

Progetto BankHazard



Indice

Parte 1: Documentazione Front-End



1. Introduzione
 - a. Scopo e obiettivi del Front-End
2. Funzionamento applicazione
 - a. Tecnologie utilizzate
3. Tecnologie utilizzate
 - a. Linguaggi di programmazione, librerie, framework
4. Guida all'uso
 - a. Descrizione delle funzionalità dell'interfaccia utente
5. Guida per gli sviluppatori
 - a. Informazioni su come contribuire al progetto
6. Approfondimenti sul codice
 - a. Funzioni principali
 - b. Funzioni per il funzionamento interno dell'applicazione

Parte 2: Documentazione Back-End

1. Introduzione
 - a. Scopo e obiettivi del back-end
2. Tecnologie utilizzate
 - a. Linguaggi di programmazione, database, librerie, framework
3. Guida per gli sviluppatori
 - a. Informazioni su come contribuire al progetto
4. Approfondimento gioco del Pachinko
 - a. Cos'è il Pachinko?
 - b. Algoritmo retrostante

Sviluppo della documentazione





Organizzazione dei file del progetto *BankHazard*

 BankHazard	20/05/2024 18:39	Cartella di file	
 bankhazardddb.sql	20/05/2024 18:39	File di origine SQL	4 KB

Tutti i file relativi al progetto *BankHazard* sono stati salvati nella cartella 'Projects', salvata a sua volta nella cartella 'htdocs' relativa al servizio XAMPP che abbiamo scaricato in locale sulle postazioni PC da lavoro.

Il file 'bankhazardddb.sql' è il file con estensione .sql che contiene il codice per lavorare con il database relazionale relativo al progetto.

Nella cartella 'BankHazard' si trovano ulteriori file e cartelle:







 BaseFunction	20/05/2024 18:39	Cartella di file	
 Log	20/05/2024 18:39	Cartella di file	
 Pages	20/05/2024 18:39	Cartella di file	
 LogoFinale2.png	20/05/2024 18:39	File PNG	272 KB

Innanzitutto, il file 'LogoFinale2.png' è l'immagine che abbiamo deciso di utilizzare come logo rappresentativo del nostro progetto *BankHazard* (visualizzabile ad inizio documento).







La cartella 'Log' contiene i file (.css, .html, .php) dedicati a:

- Login
- Logout
- Signin

sulla piattaforma digitale di *BankHazard*.

 Login.css	20/05/2024 18:39	File di origine CSS	2 KB
 Login.html	20/05/2024 18:39	File di origine HT...	1 KB
 Login.php	20/05/2024 18:39	File di origine PHP	2 KB
 Logout.php	20/05/2024 18:39	File di origine PHP	1 KB
 Signin.css	20/05/2024 18:39	File di origine CSS	2 KB
 Signin.html	20/05/2024 18:39	File di origine HT...	3 KB
 Signin.php	20/05/2024 18:39	File di origine PHP	3 KB




Nella cartella ‘Pages’, invece, si trovano tutte le cartelle e file relativi alle pagine web della web-app di *BankHazard*:

 carte	20/05/2024 18:39	Cartella di file
 conti	20/05/2024 18:39	Cartella di file
 homepage	20/05/2024 18:39	Cartella di file
 investimenti	20/05/2024 18:39	Cartella di file
 news_e_mercati	20/05/2024 18:39	Cartella di file
 pagamenti	20/05/2024 18:39	Cartella di file




Nella cartella ‘carte’ si trovano tutti i file (.html, .php, .css, .js) dedicati alla vera e propria pagina web riguardo la sezione presa in considerazione fra quelle poi disponibili nella *navbar* della web-app.

Il concetto spiegato in precedenza si applica a tutte le restanti cartelle; segue la rappresentazione grafica dell’organizzazione dei file nelle cartelle appena viste:




1. Cartella ‘carte’

 carte.css	20/05/2024 18:39	File di origine CSS	1 KB
 carte.js	20/05/2024 18:39	JSFile	0 KB
 carte.php	20/05/2024 18:39	File di origine PHP	4 KB









2. Cartella ‘conti’

 conti.css	20/05/2024 18:39	File di origine CSS	1 KB
 conti.js	20/05/2024 18:39	JSFile	1 KB
 conti.php	20/05/2024 18:39	File di origine PHP	4 KB

3. Cartella ‘homepage’

 homepage.css	20/05/2024 18:39	File di origine CSS	1 KB
 homepage.js	20/05/2024 18:39	JSFile	2 KB
 homepage.php	20/05/2024 18:39	File di origine PHP	3 KB

4. Cartella 'investimenti'




 azioni	20/05/2024 18:39	Cartella di file	
 beni_rifugio	20/05/2024 18:39	Cartella di file	
 fondi_comuni	20/05/2024 18:39	Cartella di file	
 obbligazioni	20/05/2024 18:39	Cartella di file	
 pachinko	20/05/2024 18:39	Cartella di file	
 investimenti.css	20/05/2024 18:39	File di origine CSS	1 KB
 investimenti.js	20/05/2024 18:39	JSFile	0 KB
 investimenti.php	20/05/2024 18:39	File di origine PHP	3 KB

Nella cartella 'investimenti' si può notare che, oltre ai soliti file del tipo *nome_cartella.css/.js/.php* sono presenti 5 ulteriori cartelle:




- 'azioni'
- 'beni_rifugio'
- 'fondi_comuni'
- 'obbligazioni'
- 'pachinko'.

