

- [Introduzione](#)
 - [Autore](#)
 - [Link ed informazioni utili](#)
- [Struttura dell'Applicazione](#)
 - [Front-End](#)
 - [Back-End](#)
 - [NODEJS](#)
 - [COS'E' NODEJS ED EXPRESS](#)
 - [FILE NODEJS](#)
 - [DATABASE MONGODB](#)
 - [ALBUMS](#)
 - [DESCRIZIONE](#)
 - [ATTRIBUTI](#)
 - [CARDS](#)
 - [DESCRIZIONE](#)
 - [ATTRIBUTI](#)
 - [USERS](#)
 - [DESCRIZIONE](#)
 - [ATTRIBUTI](#)
 - [EXCHANGES](#)
 - [DESCRIZIONE](#)
 - [ATTRIBUTI](#)
 - [EXCHANGES_CARDS](#)
 - [DESCRIZIONE](#)
 - [ATTRIBUTI](#)
- [Configurazione dell'applicazione](#)
- [Scelte implementative e features](#)
 - [Swagger JS](#)
 - [FILE SWAGGER.JS](#)
 - [INTERFACCIA GRAFICA SWAGGER](#)
 - [INSTALLAZIONE](#)
 - [Gestione codici HTTP](#)
 - [Esempi di Utilizo](#)
 - [Personaggi Marvel](#)
 - [Lingua](#)
 - [Tema](#)

Introduzione

Questo documento rappresenta la relazione del progetto "Album delle figurine dei super eroi", sviluppato nel contesto del corso "Programmazione Web e mobile" durante l'anno accademico 2024/2025.

Autore

Il progetto è stato realizzato da:

- [Stefano Damiani](#) (Matricola: 976496)

Link ed informazioni utili

- La pagina GitHub del progetto si trova a [questo link](#)

Struttura dell'Applicazione

Front-End

Front End: Il Front-End è la parte dell'applicazione che si occupa dell'interfaccia utente e dell'interazione con l'utente. Si concentra sulla progettazione e sull'implementazione dell'aspetto visivo dell'applicazione e sulla gestione delle interazioni utente.

All'interno della directory `/public/`, sono presenti i seguenti elementi principali:

- `/html/` Questa directory contiene i file HTML che vengono renderizzati da browser, determinando quindi interfaccia grafica dell'applicazione.
- `/css/`: Questa directory contiene i file di stile che definiscono l'aspetto visivo dell'applicazione. Alcuni dei file principali includono:
 - `/public/css/style.css` - Questo file definisce lo stile generale dell'applicazione
 - `/public/css/card.css` - Questo file definisce lo stile che riguarda le figurine e tutti i personaggi
- `/scripts/`: Questa directory contiene file JavaScript che gestiscono la logica del Front-End. Alcuni dei file principali includono:
 - `/public/scripts/theme.js` - Questo file si occupa di adattare il tema dell'applicazione a quello definito dal sistema dell'utente
 - `/public/scripts/navbar.js` - Questo file si occupa di definire la barra di navigazione sulle pagine in cui è richiesta e gestisce il caricamento della pagina di login e verifica quando l'utente è loggato.
 - `/public/scripts/login.js` - Questo file si occupa di tutte le operazioni necessarie al login dell'utente
 - `/public/scripts/register.js` - Questo file si occupa delle operazioni necessarie alla registrazione di un utente. Si occupa anche di caricare i dati dei supereroi per fare selezionare all'utente il suo preferito
 - `/public/scripts/marvel.js` - Questo file si occupa di gestire la maggior parte di ciò che riguarda i personaggi Marvel.

La suddivisione chiara tra file HTML, file CSS e file JavaScript consente una gestione efficiente del Front-End e garantisce un'esperienza utente di alta qualità.

Per ulteriori dettagli sull'implementazione del Front-End, si rimanda alle specifiche sezioni dei file e dei componenti menzionati sopra.

