

# Progetto PWM (Rivolta Luca)

Tecnologie aggiuntive utilizzate:

- React
- JWT (JSON Web Tokens)

## React

React è una libreria JavaScript sviluppata da Facebook per la creazione di interfacce utente dinamiche e reattive.

Si basa su un approccio basato sui **componenti**, che permette di costruire applicazioni web modulari, riutilizzabili e facilmente mantenibili.

Sostanzialmente ogni componente esporta una funzione JavaScript che ritorna del codice **JSX (JavaScript XML)**, un'HTML esteso, che definisce la struttura del componente.

Vedi qui un semplice esempio di un componente:

```
function SimpleComponent() {
  return (
    <div>
      <h1 className="text-white">Un semplice componente</h1>
    </div>
  );
}

export default SimpleComponent;
```

Ogni componente può avere un suo **stato**, che contiene le informazioni dinamiche che esso utilizza per determinare la sua visualizzazione e comportamento.

Quando lo stato cambia, React ri-renderizza automaticamente il componente (senza toccare il resto del DOM) per riflettere questi cambiamenti nella UI.

A livello implementativo si utilizza un **hook** chiamato `useState`, che permette di definire uno o più stati all'interno di un singolo componente.

Esempio di utilizzo:

```
import React, { useState } from 'react';

function Counter() {
  // count: contiene il valore read-only del contatore (0)
  // setCount: funzione che permette di cambiare lo stato
  const [count, setCount] = useState(0);

  const increment = () => setCount(count + 1);

  return (
    <div>
      <h1>Contatore: {count}</h1>
      <button onClick={increment}>Incrementa</button>
    </div>
  );
}

export default Counter;
```

## JWT

I JSON Web Tokens (JWT) sono uno standard aperto (RFC 7519) utilizzato per trasmettere informazioni in modo sicuro tra le parti come oggetti JSON.

Essi offrono un'alternativa più **scalabile**, sicura ed efficiente rispetto ai session cookies, in particolare quando si lavora con **applicazioni distribuite**.

Un JWT è composto da 3 parti:

1. **header**: contiene l'algoritmo di firma usato e il tipo di token
2. **payload**: contiene le informazioni da trasmettere (dette **claims**)
3. **signature**: garantisce che il token non possa essere alterato da terze parti

La firma si crea combinando l'header e il payload codificati in `base64` e firmandoli con una chiave segreta detenuta unicamente dal server:

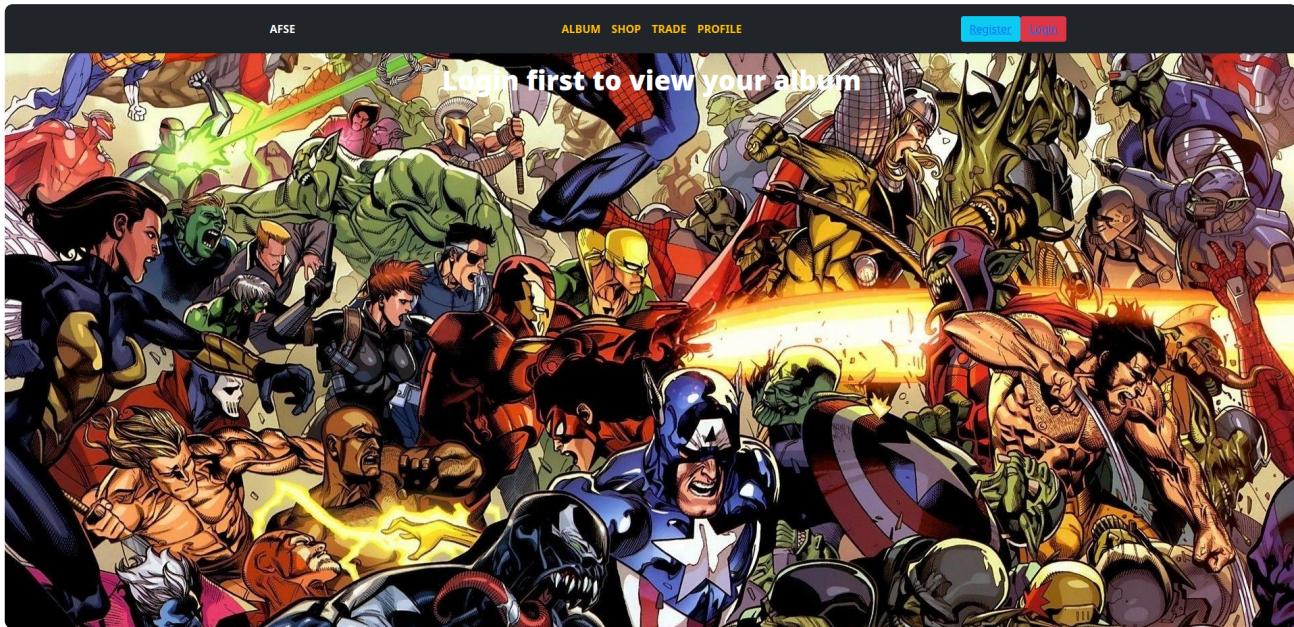
```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret
)
```