Nelle cartelle appena citate si trovano le relative sotto-cartelle e file dedicati alle relative pagine del sito:




4.1 Cartella 'azioni'

 azioni.css	20/05/2024 18:39	File di origine CSS	1 KB
 azioni.js	20/05/2024 18:39	JSFile	0 KB
 azioni.php	20/05/2024 18:39	File di origine PHP	6 KB




4.2 Cartella 'beni_rifugio':

 beni_rifugio.css	20/05/2024 18:39	File di origine CSS	3 KB
 beni_rifugio.js	20/05/2024 18:39	JSFile	2 KB
 beni_rifugio.php	20/05/2024 18:39	File di origine PHP	3 KB








4.3 Cartella 'fondi_comuni':

 fondi_comuni.css	20/05/2024 18:39	File di origine CSS	2 KB
 fondi_comuni.js	20/05/2024 18:39	JSFile	1 KB
 fondi_comuni.php	20/05/2024 18:39	File di origine PHP	3 KB

4.4 Cartella 'obbligazioni':

 obbligazioni.css	20/05/2024 18:39	File di origine CSS	2 KB
 obbligazioni.js	20/05/2024 18:39	JSFile	1 KB
 obbligazioni.php	20/05/2024 18:39	File di origine PHP	3 KB


4.5 Cartella 'pachinko'

 Immagini	20/05/2024 18:39	Cartella di file	
 Cards.php	20/05/2024 18:39	File di origine PHP	2 KB
 Machine.php	20/05/2024 18:39	File di origine PHP	2 KB
 pachinko.css	20/05/2024 18:39	File di origine CSS	2 KB
 pachinko.js	20/05/2024 18:39	JSFile	1 KB
 pachinko.php	20/05/2024 18:39	File di origine PHP	4 KB
 SettingMachine.html	20/05/2024 18:39	File di origine HT...	3 KB







4.5.1 Cartella 'Immagini'



5. Cartella 'news_e_mercati'

 news_e_mercati.css	20/05/2024 18:39	File di origine CSS	1 KB
 news_e_mercati.js	20/05/2024 18:39	JSFile	0 KB
 news_e_mercati.php	20/05/2024 18:39	File di origine PHP	3 KB

6. Cartella 'pagamenti'

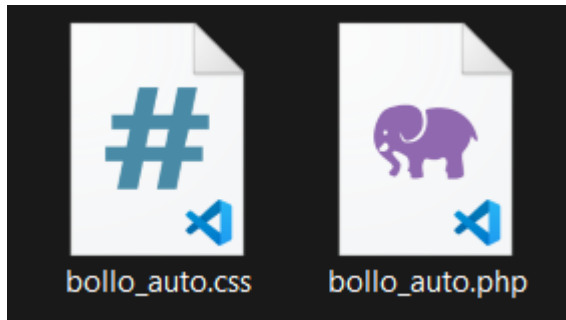
 bollo_auto	20/05/2024 18:39	Cartella di file	
 bonifico	20/05/2024 18:39	Cartella di file	
 ricarica_telefonica	20/05/2024 18:39	Cartella di file	
 pagamenti.css	20/05/2024 18:39	File di origine CSS	1 KB
 pagamenti.js	20/05/2024 18:39	JSFile	0 KB
 pagamenti.php	20/05/2024 18:39	File di origine PHP	3 KB

Nella cartella 'pagamenti' si trovano tre sotto-cartelle:




- 'bollo_auto'
- 'bonifico'
- 'ricarica_telefonica'.

Queste tre cartelle contengono le relative pagine riguardo possibili situazioni e occasioni di pagamenti effettuabili con l'uso di una home banking application:



6.1 Cartella 'bollo_auto'



6.2 Cartella 'bonifico'

 bonifico.css	20/05/2024 18:39	File di origine CSS	1 KB
 bonifico.js	20/05/2024 18:39	JSFile	1 KB
 bonifico.php	20/05/2024 18:39	File di origine PHP	4 KB

6.3 Cartella 'ricarica_telefonica'

 ricarica_telefonica.css	20/05/2024 18:39	File di origine CSS	1 KB
 ricarica_telefonica.php	20/05/2024 18:39	File di origine PHP	4 KB

Infine, per effettuare un recap e avere in mente un *disegno* completo di tutti i file del progetto *BankHazard*, si consiglia la visualizzazione del file pdf al seguente [link](#).

Parte 1: Documentazione Front-End

1. Introduzione - Scopo e obiettivi del Front-End

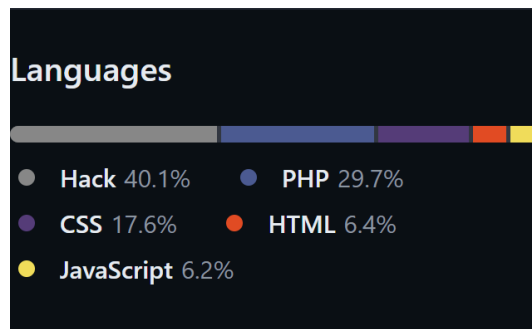
L'obiettivo, di colui che ha sviluppato la parte Front-End, era di:

- creare l'interfaccia utente (UI)
- decidere l'eventuale collocazione di immagini
- decidere l'aspetto da conferire al sito/App
- gestire la guida dell'utente nel processo di navigazione.

BankHazard è un applicativo web sviluppato con l'intento di semplificare l'accesso alle notizie e al proprio conto bancario tramite Internet. Tramite questa applicazione potrete gestire facilmente i vostri conti bancari e le vostre carte tutto da un unico portale.

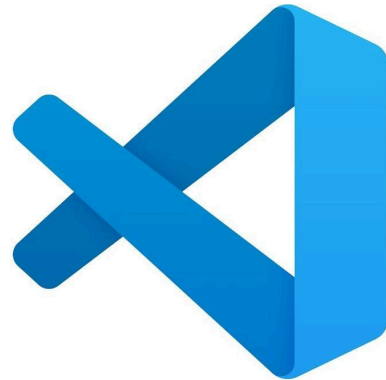
2. Tecnologie utilizzate - Linguaggi di programmazione, librerie, framework

Nel progetto sono stati utilizzati i seguenti linguaggi di programmazione:



Ulteriori tecnologie utilizzate sono state:

- Editor di codice sorgente → [Visual Studio Code](#)
- “ambiente” server → [XAMPP](#)



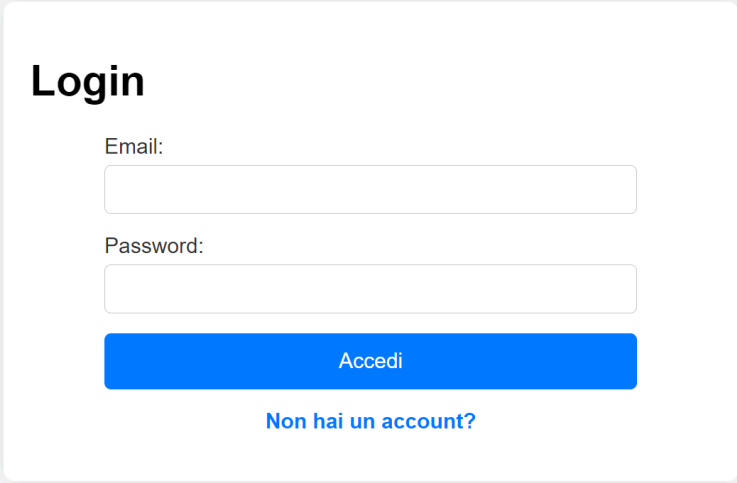
Per l'aggiunta di effetti speciali di stile al sito *BankHazard* si è scelto di utilizzare:

- [Bootstrap](#)
- [Chart.js](#)



Chart.js

Iniziamo dalla pagina di login da cui l'utente accede ai suoi conti.