Back-End

Back End: Il Back-End è responsabile delle funzionalità e della logica dell'applicazione lato server. Esso comprende una serie di elementi chiave presenti nella nostra struttura di lavoro. Possiamo suddividere il backend in 3 sezioni principali

NODEJS

COS'E' NODEJS ED EXPRESS

Node.js ed **Express** costituiscono un binomio potente nell'ambito dello sviluppo web di applicazioni scalabili ed efficienti.

Node.js fornisce un ambiente runtime JavaScript server-side, ottimizzato per l'efficienza e la scalabilità.

Express, un framework web basato su Node.js, semplifica la creazione di applicazioni web, offrendo funzionalità come la gestione delle richieste HTTP e dell'autenticazione.

FILE NODEJS

- **/lib/api/docs/**: In questa directory sono presenti i file utilizzati per la gestione della documentazione pubblica delle API dell'applicazione, inclusi:
 - `swagger.js`
 - `swagger_output.js`
- **/config/**: Questa cartella contiene i file dedicati alla configurazione dell'applicazione, ad eccezione delle variabili d'ambiente. Al suo interno, sono presenti:
 - `prefs.js` - _Questo file viene utilizzato per caricare i valori inseriti dall'utente in modo che siano direttamente utilizzabili
- **/src/lib/**: La directory lib contiene tutte le funzioni Node.js utilizzate per le funzionalità degli endpoint. Inoltre Alcuni dei file e delle directory principali sono:
 - `/src/lib/database.js`
 - `/src/lib/login.js`
 - `/src/lib/marvel.js`
 - `/src/lib/register.js`
 - `/src/lib/utills.js`
- **app.js**: Questo file rappresenta il punto di ingresso principale dell'applicazione, contenente le istruzioni per l'avvio dell'app e la definizione degli endpoint.

La struttura ben organizzata del Back-End garantisce una gestione efficiente delle funzionalità server-side e contribuisce al corretto funzionamento dell'applicazione.

DATABASE MONGODB

Nel corso di sviluppo dell'applicazione, è stato fatto largo uso del database MongoDB. Qui di seguito, vengono presentate le collezioni sono state create e utilizzate per immagazzinare i dati essenziali dell'applicazione.

MongoDB: MongoDB è un database NoSQL (non relazionale), flessibile e scalabile, noto per la sua struttura orientata ai documenti. Un documento è un record dati in formato BSON (Binary JSON) che può contenere dati di varie forme e dimensioni. Ogni documento è organizzato in *collezioni*, offrendo flessibilità nella modellazione dei dati. Per lo sviluppo di questa applicazione è stato deciso di appoggiarsi ad una versione Hosted di MongoDB, fornita da Atlas ([Maggiori informazioni](#))

Per questa applicazione sono state utilizzate le seguenti collections:

- **albums:** Collezione che gestisce informazioni degli album di figurine
- **cards:** Collezione che gestisce le carte presenti all'interno di ogni album
- **users:** Collezione che gestisce i dati degli utenti
- **exchanges:** Collezione che gestisce gli scambi proposti dagli utenti
- **exchanges_cards:** Collezione che gestisce le carte proposte all'interno degli scambi

Di seguito viene riportata una descrizione delle collections e del loro schem

ALBUMS

DESCRIZIONE

La collezione *albums* ha lo scopo di raccogliere l'anagrafica degli album di figurine degli utenti

ATTRIBUTI

- **_id:** identificatore univoco di una album, di tipo ObjectId. È un campo obbligatorio per identificare univocamente un album nel db.
- **userId:** identificatore dell'utente creatore dell'album, di tipo ObjectId. È un campo obbligatorio e serve a linkare l'album al suo creatore.
- **name:** nome dell'album, di tipo stringa. È un campo obbligatorio e contiene il nome dell'album.