Un JWT completo potrebbe apparire in questo modo:

```
eyJhbGciOiAiSFMyNTYiLCJ0eXAiOiAiSldUIIn0.eyJzdWIiOiAiMTIzNDU2Nzg5MCIsICJu
YW1lIjogIkpvag4gRG9lIiwgImlhCI6IDE1MTYyMzkwMjJ9.dQqPZ_l00oq9T9zjD6IU6KU
yfJH_z5H8lI6XU2q7Uu8
```

## FrontEnd

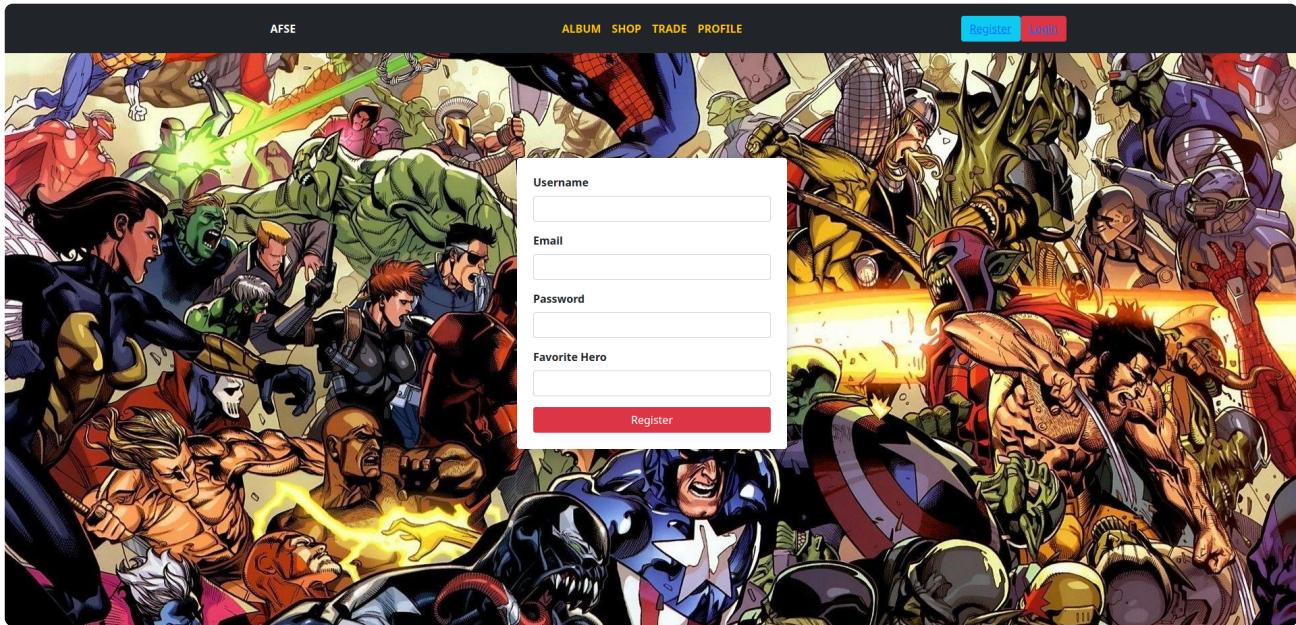
Visitando la pagina principale dell'applicazione per la prima volta ci troviamo sulla schermata sottostante:



## Registrazione

In fase di registrazione viene richiesto di inserire:

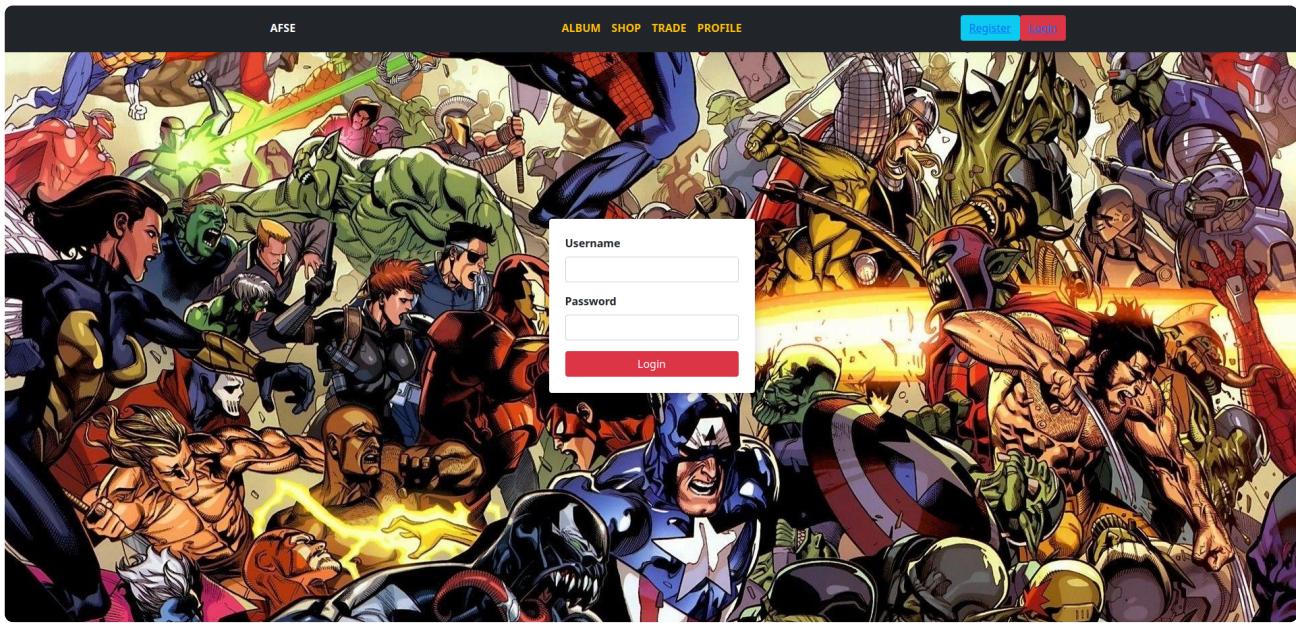
- un nome utente univoco
- una email univoca
- una password
- il proprio supereroe preferito



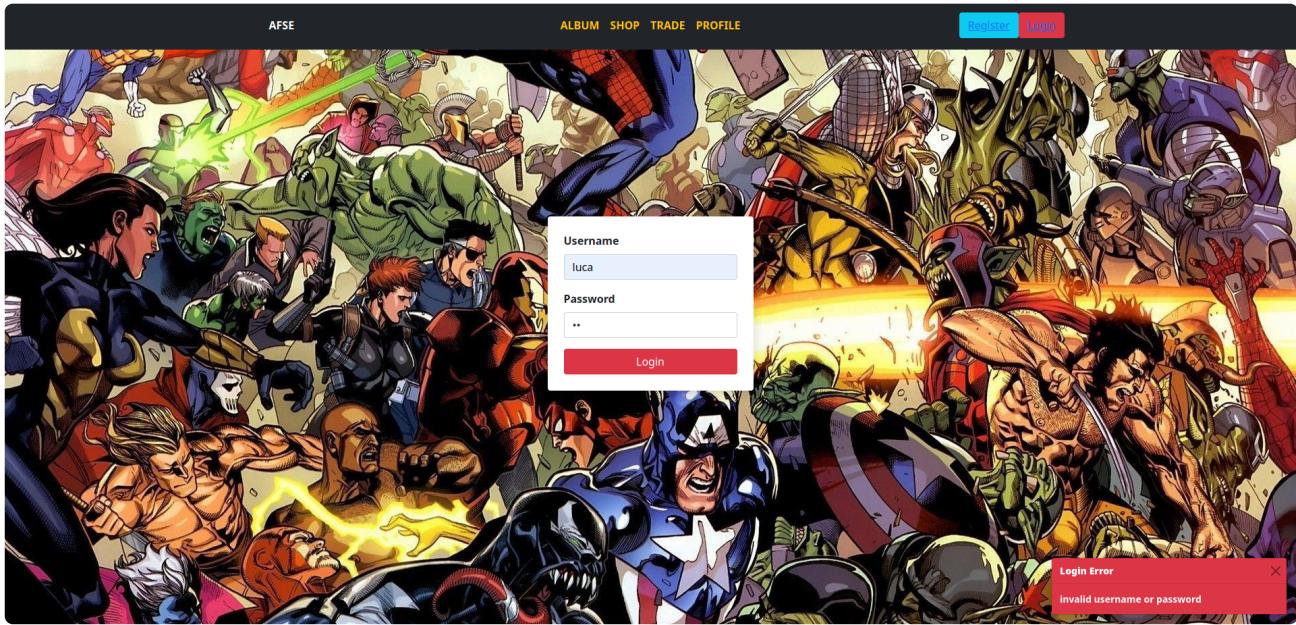
Se la registrazione va a buon fine si viene reindirizzati alla pagina di login da cui effettuare l'accesso.