A login form titled "Login" in bold black text. It features two input fields: "Email:" and "Password:". Below the password field is a blue button labeled "Accedi". At the bottom of the form is a link that says "Non hai un account?". The form is centered on a light purple background.

Login

Email:

Password:

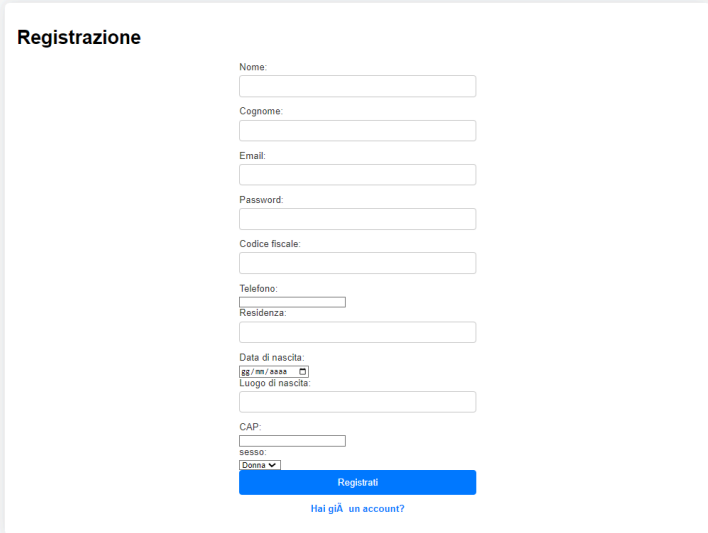
Accedi

[Non hai un account?](#)

Pagina di login

Come si può vedere dall'immagine riportata sopra per accedere l'utente dovrà essere provvisto di una email ed una password. Per gli utenti non registrati sarà disponibile una pagina di signin a cui si può accedere direttamente premendo su [Non hai un account?](#).

Una volta premuto troveremo una pagina con molti campi da compilare.

A registration form titled "Registrazione" in bold black text. It contains multiple input fields for personal information: "Nome:", "Cognome:", "Email:", "Password:", "Codice fiscale:", "Telefono:", "Residenza:", "Data di nascita:" (with a date picker), "Luogo di nascita:", "CAP:", "Sesso:" (with a dropdown menu). At the bottom is a blue button labeled "Registrati" and a link that says "Hai già un account?". The form is centered on a light purple background.

Registrazione

Nome:

Cognome:

Email:

Password:

Codice fiscale:

Telefono:

Residenza:

Data di nascita:

Luogo di nascita:

CAP:

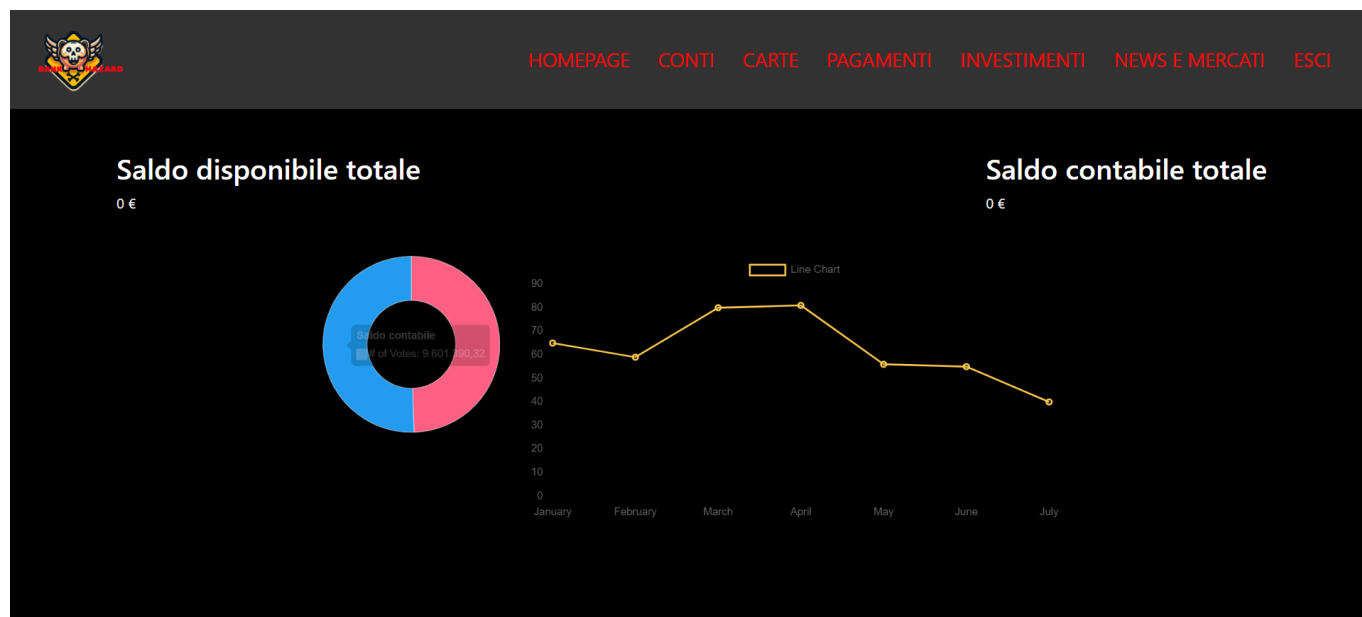
Sesso:

Registrati

[Hai già un account?](#)

Pagina di registrazione

Una volta compilati tutti i campi, verrà prima verificata la validità dell'account ed infine verrà salvato nel database dove i dati sensibili verranno criptati ai sensi della legge del regolamento 2016/679 GDPR.



Homepage piattaforma BankHazard (0)

Nella home page si potranno monitorare velocemente il saldo disponibile e le ultime spese utilizzando un piccolo grafico che segnala le spese durante le 12 mensilità.

Nota bene...