CARDS

DESCRIZIONE

La collezione *cards* è stata creata per salvare le carte trovate dagli utenti dentro i pacchetti e linkarle all'album in uso in quel momento

ATTRIBUTI

- **_id:** identificatore univoco dell'associazione Carta-Album, di tipo ObjectId. È un campo obbligatorio per identificare univocamente una carta nel database.
- **user_Id:** identificatore dell'utente proprietario della carta, di tipo ObjectId. È un campo obbligatorio e serve a linkare la carta al suo proprietario.
- **album_Id:** identificatore dell'album in cui la carta è inserita. Di tipo Stringa. È un campo obbligatorio e serve a linkare la carta all'album.
- **card_Id:** Id del personaggio fornito direttamente dalle API Marvel. Di tipo Numerico. Serve per identificare e collegare alle informazioni fornite da Marvel la carta

USERS

DESCRIZIONE

La collezione *users* è destinata a contenere i dati degli utenti all'interno dell'applicazione.

ATTRIBUTI

- **_id**: identificatore univoco di un utente, di tipo ObjectId. È un campo obbligatorio per identificare univocamente un utente nel database.
- **name**: nome dell'utente, di tipo stringa. È un campo obbligatorio e contiene il nome dell'utente.
- **surname**: cognome dell'utente, di tipo stringa. È un campo obbligatorio e contiene il cognome dell'utente.
- **username**: username dell'utente, di tipo ObjectId. È un campo obbligatorio e contiene la login, alternativa all'email, dell'utente.
- **email**: indirizzo email dell'utente, di tipo stringa. È un campo obbligatorio e contiene l'indirizzo email dell'utente.
- **password**: password dell'utente, di tipo stringa. È un campo obbligatorio e contiene la password dell'utente cifrata.
- **date**: data di nascita dell'utente, di tipo stringa. È un campo obbligatorio e contiene la data di nascita dell'utente.
- **superhero**: codice identificativo supereroe Marvel. È un campo obbligatorio e contiene il supereroe preferito dell'utente.
- **credits**: numero di crediti attualmente in possesso dell'utente. In fase di creazione corrispondono a 0. Campo obbligatorio

EXCHANGES

DESCRIZIONE

La collezione *exchanges* è stata creata per salvare gli scambi di figurine proposti dagli utenti.

ATTRIBUTI

- **_id**: identificatore univoco dello scambio, di tipo ObjectId. È un campo obbligatorio per identificare univocamente uno scambio nel database.
- **user_Id**: identificatore dell'utente proprietario della scambio, di tipo ObjectId. È un campo obbligatorio e serve a linkare lo scambio al suo proprietario.
- **album_Id**: identificatore dell'album da cui l'utente vuole scambiare. Di tipo Stringa. È un campo obbligatorio e serve a linkare lo scambio all'album.
- **requestedCard**: Id del personaggio richiesto per lo scambio, nella sintassi fornita direttamente dalle API Marvel. Di tipo Stringa. Serve per identificare la carta che viene acquisita quando un'altro utente accetta lo scambio

EXCHANGES_CARDS

DESCRIZIONE

La collezione `exchanges_cards` è stata creata per salvare le carte da donare durante uno scambio. È stata definita una tabella separata per permettere di dare più carte per una sola carta.

ATTRIBUTI

- **_id**: identificatore univoco dell'associazione Carta-Scambio, di tipo ObjectId. È un campo obbligatorio per identificare univocamente l'associazione nel database.
- **exchange_id**: identificatore univoco dello scambio, di tipo ObjectId. È un campo obbligatorio per identificare univocamente linkare la carta allo scambio.
- **user_id**: identificatore dell'utente proprietario della carta, di tipo ObjectId. È un campo obbligatorio e serve a linkare la carta e lo scambio al suo proprietario.
- **album_id**: identificatore dell'album in cui la carta doppia proposta è inserita. Di tipo Stringa. È un campo obbligatorio e serve a linkare la carta all'album, in modo che se la carta viene venduta nell'album lo scambio viene eliminato.
- **card_Id**: Id del personaggio fornito direttamente dalle API Marvel. Di tipo Numerico. Serve per identificare e collegare alle informazioni fornite da Marvel la carta proposta nello scambio

Configurazione dell'applicazione

Il progetto necessita di un file `.env` nella directory principale del dove sono contenuti i parametri necessari per il funzionamento. Il file `.env` è gestito attraverso il pacchetto npm `dotenv` che si occupa di popolare le relative variabili d'ambienti e renderne semplice l'utilizzo e accesso tramite JavaScript.

