ETH).
  + Ritirare gli Ether dallo Smart Contract (deposito del conto del proprietario del negozio).
  + Visualizzare gli ordini effettuati tramite lo Smart Contract.
  + Spedire gli ordini effettuati tramite lo Smart Contract.

# Specifiche Tecniche:

* e-Commerce con interfaccia web dinamica realizzato con (pattern MVC):
  + Model:
    - JSON
  + View:
    - Framework Bootstrap
    - HTML
    - CSS
  + Controller:
    - PHP
    - Slim Framework
    - JavaScript
    - JQuery
    - DataTables
    - JQuery Pagination plugin
    - Web3.js
    - MetaMask
  + MySQL
  + Apache Web Server

* Smart Contract Ethereum utilizzato per effettuare il pagamento di prodotti e per tenere traccia di chi compra cosa.

Realizzato con:

* + Truffle
  + Ganache
  + Solidity
  + OpenZeppelin (standard per la sicurezza)
* Editor di testo:
  + Visual Studio Code

## Descrizione tecnologie:

Nella seguente sessione si motivano le ragioni che hanno portato all’adozione delle seguenti tecnologie.

In generale le scelte sono state dettate tali tecnologie sono degli standard.

Ambler’s Standards Law: *“The more commonly accepted a standard the easier it is for team*