## Login

Per effettuare il login viene richiesto l'inserimento di email e password:



In caso i dati inseriti siano errati comparirà un messaggio d'errore in basso a destra, realizzato in `ErrorToast.jsx` con il componente `Toast` di Bootstrap.



## Album

Pagina principale dell'applicazione, a cui si viene reindirizzati dopo il login, permette di visualizzare tutte le carte possedute dall'account a gruppi di 10:

Quando un'album contiene più di 10 carte sarà possibile scorrerle tramite il controllo centrale della barra di navigazione in basso, che mostra il numero di pagine.

Ad esempio, l'immagine qui sotto, mostra la seconda pagina di un album di 15 carte:

The screenshot shows a digital comic book album titled "Album of luca:". The interface includes a navigation bar at the top with links for "ALBUM", "SHOP", "TRADE", "PROFILE", and "Logout". Below the title, there are five cards representing comic book panels. The first panel features a character with blonde hair and a green face, labeled "Hulkling". The second panel is a "IMAGE NOT FOUND" placeholder. The third panel is another "IMAGE NOT FOUND" placeholder. The fourth panel is a "IMAGE NOT FOUND" placeholder. The fifth panel features a blonde woman with blue eyes, labeled "Sue Storm". Below the cards is a large, detailed illustration of a battle scene featuring Captain America, Iron Man, and other Marvel characters. At the bottom of the screen, there are buttons for "3 credits", "1 2", and "Buy Credits".

Infine, cliccando su una qualsiasi delle carte, si aprirà un modal contenente informazioni più dettagliate sul personaggio:

The screenshot shows a detailed character information modal for "Iron Man". The modal has a header "Iron Man" and a close button "X". It contains several sections: "Description" (with text about Tony Stark creating Iron Man), "Comics (2754)" (listing "A+X (2012) #2", "A+X (2012) #7", and "Adam: Legend of the Blue Marvel (2008) #1"), "Stories (4081)" (listing "3 of?", "Interior #982", and "Interior #984"), and "Series (687)" (listing "A+X (2012 - 2014)", "Adam: Legend of the Blue Marvel (2008)", and "Aero (2019 - 2020)"). To the right of the modal, there are two smaller images: one of a blonde woman labeled "Moonstone" and another of Iron Man. Below the modal, the main album interface is visible with other cards for "Shadowcat", "The Collector (Taneleer Tivan)", "Professor X (Ultimate)", "Susan Delgado", "Hulk (Marvel Zombies)", and "Iron Man". Navigation buttons "4 credits", "1 2", and "Buy Credits" are also present.

## Shop

La sezione centrale della pagina contiene una semplice tabella a scorrimento in cui vengono listate tutte le figurine dell'album, con la possibilità di venderle per 1 credito

cliccando il tasto SELL:

The screenshot shows the 'Shop' section of the AFSE website. At the top, there are tabs for 'ALBUM', 'SHOP', 'TRADE', and 'PROFILE'. A 'Logout' button is in the top right. Below the tabs, the word 'Shop' is centered. There are three cards listed for sale:

- Hulk (Marvel Zombies)** with a green 'SELL' button.
- Iron Man** with a green 'SELL' button.
- Hulkling** with a green 'SELL' button.