Un esempio di file env

```
# Server HOST and PORT
HOST='localhost'
PORT=666

# Parametri e Credenziali MongoDB
DB_USERNAME=your_database_user - #Pay attention, this is the App user, not the manageruser
DB_PASSWORD= your_database_user_password
DB_CLUSTER=link_to_your_cluster
DB_OPTIONS=retryWrites=true&w=majority&appName=your_app_name
DB_DBNAME = your_DB_NAME

# Parametri e Credenziali Marvel
BASE_URL=http://gateway.marvel.com/v1/
PUBLIC_KEY=YOUR_PUBLIC_KEY_FROM_MARVEL
PRIVATE_KEY=YOUR_PRIVATE_KEY_FOR_MARVEL
```

Scelte implementative e features

Swagger JS

Swagger: è un framework open-source per la progettazione, la creazione e la documentazione di API RESTful. La sua utilità si concentra sulla semplificazione del processo di sviluppo API, consentendo agli sviluppatori di definire chiaramente le specifiche delle API, testarle e generare automaticamente documentazione dettagliata.

Per la generazione dello swagger ho utilizzato il module [swagger-autogen](#).

Tramite la creazione di un file `swagger.js` (`/lib/api/docs/`) con una apposita configurazione e determinati commenti nella sezione degli endpoint, è possibile generare automaticamente una documentazione per gli endpoint.

è possibile visualizzare lo swagger generato all'endpoint **/api-docs**

FILE SWAGGER.JS

NB: Il codice riportato di seguito rappresenta un esempio molto vicino a quello utilizzato in questa applicazione!

```
import swaggerAutogen from 'swagger-autogen';
import { config } from "../../config/prefs.js";
const outputFile = './swagger-output.json';
const endpointsFiles = ['../../*.js', '../../app.js']; // This will match all
.js files in lib folder and its subfolders, plus app.js

const doc = {
  "info": {
    "title": "Marvel Characters API",
    "description": "API for AFSE (Album delle Figurine dei Super Eroi)",
    "version": "1.0.0"
  },
  host: `${config.host}:${config.port}`,
  basePath: "/",
  schemes: ['http'],
  consumes: ['application/json'],
  produces: ['application/json'],
  tags: [
    {
      "name": "fetch",
      "description": "Basic endpoint."
    },
    {
      "name": "users",
      "description": "Endpoints for the management of user data and related
operations."
    }
  ]
}
```