*members to communicate”*

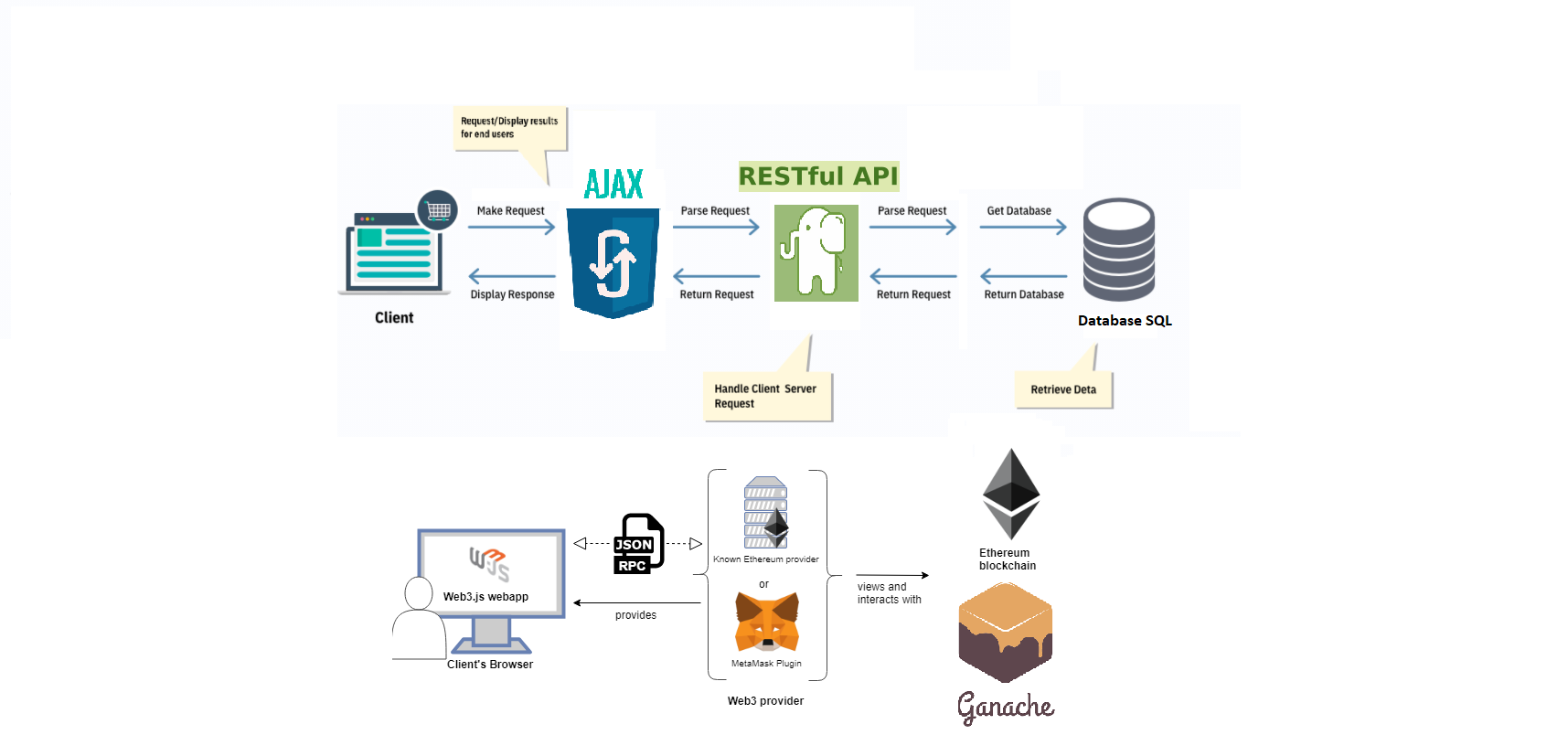
* **JSON**: “*formato adatto all'interscambio di dati fra applicazioni client/serve*r” – JSON.org. È stato scelto perché si possono rappresentare i dati in maniera più compatta ed efficiente in termini di spreco di banda e memoria e lavora bene con JavaScript essendo semplice la conversione in oggetti JavaScript. Inoltre è lo standard utilizzato per realizzare il Model delle web application.
* **Framework Bootstrap**: “*raccolta di strumenti liberi per la creazione di siti e applicazioni per il Web. Contiene modelli di progettazione basati su HTML e CSS, sia per la tipografia, che per le varie componenti dell'interfaccia, come*[*moduli*](https://it.wikipedia.org/wiki/Form)*, pulsanti e navigazione, così come alcune estensioni opzionali di JavaScript*” – Wikipedia. È stato scelto poiché uniforma i componenti delle interfacce web evitando elevati oneri di manutenzione.
* **HTML**: *“linguaggio di markup. Nato per la formattazione e impaginazione di documenti ipertestuali”* – Wikipedia. È stato scelto poiché è lo standard utilizzato per la definizione della struttura di una web application.
* **CSS**: “linguaggio *usato per definire la formattazione di documenti HTML”* – Wikipedia. È stato scelto poiché è lo standard utilizzato per la definizione della struttura di una web application.
* **PHP**: “linguaggio *di scripting interpretato utilizzato per sviluppare applicazioni web lato server, ma può essere usato anche per scrivere script a riga di comando o applicazioni stand-alone con interfaccia grafica.”* – Wikipedia. È stato scelto poiché è lo standard utilizzato per prelevare dei dati contenuti all’interno del server utili alla web application.
* **Slim Framework**: “*a PHP micro framework that helps you quickly write simple yet powerful web applications and APIs*.” – Slim*.* È stato scelto perchè consente di realizzare tecnologie *RESTful*.
* **JavaScript**: *“linguaggio di programmazione orientato agli oggetti e agli eventi, comunemente utilizzato nella programmazione Web lato client.”* – Wikipedia. È stato scelto poiché è lo standard utilizzato per realizzare il Controller delle web application.
* **JQuery**: **“***libreria JavaScript per applicazioni web”* – Wikipedia. È stato scelto poiché *“jQuery is a lightweight, "write less, do more", JavaScript library. The purpose of jQuery is to make it much easier to use JavaScript.”* – W3Schools.
* **DataTables**: *“plug-in for the jQuery Javascript library. It is a highly flexible tool, built upon the foundations of progressive enhancement, that adds all of these advanced features to any HTML table.”* – datatables.net. È stato scelto poiché è utile per la pagination delle tabelle.
* **JQuery Pagination plugin**: *“plugin that simplifies the usage of Bootstrap Pagination. It uses appropriate classes.”* – josecebe.github.io/twbs-pagination. È stato scelto poiché è utile per la pagination dei prodotti venduti mostrati nel sito.
* **Web3.js**: *“a collection of libraries that allow you to interact with a local or remote ethereum node using HTTP, IPC or WebSocket.”* – Web3.Js. È stato scelto poiché è lo standard per interagire con gli Smart Contract delle blockchain in Ethereum.
* **MetaMask**: *“provides the simplest yet most secure way to connect to blockchain-based applications. You are always in control when interacting on the new decentralized web.” – metamask.io*. È stato scelto poiché è un mezzo di connessione sicuro alla blockchain per il pagamento in ETH.
* **MySQL**: È un RDBMS utilizzato per memorizzare i diversi dati in un posto unico con un formato standard, per poterli poi eventualmente riutilizzare.
* **Apache Web Server**: “*software che realizza le funzioni di trasporto delle informazioni, di internetwork e di collegamento, ed ha il vantaggio di offrire funzioni di controllo per la sicurezza come quelle effettuate da un proxy.”* – Wikipedia
* **Truffle**: *“the most popular development framework for Ethereum with a mission to make your life a whole lot easier.”* – trufflesuite.com
* **Ganache**: “Quickly fire up a personal Ethereum blockchain which you can use to run tests, execute commands, and inspect state while controlling how the chain operates.” – trufflesuite.com
* **Solidity**: *“a contract-oriented, high-level language for implementing smart contracts. It was influenced by C++, Python and JavaScript and is designed to target the Ethereum Virtual Machine (EVM).”* – docs.soliditylang.org.
* **OpenZeppelin**: “*The standard for secure blockchain applications, it provides security products to build, automate, and operate decentralized applications.”* -openzeppelin.com