- *Il **saldo contabile** rappresenta l'ammontare totale dei fondi presenti sul conto, inclusi quelli ancora in attesa di essere depositati o prelevati. Questo saldo comprende tutte le operazioni registrate sul conto corrente e visualizzabili nella lista movimenti. Tuttavia, il saldo contabile non viene aggiornato in tempo reale, ma in un momento puntuale. Quindi, potrebbe non corrispondere alla reale liquidità in un dato momento.*
- *Il **saldo disponibile**, invece, indica la somma effettivamente utilizzabile in quel momento. Questo saldo include anche le spese recenti non ancora inserite nella lista movimenti. Il saldo disponibile viene aggiornato in tempo reale e tiene conto di tutte le operazioni, sia in entrata che in uscita.*

Dalla home page si può navigare con la semplice interfaccia di routing (*navbar* di Bootstrap 5) integrata tramite una navbar a capo della pagina come si può notare nell'immagine 1.4.



Navbar piattaforma BankHazard (1)

Tramite la navbar si può passare alle varie pagine da dove si può gestire la pagina selezionata. Tra le pagine navigabili possiamo trovare:

- 'HOMEPAGE', ci riporta alla home page;
- 'CONTI', ci fa vedere la pagina con tutti i conti corrente che abbiamo;
- 'CARTE', dove possiamo controllare tutte le nostre carte (di credito, di debito, bancomat e prepagate);
- 'PAGAMENTI', dove si potranno fare i pagamenti come: bonifici, pagamento bolli, ...;
- 'INVESTIMENTI', dove si potranno investire i propri danari direttamente da un conto o da una carta;
- 'NEWS E MERCATI', dove si potranno leggere le ultime notizie riguardanti il business;
- 'ESCI', che corrisponde al tasto 'logout' che permette di distruggere la sessione legata all'account;

ID Conto

IBAN

Saldo Corrente

Saldo Contabile

Intestatario

Campi tabella 'conti' degli utenti di BankHazard (2)

IBAN

Scadenza

Numero Carta

Saldo Disponibile

Saldo Contabile

CVV

Tipologia

Campi tabella 'carte' degli utenti di BankHazard (3)




Effettua un Pagamento

Bonifico

Pagamento Bollo Auto

Ricarica Telefonica

Opzioni di pagamento sulla piattaforma di BankHazard (4)



[HOMEPAGE](#) [CONTI](#) [CARTE](#) [PAGAMENTI](#) [INVESTIMENTI](#) [NEWS E MERCATI](#) [ESCI](#)

Beneficiario


IBAN Destinatario

Importo

Causale

Procedi

Schermata opzione 'Bonifico' (5)



[HOMEPAGE](#) [CONTI](#) [CARTE](#) [PAGAMENTI](#) [INVESTIMENTI](#) [NEWS E MERCATI](#) [ESCI](#)