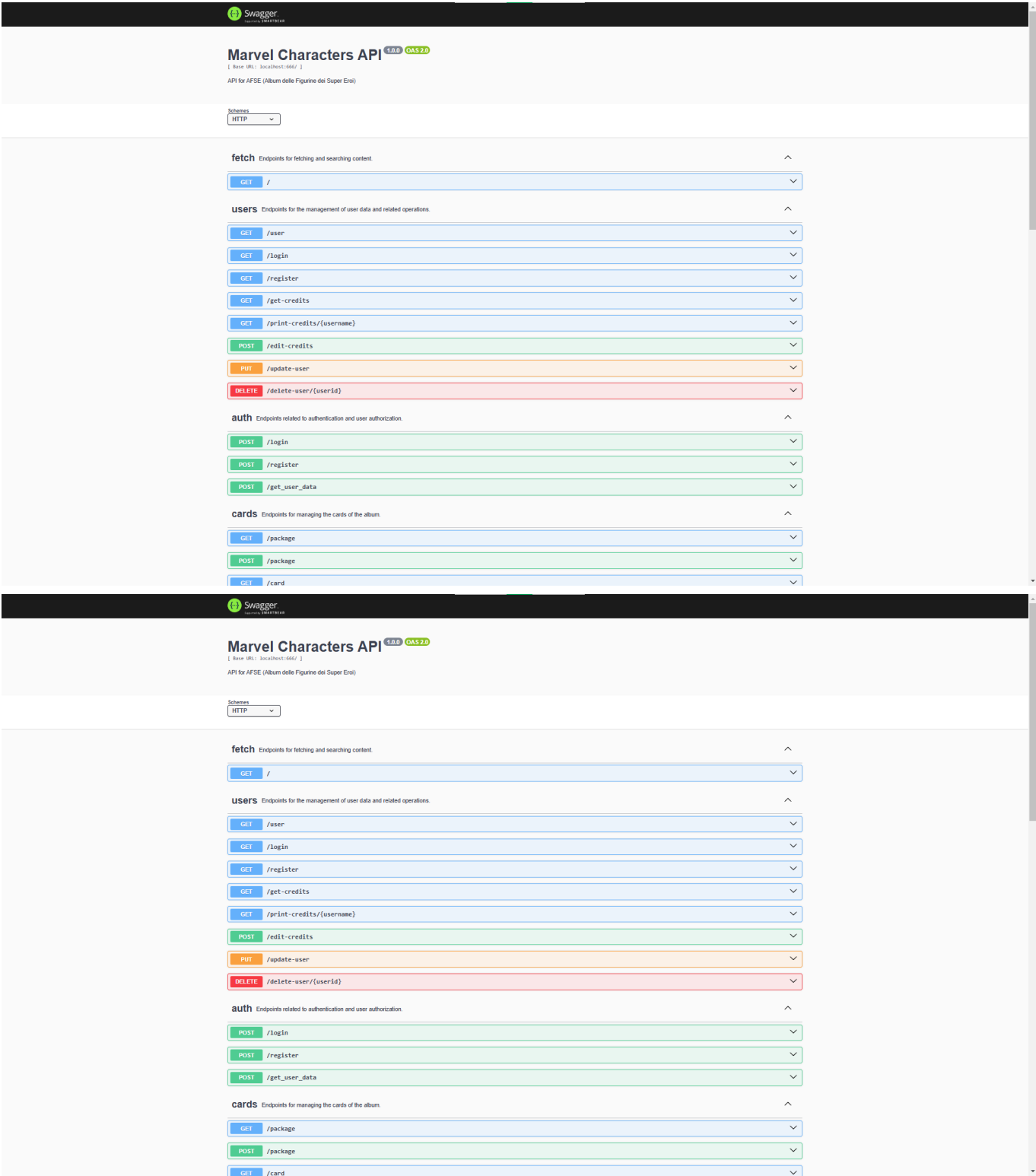
```

    },
    {
      "name": "auth",
      "description": "Endpoints related to authentication and user authorization."
    },
    {
      "name": "cards",
      "description": "Endpoints for managing the cards of the album."
    },
    {
      "name": "exchanges",
      "description": "Endpoint to manage exchanges."
    },
    {
      "name": "database",
      "description": "Endpoint to check the database connection."
    }
  ],
  definitions: {
    user: {
      _id: "ObjectId('64df73b31e5eda5eb868ddcd')",
      name: "John",
      username: "Jhonny",
      surname: "Doe",
      email: "jhonny@example.com",
      password: "hashed_password",
      credits: 100,
      superhero : 1011006
    },

    loggeduser: {
      $_id: "64df73b31e5eda5eb868ddcd",
      $username: "johndough",
      $email: "johndough@gmail.com",
      $name: "John"
    },
    loginrequest: {
      email: "johndough@gmail.com",
      username: "johndough",
      $password: "password"
    }
  }
};
const swagger = swaggerAutogen(outputFile, endpointsFiles, doc)