## Riferimenti:

* <https://getbootstrap.com/>
* <https://www.slimframework.com/>
* https://jquery.com/
* <https://datatables.net>
* <http://josecebe.github.io/twbs-pagination/>
* <https://web3js.readthedocs.io>
* <https://metamask.io/>
* <https://www.mysql.com/>
* <https://httpd.apache.org/>
* <https://www.trufflesuite.com/truffle>
* <https://www.trufflesuite.com/ganache>
* <https://docs.soliditylang.org/>
* <https://openzeppelin.com/>

# Progettazione sistema:

Di seguito è stata rappresentata graficamente quella che è l’architettura del sistema sia a livello hardware che software:



# Header.php - Cliente

Questo documento contiene:

* Codice HTML per definire la struttura della “Navbar” contenente:
  + Link alla pagina Home.
  + Link all’Utente.
    - Se utente non loggato:
      * Link alla pagina di Registrazione.
      * Link alla pagina di Accesso.
    - Se utente loggato:
      * Link alla pagina del Profilo.
      * Link ad Esci.
  + Link alla pagina Chi Siamo.
  + Link alla sezione del “Footer” Contattaci.
  + Link al carrello (se utente loggato).
  + Link alla sezione di ricerca dei prodotti.
* Codice CSS per gestire lo stile della pagina.
* Codice PHP per gestire i link da far visualizzare all’utente nel caso sia loggato o non loggato.

Questa pagina viene inclusa in tutte le pagine del sito.

## Logout.inc.php:

Questo documento contiene:

* Codice PHP per gestire la disconnessione dell’utente.

# Header.php - Impiegato:

Questo documento contiene:

* Codice HTML per definire la struttura della “Navbar” contenente:
  + Se utente loggato:
    - Link ad Esci.
* Codice CSS per gestire lo stile della pagina.
* Codice PHP per gestire i link da far visualizzare all’impiegato nel caso sia loggato o non loggato.

Questa pagina viene inclusa in tutte le pagine del sito.

## Logoutemp.inc.php:

Questo documento contiene:

* Codice PHP per gestire la disconnessione dell’impiegato.

# Footer.php

Questo documento contiene:

* Codice HTML per definire la struttura contenente:
  + Una breve descrizione.
  + Sezione contatti con link di collegamento in base al tipo (chiamata, e-mail, posizione).
  + Copyright.
* Codice CSS per gestire lo stile della pagina.

Questa pagina viene inclusa in tutte le pagine del sito.

# Index.php - Cliente:

Questo documento contiene:

* Codice HTML per definire la struttura della pagina home contenente:
  + Slider di immagini.
  + Testi descrittivi.
  + Card dei diversi prodotti venduti.
* Codice CSS per gestire lo stile della pagina.

# loadCardProducts.js:

Questo documento contiene:

* Codice JavaScript per gestire:
  + L’aggiunta dei prodotti nel carrello.
  + L’estrapolazione e memorizzazione del carrello.
  + La visualizzazione di messaggi personalizzati a schermo.
  + L’inserimento e la visualizzazione nei prodotti nelle card.
  + La pagination delle card. (Attraverso l’uso di un plugin)

# Index.php - Impiegato:

Questo documento contiene:

* Codice HTML per definire la struttura della pagina di accesso contenente:
  + Un form che deve essere compilato per intero per poter autenticare l’impiegato.
* Codice CSS per gestire lo stile della pagina.
* Codice PHP per gestire:
  + L’autenticazione dell’impiegato.
  + La visualizzazione degli errori nel caso di mancata previa registrazione/compilazione dei campi del form.

## Loginemp.inc.php:

Questo documento contiene:

* Codice PHP per gestire i diversi errori dovuti alla non compilazione corretta dei campi da parte dell’utente in fase di login.
* Codice PHP per gestire l’accesso dell’impiegato al sistema.