Pagamento Bollo Auto

Importo

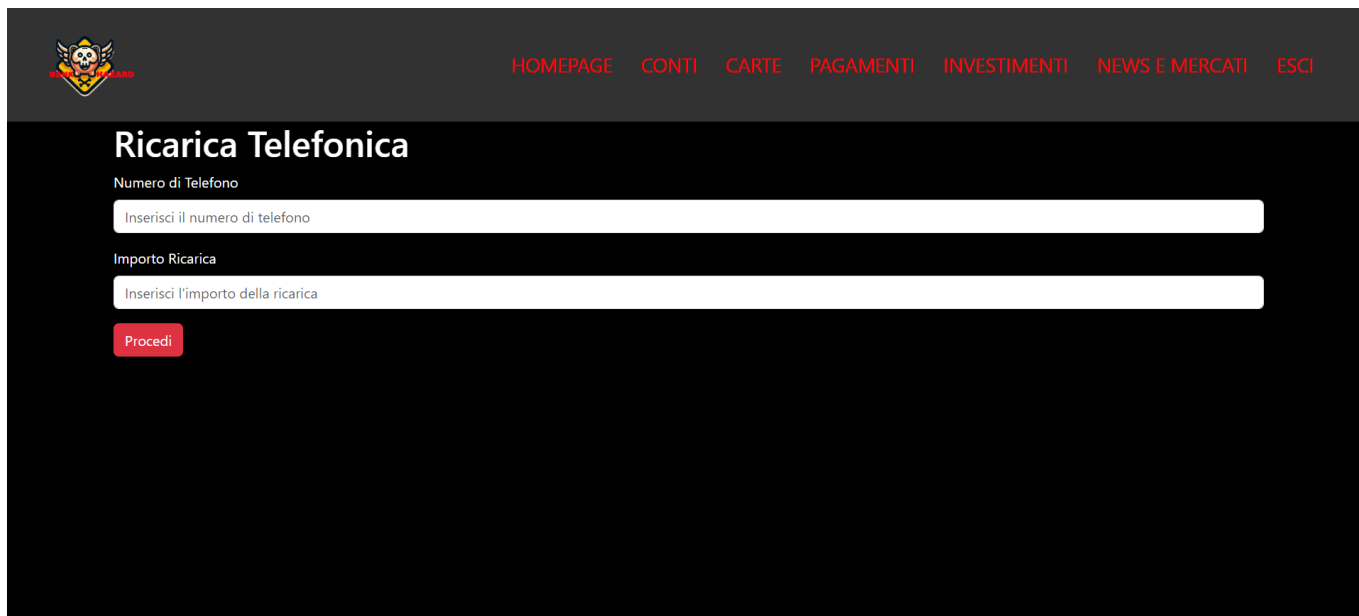
IBAN

Data di scadenza

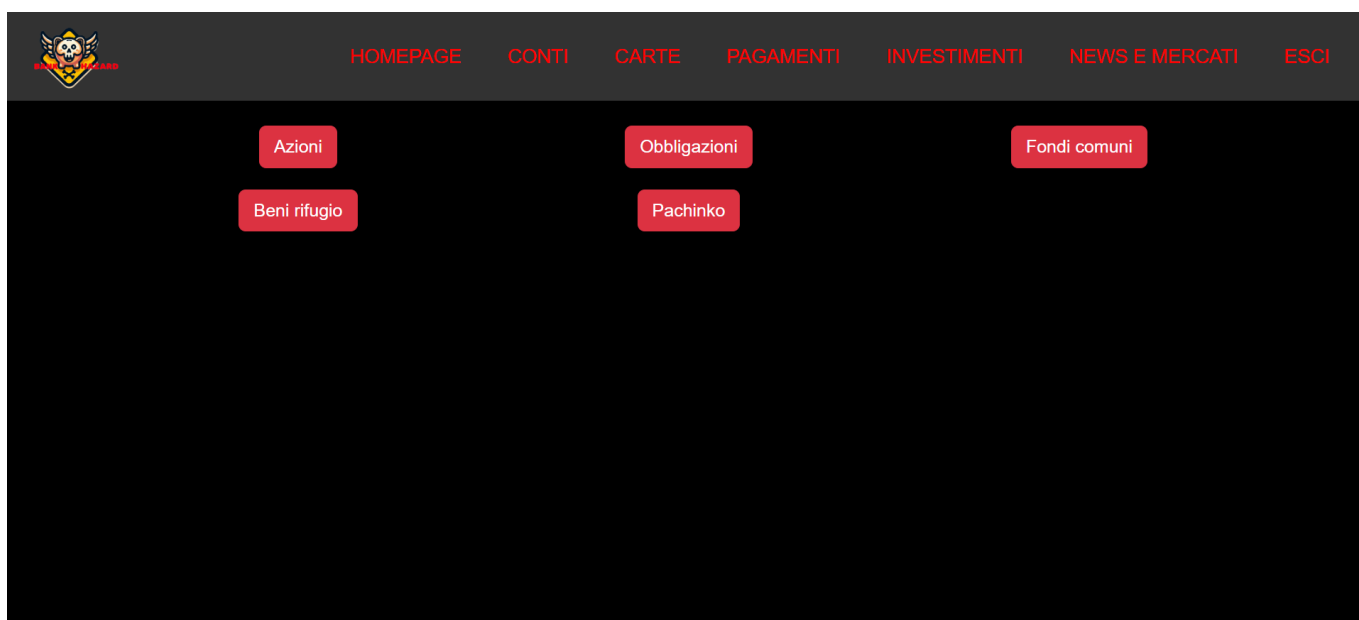
gg/mm/aaaa

Procedi

Schermata opzione 'Pagamento Bollo Auto' (6)



Schermata opzione ‘Ricarica Telefonica’ (7)



Schermata pagina ‘Investimenti’ (8)

Nota bene...

*Le **azioni** e le **obbligazioni** sono due tipi di strumenti finanziari che gli investitori possono utilizzare per costruire un portfolio di investimenti. Sotto segue una spiegazione delle loro differenze principali.*

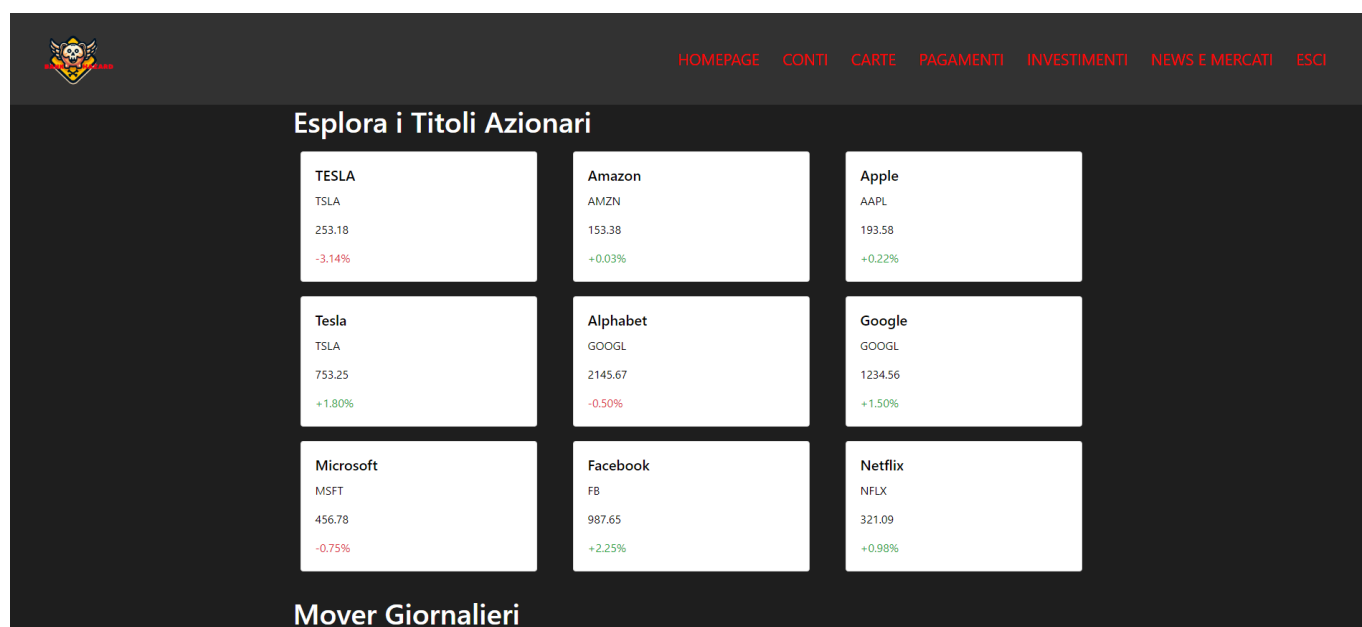
Azioni	Obbligazioni
<i>Quando acquisti azioni, stai acquistando una quota di proprietà in un’azienda quotata in borsa. Come azionista, hai diritto a una parte degli utili dell’azienda, che possono essere</i>	<i>Le obbligazioni sono essenzialmente prestiti che l’investitore fa a un’entità, che può essere un’azienda o un governo. In cambio del prestito, l’emittente dell’obbligazione si impegna</i>

distribuiti sotto forma di dividendi. Il valore delle azioni può fluttuare in base all'andamento dell'azienda e alle condizioni di mercato. Investire in azioni offre la possibilità di guadagni più elevati, ma comporta anche un rischio maggiore.

a pagare all'investitore un interesse periodico (la cedola) e a restituire il capitale prestato alla scadenza del titolo. Le obbligazioni sono generalmente considerate meno rischiose rispetto alle azioni e offrono un flusso di reddito più prevedibile.

Scegliere tra azioni e obbligazioni dipende dagli obiettivi di investimento, dalla tolleranza al rischio e dall'orizzonte temporale dell'investitore. Una strategia di investimento diversificata potrebbe includere entrambi i tipi di strumenti per bilanciare rischio e rendimento.