```

INTERFACCIA GRAFICA SWAGGER



INSTALLAZIONE

npm install --save-dev swagger-autogen
ulteriori informazioni sono presenti al link sopra riportato


Gestione codici HTTP


I codici HTTP sono standard utilizzati per indicare lo stato di una richiesta HTTP effettuata tra un client (spesso un browser web) e un server. Nell'applicazione, vengono ampiamente utilizzati alcuni di questi codici per comunicare lo stato delle richieste e delle risposte:


- **Codice 400 (BAD REQUEST):** Questo codice indica che la richiesta effettuata dal client è stata malformata o non valida. Viene utilizzato quando i dati inviati non corrispondono alle aspettative del server.
- **Codice 401 (UNAUTHORIZED):** Indica che l'accesso a una risorsa richiede l'autenticazione.
- **Codice 404 (NOT FOUND):** Indica che la risorsa richiesta non è stata trovata sul server.
- **Codice 500 (INTERNAL SERVER ERROR):** Questo codice indica un errore interno del server.
- **Codice 200 (OK):** Codice di successo. Indica che la richiesta è stata elaborata correttamente e che il server sta restituendo i dati richiesti al client.

Esempi di Utilizo

Pagina Iniziale



Find superhero | Album | Exchange | Packages | Credits: 1211.2 |  Stefano Damiani ▾



Welcome to SuperHeroes album community

Album delle Figurine dei Super Eroi is a network of marvel fans, just like you! If you want to discover something about your favourite characthes, this is the place for you!

What it is?

AFSE is a social group of people where you can create your card album, open card packages and trade your cards with other people.

Key Features

- Create and populate your cards album.
- Explore a vast selection of SuperHeroes.
- Experience the surprise of opening packages of cards.
- Connect with others enthusiasts and trade your cards.

Using Marvel APIs

AFSE utilizes Marvel's APIs to provide access to a wide range of characters and informations about them. You can learn more about Marvel's APIs [here](#).

About this project

This platform uses Node.js and Express.js to create a robust and scalable backend that can handle complex operations efficiently. It relies on custom API endpoints to facilitate seamless communication between the frontend and backend systems. You can explore the API documentation [here](#) to gain a deeper understanding of our endpoints and data structures.

About the Author

Album delle Figurine dei SuperEroi was developed by [Stefano Damiani](#)

Profilo

AFSE

[Find superhero](#) | [Album](#) | [Exchange](#) | [Packages](#) | Credits: 1211.2 | [Stefano Damiani](#)

User profile

Here you can manage your data, even delete your account if you want. **Deletion of the account is irreversible**

Email address
stefano.damiani@studenti.unimi.it

Username
SDAMIANI

The username length must be between 4 and 16 characters

You can change the password here.

Confirm password

The password must be at least 7 characters

Name
Stefano

Surname
Damiani

Date of birth
20/05/2001

Favourite superhero
Wolverine

Save

Delete account

Powered By

MARVEL

Express

MongoDB


Cerca superEroe - Non ancora trovato

AFSE

[Find superhero](#) | [Album](#) | [Exchange](#) | [Packages](#) | Credits: 1211.2 | [Stefano Damiani](#)

Favourite superhero
Thor

Thor



As the Norse God of thunder and lightning, Thor wields one of the greatest weapons ever made, the enchanted hammer Mjolnir. While others have described Thor as an over-muscled, oafish imbecile, he's quite smart and

Powered By

MARVEL

Express

MongoDB

11 / 17


Cerca superEroe - Trovato

AFSE

Find superheroAlbumExchangePackagesCredits: 1211.2Stefano Damiani

Favourite superheroes
Wolverine (Ultimate)

Wolverine (Ultimate)



Decades after participating in military airdrops with Captain America during WWII, James Howlett was abducted and experimented upon by a covert government unit, who bonded unbreakable adamantium