# Signup.php:

Questo documento contiene:

* Codice HTML per definire la struttura della pagina di registrazione contenente:
  + Un form che deve essere compilato per intero per poter registrare l’utente nel database di sistema.
* Codice CSS per gestire lo stile della pagina.
* Codice PHP per gestire:
  + La registrazione dell’utente all’interno del database di sistema.
  + La visualizzazione degli errori nel caso di mancata registrazione/compilazione dei campi del form.

## Signup.inc.php:

Questo documento contiene:

* Codice PHP per gestire i diversi errori dovuti alla non compilazione corretta dei campi da parte dell’utente in fase di registrazione.
* Codice PHP per gestire l’iscrizione dell’utente al sistema.

# Login.php:

Questo documento contiene:

* Codice HTML per definire la struttura della pagina di accesso contenente:
  + Un form che deve essere compilato per intero per poter autenticare l’utente.
* Codice CSS per gestire lo stile della pagina.
* Codice PHP per gestire:
  + L’autenticazione dell’utente.
  + La visualizzazione degli errori nel caso di mancata previa registrazione/compilazione dei campi del form.

## Login.inc.php:

Questo documento contiene:

* Codice PHP per gestire i diversi errori dovuti alla non compilazione corretta dei campi da parte dell’utente in fase di login.
* Codice PHP per gestire l’accesso dell’utente al sistema.

# Chisiamo.php:

Questo documento contiene:

* Codice HTML per definire la struttura della pagina “Chi Siamo” contenente:
  + Una descrizione generale del team di sviluppo.
  + Immagine, Descrizione e Social per ognuno degli sviluppatori.
* Codice CSS per gestire lo stile della pagina.

# Cart.php:

Questo documento contiene:

* Codice HTML per definire la struttura della pagina del carrello contenente:
  + Una tabella riepilogativa contenente i prodotti selezionati con la relativa quantità e prezzo finale sia singolo che totale in relazione alla quantità.
* Codice CSS per gestire lo stile della pagina.

# Cart.js:

Questo documento contiene:

* Codice JavaScript per gestire:
  + L’eliminazione di un prodotto dal carrello.
  + L’eliminazione di tutti i prodotti dal carrello.
  + La visualizzazione dei prodotti all’interno del carrello.
  + Il collegamento con la pagina di “Checkout”.

# Checkout.php:

Questo documento contiene:

* Codice HTML per definire la struttura della pagina di pagamento contenente:
  + Un form dei dati della carta di credito che deve essere compilato per poter terminare con successo l’acquisto.
  + Una sezione dedicata al pagamento in ETH tramite una blockchain Ethereum.
  + Una sezione dedicata al pagamento in ETH tramite MetaMask.
* Codice CSS per gestire lo stile della pagina.

# checkoutCreditCard.js

Questo documento contiene:

* Codice JavaScript per gestire:
  + Il controllo che il cliente abbia compilato tutti i campi del form e in maniera valida.
  + L’acquisto dei prodotti che sono contenuti nel carrello.

# checkoutETH.js

Questo documento contiene:

* Codice JavaScript per gestire:
  + L’inserimento dei dati relativi al proprio account di blockchain Ethereum tramite il quale si vuole effettuare l’acquisto
  + L’acquisto dei prodotti che sono contenuti nel carrello.
  + Il controllo che la chiave privata o l’indirizzo inserito siano corretti
  + Il controllo che il cliente abbia abbastanza criptovaluta per concludere l’acquisto
  + La conversione dell’importo della spesa totale da euro a ETH
  + La transazione in blockchain e il pagamento dei prodotti per mezzo dello Smart Contract.

# metamask.js

Questo documento contiene:

* Codice JavaScript per gestire:
  + Il pagamento tramite MetaMask

# Search.php:

Questo documento contiene:

* Codice HTML per definire la struttura della pagina dei risultati della ricerca di un prodotto contenente:
  + Card dei diversi prodotti venduti.
* Codice CSS per gestire lo stile della pagina.

# SearchBook.js

Questo documento contiene:

* Codice JavaScript per gestire:
  + La visualizzazione dei prodotti presenti nelle card.
  + L’aggiunta del prodotto nel carrello.

# Profile.php:

Questo documento contiene:

* Codice HTML per definire la struttura della pagina Profilo dell’utente contenente:
  + Lo storico degli ordini.
    - Contiene una tabella dei prodotti ordinati indicando se questi sono stati spediti o meno.
  + Dettagli dell’account.
    - Contiene un form per poter modificare i dati dell’utente contenuti nel database di sistema.
  + Visualliza operazioni ETH
    - Contiene un form dove l’utente deve inserire il proprio indirizzo blockchain e la propria chiave privata. Una volta che i dati inseriti sono validi ci si collega con Ethereum. (ProfileETH.js) Dove si potrà vedere il proprio bilancio in ETH e gli ordini effettuati con lo Smart Contract.
* Codice CSS per gestire lo stile della pagina.