- **I fondi comuni di investimento** sono strumenti finanziari che permettono agli investitori di mettere in comune il proprio capitale per investirlo in un portfolio diversificato di attività finanziarie, come azioni, obbligazioni e titoli di stato. Questi fondi sono gestiti da professionisti, le Società di Gestione del Risparmio (SGR), che si occupano di selezionare gli investimenti e di gestire il fondo secondo una strategia predefinita.
- **I mover giornalieri**, noti anche come **market mover**, sono fattori che possono influenzare significativamente i mercati finanziari nel corso di una giornata.



Schermata opzione 'Azioni' (10)



[HOMEPAGE](#) [CONTI](#) [CARTE](#) [PAGAMENTI](#) [INVESTIMENTI](#) [NEWS E MERCATI](#) [ESCI](#)

Esplora le Obbligazioni

TESLA

Prezzo corrente: 253.18
Variazione percentuale: -31.4%


Amazon

Apple

Microsoft

Mover Giornalieri
EBIXQ Ethix Inc

Schermata opzione ‘Obbligazioni’ (11)



[HOMEPAGE](#) [CONTI](#) [CARTE](#) [PAGAMENTI](#) [INVESTIMENTI](#) [NEWS E MERCATI](#) [ESCI](#)

Esplora i Fondi Comuni di Investimento


Fondo Azionario Globale
Gestito da Investimenti S.p.A.
Rendimento annuo: 7%
[Investi Ora](#)

Fondo Obbligazionario Europeo
Gestito da Banca Europa
Rendimento annuo: 5.5%
[Investi Ora](#)

Fondo Bilanciato Globale
Gestito da Investimenti Bilanciati Ltd.
Rendimento annuo: 6.2%
[Investi Ora](#)

[Visualizza Tutti](#)

Schermata opzione ‘Fondi comuni’ (12)



[HOMEPAGE](#) [CONTI](#) [CARTE](#) [PAGAMENTI](#) [INVESTIMENTI](#) [NEWS E MERCATI](#) [ESCI](#)

Esplora i Beni Rifugio

Oro
Prezzo attuale: €1500 per oncia
Variazione giornaliera: +0.22%

Argento
Prezzo attuale: €20 per oncia
Variazione giornaliera: -0.15%


USDT
Prezzo attuale: €1
Variazione giornaliera: 0.00%

Mover Giornalieri

Oro
+0.22%

Argento
-0.15%

Schermata opzione ‘Beni Rifugio’ (13)



[HOMEPAGE](#) [CONTI](#) [CARTE](#) [PAGAMENTI](#) [INVESTIMENTI](#) [NEWS E MERCATI](#) [ESCI](#)

Pachinko settings

Quota investita

Modalità di gioco

Schermata opzione ‘Pachinko’ (14)



[Wall Street scende sotto i 5.000 punti. Ma per i prossimi mesi ha due grandi cartucce - Borse, Milano dribbla guerra e tassi](#)

Breve descrizione della notizia 1.

[UniCredit spiega ai clienti come evitare le frodi: ecco i 3 punti chiave](#)

Breve descrizione della notizia 2.

[Se \(anche\) le banche remano contro il ribasso dei tassi](#)

Breve descrizione della notizia 2.

[Finanziamenti, Italia più cara di Francia e Germania soprattutto sul credito al consumo](#)

Breve descrizione della notizia 2.

[S&P conferma il rating BBB dell'Italia](#)

Breve descrizione della notizia 2.

<https://it.investing.com/news/latest-news>

Schermata opzione 'News e Mercati' (15)

4. Approfondimenti sul codice - Funzioni principali

4.1 Navbar (routing)

```
<nav>
  <ul class="nav">
    <li class="nav-item">
      <a class="nav-link active" href="../homepage/homepage.php">Homepage</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="../conti/conti.php">Conti</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="../carte/carte.php">Carte</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="../pagamenti/pagamenti.php">Pagamenti</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="../investimenti/investimenti.php">Investimenti</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="../news_e_mercati/news_e_mercati.php">News e Mercati</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="../logout/logout.php">Esci</a>
    </li>
  </ul>
</nav>
```

4.2 Line Chart e grafico "a torta"

```
// Codice JavaScript per il grafico del saldo (a torta)
```

```

var saldoCtx = document.getElementById('saldo-chart').getContext('2d');
var saldoChart = new Chart(saldoCtx, {
  type: 'doughnut',
  data: {
    labels: ['Saldo disponibile', 'Saldo contabile'],
    datasets: [{
      label: '# of Votes',
      data: [9421093.16, 9601390.32],
      backgroundColor: [
        'rgba(255, 99, 132, 1)',
        'rgba(54, 162, 235, 1)'
      ],
      borderColor: [
        'rgba(255,255,255,.5)',
        'rgba(255,255,255,.5)'
      ],
      borderWidth: 1
    }]
  },
  options: {
    responsive: false,
    plugins: {
      legend: {
        display: false
      }
    }
  }
});

// Codice JavaScript per il grafico a linee
var lineCtx = document.getElementById('line-chart').getContext('2d');
var lineChart = new Chart(lineCtx, {
  type: 'line',
  data: {
    labels: ['January', 'February', 'March', 'April', 'May', 'June', 'July'],
    datasets: [{
      label: 'Line Chart',
      data: [65, 59, 80, 81, 56, 55, 40],
      borderColor: 'rgba(255, 206, 86, 1)',
      borderWidth: 2
    }]
  },
  options: {
    scales: {
      y: {
        beginAtZero: true
      }
    }
  }
});

```

4.3 Cards (*HTML + CSS, in realtà lo stile delle cards è unico per ogni pagina*)

```

<div class="col-sm-4">
  <div class="card stock-card">
    <div class="card-body">
      <h5 class="card-title">TESLA</h5>
      <p class="card-text">TSLA</p>
      <p class="card-text">253.18</p>
      <p class="card-text text-danger">-3.14%</p>
    </div>
  </div>
</div>

```

```

.stock-card {
  margin: 10px;
}

.card-body {
  color: #000;
}

```

Parte 2: Documentazione Back-End

1. Introduzione - Scopo e obiettivi del Back-End

Il Back-End nelle applicazioni web serve ad interfacciare la base di dati con l'interfaccia utente, oltre ad agevolare notevolmente la computazione da parte del dispositivo usato dall'utente.