Series:

Ultimate Comics Wolverine (2013)

Ultimate Comics X-Men (2010 - 2013)

Ultimate Extinction (2006)

Ultimate Fantastic Four/Ultimate X-Men Annual (2008)

ULTIMATE FANTASTIC FOUR/X-MEN (2006)

Ultimate Galactus Trilogy (2007)

Ultimate Marvel Team-Up (2001 - 2002)

Ultimate Marvel Team-Up Ultimate Collection (2006)

Ultimate Nightmare (2004 - 2005)

Ultimate Power (2006 - 2006)

Ultimate Spider-Man (2000 - 2009)

Ultimate Spider-Man Vol. 12: Superstars (2005)

Ultimate Wolverine (2025 - Present)

Ultimate Wolverine Vs. Hulk (2005 - 2009)

Ultimate X-Men (2001 - 2009)

Ultimate X-Men Annual (2005 - 2006)

Ultimate X-Men MGC (2011)

Ultimate X-Men Ultimate Collection Book 1 (2006)

ULTIMATE X-MEN VOL. 14: PHOENIX TPB (2006)

ULTIMATE X-MEN VOL. 16: CABLE TPB (2007)

Comics:

Album

AFSE

Find superheroAlbumExchangePackagesCredits: 1211.2Stefano Damiani

Select your album

Select the album you want to see and put cards in it, if you open a package

If you don't have any album you can create one.


Album 1

Album 2

New album name


Create album

Vargas




Vargas claimed that mutants are deviations from the norm, an evolutionary dead end nature explored until it found the true path for the next step in mutant kinds development, the legitimate Homo Sapiens Superior.

Titania




Dependent and hopeless after it was discovered that she was not, in fact, Spider-Woman, Mary MacPherran encountered Doctor Doom, who offered her a chance to become superpowered in exchange for joining his army.

Wolverine (Ultimate)




Decades after participating in military airdrops with Captain America during WWII, James Howlett was abducted and experimented upon by a covert government unit, who bonded unbreakable adamantium

Wolverine (Ultimate)




Decades after participating in military airdrops with Captain America during WWII, James Howlett was abducted and experimented upon by a covert government unit, who bonded unbreakable adamantium

Ultimate Spider-Man (USM)



For the past year, Peter Parker has been saving New York City from villains as the masked hero, Spider-Man. Facing thugs and evil geniuses alike, he learns how to balance heroics with homework and friends. Despite

Ultimate Spider-Man (USM)



For the past year, Peter Parker has been saving New York City from villains as the masked hero, Spider-Man. Facing thugs and evil geniuses alike, he learns how to balance heroics with homework and friends. Despite

localhost566/card

Scambi - Pagina base

AFSE

Find superhero | Album | Exchange | Packages | Credits: 1211.2 | Stefano Damiani

Exchange card

In this page you will find a selection of available exchanges.

You will see exchanges that give you a card that you don't have, and send a card that you have at least twice in the selected album.

You can also sell cards for credits using the button below. A card values 0.2 credits.

If you proposed an exchange with a card that you sell, that exchange will be eliminated.

Sell duplicate cards

Create new exchange

Available exchanges

Card to Give	Card/s to Receive	Action
Ultimate Spider-Man (USM)	Boom Boom	Accept Exchange

Your exchanges

Card to Recieve	Card/s to Give	Action
She-Hulk (HAS)	Wolverine (Ultimate), Ultimate Spider-Man (USM)	Delete Exchange

Powered By

MARVEL

Express

MongoDB.

Scambi - Crea scambio

AFSE

Find superhero | Album | Exchange | Packages | Credits: 1211.2 | Stefano Damiani


Create exchange

In this page you can create an exchange.

You can only trade in a card that you have at least twice, and you can oly request a card that you don't have.