# Profile.js:

Questo documento contiene:

* Codice JavaScript per gestire:
  + La visualizzazione dello storico degli ordini attraverso una tabella.
  + La visualizzazione dei dati dell’utente presenti nel form.
  + La modifica dei dati dell’utente.
  + L’accesso tramite indirizzo e chiave privata per poi poter collegarsi con Ethereum alla pagina riguardante le operazioni effettuate con lo Smart Contract.

# ProfileETH.js:

Questo documento contiene:

* Codice JavaScript per gestire:
  + La visualizzazione del bilancio in ETH.
  + La visualizzazione dello storico degli ordini pagati in ETH attraverso una tabella.

## Editdetails.inc.php:

Questo documento contiene:

* Codice PHP per gestire i diversi errori dovuti alla non compilazione corretta dei campi da parte dell’utente in fase di modifica dei propri dati.
* Codice PHP per gestire la modifica dei dati dell’utente contenuti nel sistema.

# Dashboard.php:

Questo documento contiene:

* Codice HTML per definire la struttura della pagina Profilo dell’utente contenente:
  + L’acquisto di nuove copie dei prodotti.
    - Contiene una tabella dei prodotti disponibili con la possibilità di acquistare nuove copie.
  + L’acquisto di nuovi prodotti.
    - Contiene un form per poter acquistare nuovi prodotti.
  + La spedizione dei prodotti
    - Contiene una tabella dei prodotti ordinati ancora non spediti con la possibilità di spedirli.
  + La statistica sulle vendite dei prodotti
    - Contiene dei grafici inerenti alle vendite per ciascun prodotto sulle vendite totali.
  + La visualizzazione di ciò che è inerente al mondo della Blockchain/Smart Contract
    - Visualizzazione del bilancio del negozio, con la relativa criptovaluta presente nello Smart Contract con la possibilità di ritirare tale somma
    - Contiene una tabella degli ordini effettuati dai cliente attraverso le modalità di pagamento inerenti all’utilizzo della cripto valuta ETH.
* Codice CSS per gestire lo stile della pagina.

# Dashboard.js:

Questo documento contiene:

* Codice JavaScript per gestire:
  + L’inserimento di nuove copie del prodotto.
  + L’inserimento di un nuovo prodotto.
  + La spedizione di un prodotto.
  + La visualizzazione del grafico delle statistiche.
  + Visualizzazione del bilancio del negozio in ETH.
  + Visualizzazione del saldo presente all’interno dello Smart Contract.
  + Il versamento della cripto valuta presente nello Smart Contract nel saldo del negozio
  + La visualizzazione degli ordini effettuati dai clienti attraverso l’utilizzo dello Smart Contract.

# Funzioni incluse dagli altri files:

## Dbh.inc.php:

Questo documento contiene:

* Codice PHP per gestire la connessione con il database.

## getSmartContractInfo.js:

Questo documento contiene:

* Codice JavaScript per gestire:
  + L’ ottenimento delle informazioni che sono contenute nel database di sistema necessarie al collegamento con lo Smart Contract.
  + La connessione con la blockchain e lo Smart Contract.

## getSmartContractInfo.inc.php:

Questo documento contiene:

* Codice PHP per gestire:
  + L’ ottenimento delle informazioni che sono contenute nel database di sistema necessarie al collegamento con lo Smart Contract.

## isLogged.inc.php:

Questo documento contiene:

* Codice PHP per gestire:
  + Il controllo se l’utente è loggato o meno.

## getAddressETH.inc.php:

Questo documento contiene:

* Codice PHP per gestire:
  + L’ottenimento dell’indirizzo Ethereum del cliente.

## buyNewBook.inc.php:

Questo documento contiene:

* Codice PHP per gestire:
  + L’inserimento di un nuovo libro nel database del sistema
  + L’aggiunta dell’immagine di copertina inserita tramite form nella cartella contenente tutte le immagini presenti sul sito

## Function.inc.php:

Questo documento contiene:

### emptyInputSignup:

Codice PHP usato per controllare che l’utente abbia immesso tutti i dati all’interno dei campi di registrazione.

### invalidEmail:

Codice PHP usato per controllare che l’utente abbia inserito una e-mail valida nel campo e-mail.