2. Tecnologie utilizzate

a. Linguaggi di programmazione

Nel Back-End è stato utilizzato PHP con funzioni derivanti da PHP 7

b. Database

Per il database è stata utilizzata la tipologia relazionale (o SQL) con DBMS MySQL.

c. Librerie

Per le librerie utilizzate non c'è molto da dire infatti l'unica degna di nota è openssl che permette di criptare i propri dati tramite algoritmi simmetrici e asimmetrici.

d. Framework

Non sono stati usati particolari framework per lo sviluppo del Back-End.

3. Guida alla continuazione e al miglioramento del progetto

Per un eventuale miglioramento del progetto servono innanzitutto delle informazioni di base riguardo alle funzioni del Back-End. Partiamo dalle tre funzioni di base (file BaseFunction.php):

```

public static function DBconnection() : mysqli{
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "bankhazarddb";
    $conn = new mysqli($servername, $username, $password, $dbname);

    if ($conn->connect_error) {
        die("Connessione al database fallita: " . $conn->connect_error);
    }
    return $conn;
}

```

DBconnection serve a connettersi al database e viene nei punti di codice dove devono venire eseguite Query al database.

```

public static function CreateSession() : bool {
    session_save_path();
    return session_start();
}

```

La funzione CreateSession serve a creare la sessione in maniera da mantenere la continuità dei dati.

```

public static function takeID(string $email) : int {
    $conn = BaseFunction::DBconnection();
    return $conn->query("SELECT DISTINCT ID_utente FROM utenti WHERE email = $email")->fetch_assoc();
}

```

takeID serve semplicemente a recuperare l'ID passando l'e-mail dell'utente indicato.

Dopo le funzioni di base abbiamo le funzioni riguardanti la banca:

```

public static function Bonifico(int $idIntestatario, string $nomeIntestatario, float $importo, string $IBAN, string $causale) : bool|string {
    $conn = BaseFunction::DBconnection();
    //disattivo gli autocommit per iniziare una transaction
    $conn->autocommit(false);
    //controllo se l'intestatario ha abbastanza danari
    $sql = "SELECT SaldoCorrente FROM contocorrente WHERE Intestatario = ?";
    $result = $conn->prepare($sql);
    $result->bind_param("i", $idIntestatario);
    $result->execute();
    $result = $result->get_result();
    $saldoCorrente = $result->fetch_assoc()["SaldoCorrente"] - $importo;
    //faccio l'update del conto altrimenti esco dalla funzione
    if($saldoCorrente >= 0){
        $sql = "UPDATE contocorrente SET SaldoCorrente = ? WHERE Intestatario = ?";
        $result = $conn->prepare($sql);
        $result->bind_param("ii", $saldoCorrente, $idIntestatario);
        $result->execute();
        $result = $result->get_result();
    }
}

```

La funzione Bonifico serve per fare i bonifici e sfrutta le transazioni per renderle sicure.

```

public static function ZontaSchei(int $idIntestatario, float $importo) : bool|string {
    $conn = BaseFunction::DBconnection();
    //controllo i danari posseduti dal gentil cittadino
    $sql = "SELECT SaldoCorrente FROM contocorrente WHERE Intestatario = ?";
    $result = $conn->prepare($sql);
    $result->bind_param("i", $idIntestatario);
    $result->execute();
    $result = $result->get_result();
    $saldoCorrente = $result + $importo;
    //faccio l'update del conto
    $sql = "UPDATE contocorrente SET SaldoCorrente = ? WHERE Intestatario = ?";
    $result = $conn->prepare($sql);
    $result->bind_param("ii", $saldoCorrente, $idIntestatario);
    $result->execute();
    $result = $result->get_result();
    return true;
}

```

ZontaSchei è una funzione che permette l'aggiunta di danaro in un account.

```

public static function CavaSchei(int $idIntestatario, int $importo) : bool|string {
    $conn = BaseFunction::DBconnection();
    //controllo i danari posseduti dal gentil cittadino
    $sql = "SELECT SaldoCorrente FROM contocorrente WHERE Intestatario = ?";
    $result = $conn->prepare($sql);
    $result->bind_param("i", $idIntestatario);
    $result->execute();
    $result = $result->get_result();
    $saldoCorrente = $result - $importo;
    //faccio l'update del conto altrimenti esco dalla funzione
    if($saldoCorrente >= 0){
        $sql = "UPDATE contocorrente SET SaldoCorrente = ? WHERE Intestatario = ?";
        $result = $conn->prepare($sql);
        $result->bind_param("ii", $saldoCorrente, $idIntestatario);
        $result->execute();
        $result = $result->get_result();
    }else{
        return "not enought money, for CAPITALISM";
    }
    return true;
}

```

CavaSchei è una funzione che serve a detrarre denaro da un conto o una carta.

```

public static function newCard(string $IBAN, int $idIntestatario, int $numeroCarta, int $CVV, string $tipologia, int $contoCorrelato = null) : void
{
    $conn = BaseFunction::DBconnection();
    if($contoCorrelato == null){
        $insert_query = "INSERT INTO carte (IBAN, Scadenza, NumeroCarta, SaldoDisponibile, SaldoContabile, CVV, Tipologia, Intestatario) VALUES (?, ?, ?, ?, ?, ?, ?)";
        $result = $conn->prepare($insert_query);
        $dataScadenza = self::calcolaScadenza();
        $contantiBase = 0;
        $result->bind_param("ssiddisi", $IBAN, $dataScadenza, $numeroCarta, $contantiBase, $contantiBase, $CVV, $tipologia, $idIntestatario);
        $result->execute();
    }else{
        $insert_query = "INSERT INTO carte (IBAN, Scadenza, NumeroCarta, SaldoDisponibile, SaldoContabile, CVV, Tipologia, Intestatario, ContoCorrelato) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)";
        $result = $conn->prepare($insert_query);
        $dataScadenza = self::calcolaScadenza();
        $contantiBase = 0;
        $result->bind_param("ssiddisii", $IBAN, $dataScadenza, $numeroCarta, $contantiBase, $contantiBase, $CVV, $tipologia, $idIntestatario, $contoCorrelato);
        $result->execute();
    }
}

```

newCard serve per creare una nuova carta.

```

public static function calcolaScadenza() : string
{
    $dataOggi = date("Y/m/d");
    $dataScadenza = date('Y-m-d', strtotime($dataOggi. ' + 5 years'));
    return $dataScadenza;
}

```

calcolaScadenza serve a dare la data di scadenza alle carte.

```

public static function newBankAccount(string $IBAN, int $idIntestatario) : void
{
    $conn = BaseFunction::DBconnection();

    $insert_query = "INSERT INTO carte (IBAN, SaldoDisponibile, SaldoContabile, Intestatario) VALUES (?, ?, ?, ?)";
    $result = $conn->prepare($insert_query);
    $contantiBase = 0;
    $result->bind_param("sddi", $IBAN, $contantiBase, $contantiBase, $idIntestatario);
    $result->execute();
}

```

newbankAccount serve a creare un nuovo conto bancario.