At the bottom of the page are buttons for '2 credits', 'BUY PACKET', and 'Buy Credits'.

Cliccando sul pulsante OPEN PACKET si acquisterà un pacchetto contenente 5 carte per 1 credito.

Le carte trovate verranno mostrate in un modal, il cui contorno è verde quando sono nuove e rosso in caso di duplicati:

The screenshot shows the 'Shop' section of the AFSE website with a modal window open. The modal contains five cards:

- Pet Avengers** (green border)
- Professor X (Ultimate)** (red border)
- Susan Delgado** (red border)
- Hulk (Marvel Zombies)** (red border)
- Gladiator (Kallark)** (red border)

Below the cards is a blue 'Back to album' button. The background of the shop page shows other cards and navigation buttons for '2 credits', 'BUY PACKET', and 'Buy Credits'.

## Trade

La pagina trade permette agli utenti di effettuare scambi complessi comprendenti massimo 3 carte e dei crediti.

Nella sezione della pagina si trova una tabella che mostra tutte le offerte esistenti:

The screenshot shows a 'Trade' section with two offers listed. The first offer is from 'pippo' to 'Iron Fist (USM)' for 'Gamora' with 0 credits. The second offer is from 'luca' to 'Black Widow' for 'Iron Man' and 'Shadowcat' with 0 credits. A red border highlights the second offer. At the top right, there are buttons for 'Logout', 'Fill', 'Delete', and 'Buy Credits'. The bottom navigation bar includes '2 credits', 'CREATE TRADE', and 'Buy Credits'.

La quarta colonna indica quale azione l'utente attualmente loggato può fare sullo scambio:

- se è il proprietario, viene mostrato un pulsante DELETE per cancellarlo
- se non è il proprietario, ma non possiede tutte le carte per completare l'ordine, la riga avrà il bordo rosso e il tasto FILL sarà disattivato
- se non è il proprietario e possiede tutte le carte per completare l'ordine, la riga avrà il bordo verde e il tasto FILL permetterà di accettare tale scambio

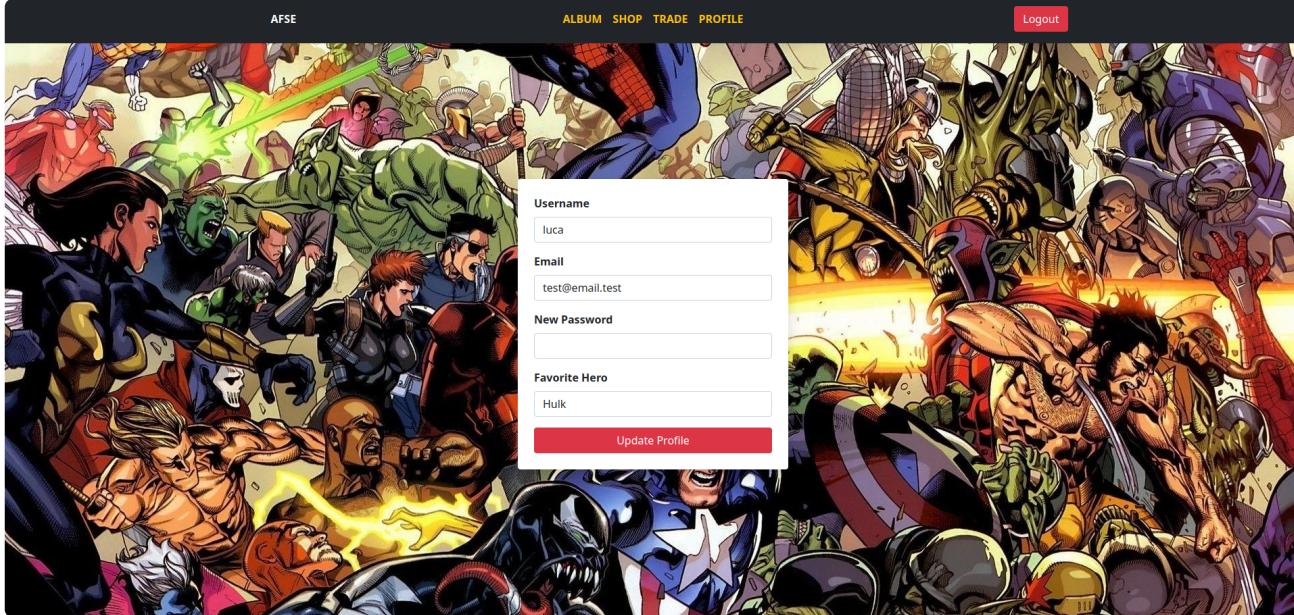
Per poter creare un nuovo ordine è necessario cliccare il tasto CREATE TRADE che si trova al centro della navbar bassa, che aprirà un modal dove sarà possibile selezionarne i dettagli:

The screenshot shows a 'Trade' section with a modal dialog for creating a new trade. The modal has fields for 'Credits Out (max 2)', 'Cards Out (from your album)', 'Credits In', and 'Cards In'. There are 'ADD', 'Close', and 'Create' buttons at the bottom. The background shows the same offers as the previous screenshot.

**⚠️ a ordine creato, le carte e i crediti in offerta vengono tolti per evitare il loro riutilizzo, per riaverli basta cancellare l'ordine**

## Profile

La pagina profile permette all'utente loggato di modificare i dettagli del proprio profilo:



## BackEnd

Il database si struttura su 3 collezioni:

- `users` : contengono i dati dei singoli utenti, la password è hashata tramite SHA256

```
_id: ObjectId('675fe61951216b9ba37b4f7c')
username: "luca"
email: "test@email.test"
password: "03ac674216f3e15c761eela5e255f067953623c8b388b4459e13f978d7c846f4"
favoriteHero: 1009351
credits: 2
```

- `albums` : contengono i dati sull'album di un'utente, identificato dal campo `ownerId`

```
_id: ObjectId('675fe84fc673318d7bc007ff')
ownerId: ObjectId('675fe84fc673318d7bc007fe')
descripton: null
  cards: Array (6)
    0: 1009189
    1: 1010844
    2: 1010673
    3: 1011495
    4: 1011292
    5: 1010957
```

- `orders` : contengono tutti gli ordini non completati

```

_id: ObjectId('67881799b180e6a77e0924f2')
creatorId : "675fe61951216b9ba37b4f7c"
fillerId : null
offer : Object
  cards : Array (2)
    0: 1009368
    1: 1009574
  credits : 0
request : Object
  cards : Array (1)
    0: 1009189
  credits : 0

```

Per gestire l'univocità dei campi `username` e `password` vengono utilizzati due **indici unique**, che generano un errore in caso una modifica alla collezione introduca dei dati duplicati:

Name, Definition, and Type	Size	Usage	Properties	Action
<code>_id</code>	36.0 KB	<1/min since Wed Jan 8 2025		
<code>username_1</code>	36.0 KB	<1/min since Wed Jan 8 2025	UNIQUE	⚡️ 🔴
<code>email_1</code>	36.0 KB	<1/min since Wed Jan 8 2025	UNIQUE	⚡️ 🔴

L'applicazione express utilizza il **pattern view-controller** dove:

- `controller` : includono tutte le operazioni sul database
- `view` : interfaccia API che prima sanifica i dati ed effettua autenticazione e poi chiama la relativa funzione del controller per aggiornare lo stato

L'autenticazione JWT si trova nella funzione `authenticateRoute()` di `middleware/auth.middleware.js`, che viene posta su tutti gli endpoint che la necessitano.

Data la lentezza nel recuperare i dati dei singoli eroi dalle API marvel, è presente un semplice **sistema di caching** che memorizza le nuove richieste e risponde con quelle memorizzate quando possibile, vedi implementazione:

```

async getCharacterById(id, fullData) {
  try {

```

```
let resp
// check if this id is cached
if (hero_cache.has(id)) {
    // no need to fetch marvel APIs, just return the
    // cached data
    resp = hero_cache.get(id)
} else {
    resp = await
baseMarvelRequest(`/characters/${id}`)
    resp = resp.data.results[0]
    // cache this id for future requests
    hero_cache.set(id, resp)
}
/// ...
```

**⚠ quando il server viene spendo la cache viene scritta su un file JSON, che se possibile verrà caricato al riavvio**