### emailExists:

Codice PHP usato per controllare che l’utente non stia usando una e-mail che già esiste ed è già usata.

### passwordMatch:

Codice PHP usato per controllare che la password inserita e la password inserita per la verifica coincidano.

### invalidPassword:

Codice PHP usato per controllare che la password rispetti i criteri.

### createUser:

Codice PHP usato per registrare l’utente all’interno del database di sistema.

### emptyInputLogin:

Codice PHP usato per controllare che l’utente abbia immesso tutti i dati all’interno dei campi di accesso.

### loginUser:

Codice PHP usato per garantire l’accesso da parte dell’utente al sistema.

* viewDetails  
  codice PHP usato per visualizzare i dati dell’account dell’utente.

### emailDetailsExists

Codice PHP usato per controllare che la stessa e-mail non sia già utilizzata.

### editUserDetails

Codice PHP usato per poter modificare i dati dell’account dell’utente.

### loginEmp

Codice PHP usato per garantire l’accesso da parte dell’impiegato al sistema.

updateQuantity

Codice PHP usato per aggiornare la quantità di prodotto disponibile all’acquisto del cliente.

### viewFormat

Codice PHP usato per visualizzare il tipo di formato del prodotto utile durante la fase di inserimento di un nuovo prodotto da parte dell’impiegato.

### viewLanguage

* Codice PHP usato per visualizzare la lingua del prodotto utile durante la fase di inserimento di un nuovo prodotto da parte dell’impiegato.

viewGenre

* Codice PHP usato per visualizzare il genere del prodotto utile durante la fase di inserimento di un nuovo prodotto da parte dell’impiegato.

### buyNewBook

Codice PHP usato per poter inserire nel database di sistema un nuovo libro da parte dell’impiegato.

### viewAddressETH

Codice PHP usato per gestire il collegamento con Ethereum.

## PassShow.js

Questo documento contiene:

* Codice JavaScript per visualizzare/nascondere la password.

## chart.js

Questo documento contiene:

Codice JavaScript usato per ottenere le statistiche di vendita dei prodotti sulle vendite totali e mostrarli attraverso un grafico.

## routes.php

Questo documento contiene:

* Codice PHP per gestire dei servizi RESTful. I Route - Endpoint usati si occupano di:
  + Fornire l’elenco e le informazioni sui prodotti disponibili nel database di sistema.
  + Fornire l’elenco degli indirizzi degli account Ethereum dei clienti.
  + Fornire l’elenco dei prodotti che soddisfano i criteri di ricerca.
  + Fornire l’elenco delle informazioni sui prodotti che sono contenuti nel carrello del cliente
  + Gestire il processo d’acquisto di uno o più prodotti
  + Gestire il processo di verificare che la quantità desiderata di prodotto sia ancora disponibile ed eventualmente decrementarla nel database di sistema a quella attuale.
  + Gestire il processo di aggiornamento della quantità di prodotto decrementata quando la transazione in blockchain non va a buon fine
  + Fornire al cliente l’elenco dei suoi ordini pagati con carta di credito.
  + Fornire all’impiegato le statistiche sulle vendite dei prodotti.
  + Fornire all’impiegato l’elenco dei prodotti che devono essere spediti ai clienti.
  + Gestire il processo di spedizione degli ordini (cambiare lo stato salvato nel database di sistema).
  + Gestire il processo di aggiornamento della quantità di prodotto disponibile nel database di sistema da parte dell’impiegato.
  + Fornire il numero di prodotti totali contenuti nel database di sistema (utile al sistema di Pagination).
  + Fornire l’elenco dei prodotti contenuti nel sistema (secondo il sistema Pagination che si limita ad N risultati).