Infine ci sono le funzioni per la criptazione e la decriptazione:

```

public static function Cryptazionamento(string $dataToEncrypt) : string {
    return openssl_encrypt($dataToEncrypt, self::$cypherMethod, self::$key, $options=0, self::$iv);
}

```

Cryptazionamento è una funzione che fa uso della libreria openssl per criptare i dati passati come variabile.

```

public static function Decriptazionamento(string $encryptedData) : string | false {
    return openssl_decrypt($encryptedData, self::$cypherMethod, self::$key, $options=0, self::$iv);
}

```

Decriptazionamento serve a decriptare i dati dal database.

4. Approfondimento gioco del Pachinko

a. Cos'è il Pachinko?

Il Pachinko è un gioco d'azzardo (simile alle nostrane slot machine) originario del Giappone.

La vittoria si ottiene tramite un tris di immagini uguali.

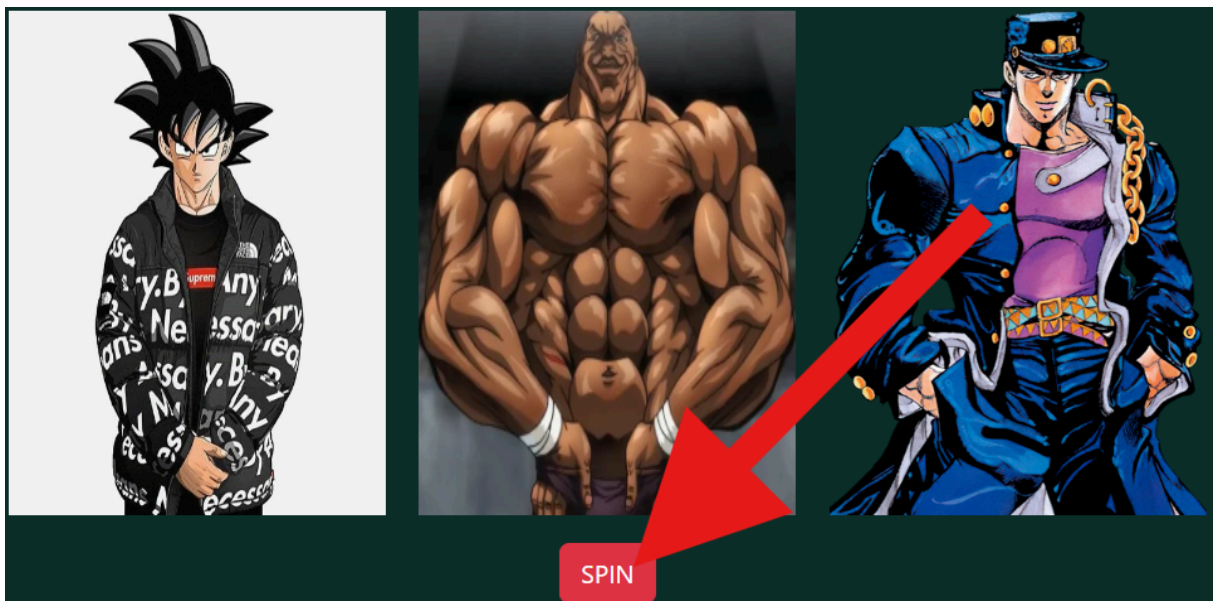
Vista la provenienza, spesso queste simpatiche macchinette, portatrici di problemi finanziari, sono a tema anime o manga (rispettivamente cartoni animati e fumetti).

b. Algoritmo retrostante

L'algoritmo dietro al Pachinko non è complicato, infatti, prevede la generazione casuale di 3 numeri che porteranno all'apparizione dell'immagine associata.

```
function spin(Machine $mac, int &$numGiri) : bool
{
    $first = $mac->reroll();
    $second = $mac->reroll();
    $third = $mac->reroll();
    $numGiri--;
    $json = array('imgLeft' => $first->imgPath, 'imgCentre' => $second->imgPath, 'imgRight' => $third->imgPath);
    file_put_contents('json.json', json_encode($json));
    return $mac->victoryAlgorithm($first, $second, $third);
}
```

La funzione spin simula la giocata che si effettuerebbe tramite l'abbassamento della leva nella macchinetta originale. La funzione spin è richiamata ogni qualvolta venga premuto il pulsante SPIN.



L'algoritmo di vittoria va semplicemente a confrontare gli identificatori delle immagini e può dare due risultati:

- true, nel caso in cui si abbia vinto;
- false, nel caso in cui si perda;


```

function victoryAlgorithm(Card $first, Card $second, Card &$third) : bool {
    if($this->mode)
        if($first == $second && $second == $third && $first == $third){
            $won = self::wonnaWin($third);
            if($won === true){
                $this->incomingProbabilty = self::INITIAL_PROBABILITY;
                return true;
            }else{
                $third->impostaCarta($third->valore);
                $this->incomingProbabilty /= 2;
                return false;
            }
        }else if($first == $second || $second == $third || $third == $first){
            $this->incomingProbabilty /= 2;
            return false;
        }else{
            return false;
        }
    }else{
        if($first == $second && $second == $third && $first == $third){
            $this->incomingProbabilty = self::INITIAL_PROBABILITY;
            return true;
        }else{
            return false;
        }
    }
}

```

La funzione wonnaWin serve per ridurre le probabilità di vincere rendendole di circa 1 su 87000 (al primo giro).

```

function wonnaWin(Card &$val) : bool | int {
    if(rand(0, $this->incomingProbabilty) == 1) return true;
    else if($val->valore < 6) return $val->valore+1;
    else return $val->valore-1;
}

```

Una volta vinto il denaro scommesso viene raddoppiato (speraci).

© - Documentazione BankHazard - Tutti i diritti sono riservati.