# Use Case Test:

|  |  |  |
| --- | --- | --- |
| Caso di test | Azione | Risultati attesi |
| 1 | Click sul link home della navbar | Rimani sulla home |
| 2 | Click sul link Chi siamo della Navbar | Si viene rediretti nella pagina chi siamo |
| 3 | Click sul link Contattaci della Navbar | Si viene rediretti nella sezione del footer che contiene i contatti |
| 4 | Click sul link posizione nella sezione contatti del footer | Si viene rediretti nella pagina di Google Maps |
| 5 | Click sul link del numero di telefono nella sezione contatti del footer | Si avvia la possibilità di fare una chiamata con una specifica applicazione |
| 6 | Click sul link della e-mail nella sezione contatti del footer | Si avvia la possibilità di poter inviare una e-mail tramite una specifica applicazione |
| 7 | Collegamento alla pagina index del sito | Vengono caricati i prodotti con le relative specifiche all’interno di card nella pagina |
| 8 | Click sul link utente della Navbar | Viene mostrato un menu a tendina con le voci per garantire l’accesso o la registrazione al sistema |
| 9 | Click sul link Sezione ricerca | Si viene rediretti nella pagina di Ricerca |
| 9 | Premere il tasto cerca senza inserire nulla | Viene richiesto di inserire il testo |
| 10 | Ricercare un prodotto dopo aver inserito i parametri di ricerca | Si viene rediretti alla pagina con i risultati (esito positivo) o un messaggio di non disponibilità (esito negativo). |
| 11 | Click su Accesso, contenuto nel menu a tendina nel link Utente della Navbar | Si viene rediretti alla pagina di login |
| 12 | Click su Registrazione contenuto nel menu a tendina nel link Utente della Navbar | Si viene rediretti alla pagina di registrazione |
| 13 | Aggiunta al carrello di un prodotto senza aver effettuato l’accesso | Si viene rediretti alla pagina di accesso |
| 14 | Aggiunta al carrello di un prodotto dopo aver effettuato l’accesso | Viene aggiunto l’item al carrello e il contatore associato all’icona viene incrementato |
| 15 | Aggiunta al carrello di un prodotto già immesso | La quantità viene aggiornata, sommando quella già presente con quella appena inserita |
| 16 | Se il prodotto ha quantità pari a zero o superiore alla quantità presente | Non è possibile aggiungere il prodotto al carrello |
| 17 | Premere il tasto aggiungi al carrello senza aver specificato la quantità desiderata | Viene richiesto di specificare la quantità |
| 18 | Immissione di un valore di quantità desiderata di prodotto non valida | Viene richiesto di immettere un valore valido |
| 19­ | Compilare tutti i campi della pagina di accesso con credenziali valide | Si viene rediretti alla home del sito ma con i permessi garantiti dopo l’autenticazione |
| 20 | Premere accedi senza aver compilato tutti i campi della pagina accesso o avendo compilato i campi con valori non validi | Vengono visualizzati messaggi di errore |
| 21 | Compilare tutti i campi della pagina di registrazione con credenziali valide | Si viene rediretti alla pagina di login |
| 22 | Premere registrami senza aver compilato tutti i campi della pagina registrazione o avendo compilato i campi con valori non validi | Vengono visualizzati messaggi di errore |
| 23 | Premere registrati nella pagina di login | Si viene rediretti nella pagina di registrazione contenente un form |
| 24 | Una volta fatto l’accesso | Nella Navbar compare il messaggio Ciao \*Nome Utente\* |
| 25 | Una volta fatto l’accesso | Sotto il tab dedicato all’utente si hanno due opzioni: Profilo Utente ed Esci |
| 26 | Cliccando Profilo Utente | Compare la schermata dove è possibile selezionare lo storico degli ordini, i dettagli del proprio account, gli ordini in ETH e inoltre viene visualizzata una animazione sulle spedizioni |
| 27 | Selezionando “Visualizza Storico Ordini” | Viene mostrata una tabella contenente tutti i prodotti di cui si è effettuato l’ordine indicando l’eventuale spedizione |
| 28 | Selezionando “Visualizza Dettagli Account” | Viene mostrato un form contenete i vari dati dell’utente loggato con la possibilità di modificarli. |
| 29 | Selezionando “Visualizza Operazioni ETH” | Viene mostrato un form dove si deve inserire il proprio indirizzo blockchain con la propria chiave privata per potersi collegare e visualizzare dettagli sulle operazioni effettuate tramite la cripto valuta ETH. |
| 30 | Se le credenziali immesse non sono valide | Visualizzato messaggio d’errore |
| 31 | Collegatosi con Ethereum tramite l’apposito form | Sarà possibili visualizzare il proprio bilancio ETH e gli ordini effettuati attraverso la cripto valuta. |
| 32 | Selezionando “Visualizza Dettagli Account” e cliccando il pulsante Salva | Cambierà i dati dell’utente loggato con i dati modificati |
| 33 | Cliccando su Esci | Effettuerà il Log Out dell’utente che aveva effettuato l’accesso. |
| 34 | Effettuato uno o più ordini per un prodotto, cliccando sul carrello e infine sul bottone Check-Out | Apre una nuova pagina contenente un form per effettuare il Check-Out in 3 modalità diverse: Carta di Credito, Blockchain Ethereum e Meta Mask. |
| 35 | Selezionando “Carta di credito” e compilato tutti i campi del form con dati validi e cliccando “Conferma l’ordine” | L’ordine è andato a buon fine |
| 36 | Se non si compilano i campi | Sistema segnalerà la mancanza di dati e non permetterà di portare alla fine il procedimento. |
| 37 | Il campo Titolare Carta ha un vincolo | I caratteri inseriti devono essere alfabetici e non numerici |
| 38 | Il campo Numero della Carta ha un vincolo | I caratteri inseriti non devono essere alfabetici ma numerici e pari a 16 |
| 39 | Il campo MM ha un vincolo | I caratteri inseriti non devono essere alfabetici ma numerici e deve essere compreso tra 1 e 31 |
| 40 | Il campo YY ha un vincolo | I caratteri inseriti non devono essere alfabetici ma numerici e deve essere uguale o maggiore di ventuno |
| 41 | Il campo CVV presenta un elemento che da informazioni aggiuntive sul campo. | Indica il numero e la posizione delle cifre indicate, non è possibile inserire caratteri alfabetici |
| 42 | Se i campi del form quali: Gas Price, Gas Limit, Chiave Privata non vengono compilati in maniera appropriata | Viene visualizzato un messaggio d’errore e non è possibile continuare con il pagamento |
| 43 | Selezionando “Blockchain Ethereum” e cliccando Paga | Si aprirà un form nel quale si potrà visualizzare il proprio indirizzo Blockchain. Una volta inserita la chiave privat sarà possibile concludere l’acquisto |
| 44 | Selezionando “Meta Mask” e cliccando Paga | Il plugin di Meta Mask si aprirà permettendo all’utente di inserire i proprio dati per concludere l’acquisto |
| 45 | Se la cripto valuta non risultasse essere sufficiente per concludere il pagamento | Il pagamento non va a buon fine e si interrompe fase d’acquisto |
| 46 | Se la transazione non dovesse andare a buon fine | All’utente verrà notificato l’esito |
| 47 | Pagina principale dell’impiegato | L’impiegato dovrà inserire il proprio username e password |
| 48 | Inserendo la password | È possibile tramite l’apposito bottone visualizzare la password cifrata in chiaro |
| 49 | Se uno o entrambi i campi non sono compilati | Viene visualizzato un messaggio di errore |
| 50 | Se il Nome Utente inserito non è presente | Viene visualizzato un messaggio di errore |
| 51 | Se la Password inserita non è corretta | Viene visualizzato un messaggio di errore |
| 52 | Effettuato l’accesso alla dashboard dell’impiegato appare | Una schermata con dei link ad altre pagine, visualizzazione del grafico delle statistiche e il tasto Esci.  Link ad altre pagine:   * Aggiungi Nuove Copie * Inserisci un nuovo libro * Spedisci * Statistiche Vendite * Smart Contract & Blockchain |
| 53 | Cliccando su Esci | Verrà effettuato il log out dal sistema |
| 54 | Cliccando su Aggiungi Nuove Copie | Viene visualizzata una tabella con tutti i prodotti presenti nel sistema con un relativo form per inserire delle nuove copie |
| 55 | Nel form per aggiungere nuove copie si deve inserire un numero superiore a zero. Il campo deve essere non vuoto. | Il sistema notifica l’errore non procedendo con l’operazione |
| 56 | Cliccando su Inserisci un nuovo libro | Viene visualizzato un form con tutti i campi che contraddistinguono il prodotto e un bottone per memorizzare il nuovo prodotto |
| 57 | Cliccando il tasto Salva | Se tutti i campi sono stati compilati nel modo corretto, il sistema procederà con l’aggiornamento altrimenti segnalerà la mancanza/correttezza di un eventuale dato |
| 58 | Inserimento di un prodotto già esistente | Il sistema segnalerà il mancato completamento dell’operazione di inserimento del prodotto |
| 59 | Cliccando su Spedisci | Viene visualizzata una tabella con tutti i prodotti non ancora spediti presenti nel sistema con un relativo pulsante per la spedizione |
| 60 | Cliccando su Statistiche Vendite | Viene visualizzato un grafico inerente alle statistiche di vendita per ogni prodotto in relazione alle vendite totali |
| 61 | Cliccando su Smart Contract & Blockchain | Viene visualizzato il bilancio del negozio (ETH), il bilancio dello Smart Contract (ETH) e gli ordini effettuati tramite lo Smart Contract.  È possibile ritirare gli Ether dallo Smart Contract (nel deposito del conto del proprietario del negozio). |