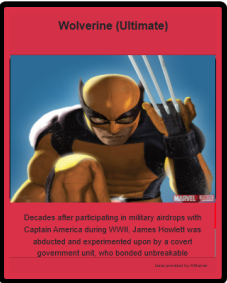
Sell duplicate cards

Ultimate Spider-Man (USM)



For the past year, Peter Parker has been saving New York City from villains as the masked hero, Spider-Man. Facing bugs and evil geniuses alike, he learns how to balance heroics with homework and friends.

Wolverine (Ultimate)



Decades after participating in military airings with Captain America during WWII, James Howlett was abducted and experimented upon by a covert government unit, who bonded unbreakable

Select the card that you want to obtain with this exchange

Requested superhero

X-23

Create exchange

Powered By

MARVEL

Express

MongoDB.

Scambi - Vendi carte


AFSE

Find superhero | Album | Exchange | Packages | Credits: 1211.2 | Stefano Damiani

Sell cards


In this page you can sell your duplicate cards.
A card values 0.2 credits.
If you proposed an exchange with a card that you sell, that exchange will be eliminated.

Wolverine (Ultimate)



Decades after participating in military airdrops with Captain America during WWII, James Howlett was abducted and experimented upon by a covert government unit, who bonded unbreakable claws to his hands.

Ultimate Spider-Man (USM)



For the past year, Peter Parker has been saving New York City from villains as the masked hero, Spider-Man. Facing thugs and evil geniuses alike, he learns how to balance heroics with homework and friends.

Powered By

MARVEL

JS Express

MongoDB

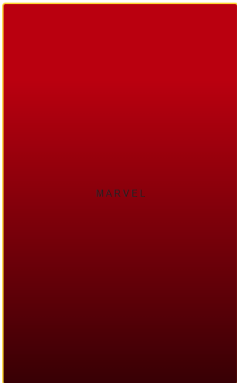
Pacchetti - Pacchetto non aperto

AFSE

Find superhero | Album | Exchange | Packages | Credits: 1209.2 | Stefano Damiani

Open a card package

In this package you will find 5 cards, it will cost 1 credit.
Current credits: 1209.2
You can open it only if you selected an album.



Powered By

MARVEL

JS Express

MongoDB

Pacchetti - Pacchetto aperto


AFSE

Find superhero | Album | Exchange | Packages | Credits: 1210.2 | Stefano Damiani

Open a card package

In this package you will find 5 cards, it will cost 1 credit.
Current credits: 1210.2
You can open it only if you selected an album.


Vulture (Adrian Toomes)



Adrian Toomes is a former electronics engineer who employs a special harness of his own design that allows him to fly and endows him with enhanced strength.

Image provided by Marvel. © 2024 MARVEL.


Wonder Man



Simon Williams agreed to undergo an experiment to give him superhuman powers, and Baron Zemo gave him the costumed guise of Wonder Man, warning Simon that he would die without further treatments from

Image provided by Marvel. © 2024 MARVEL.

Adam Warlock



Adam Warlock is an artificially created human who was born in a cocoon at a scientific complex called The Beehive.

Image provided by Marvel. © 2024 MARVEL.

Whiplash (Mark Scariotti)




IMAGE NOT FOUND

Filled with dreams of grandeur, Mark Scariotti was a brilliant young man with a bright future who graduated with top honors from college and immediately had a promising job at Stark International, but it all went

Image provided by Marvel. © 2024 MARVEL.

Glenn Talbot




IMAGE NOT FOUND

Major Glenn Talbot was member of United States Air Force, United States Army and Hulkbusters

Image provided by Marvel. © 2024 MARVEL.

OK

Crediti - Acquista crediti

AFSE

Find superhero | Album | Exchange | Packages | Credits: 1211.2 | Stefano Damiani

Buy credits

Here you can buy credits to buy packages.

You have 1211.2 credits

5€

10€

15€

5 Credits

12 Credits

18 Credits

20€

25€

50€

24 Credits

30 Credits

60 Credits

Buy

Apple Pay

Amazon Pay

G Pay

Powered By

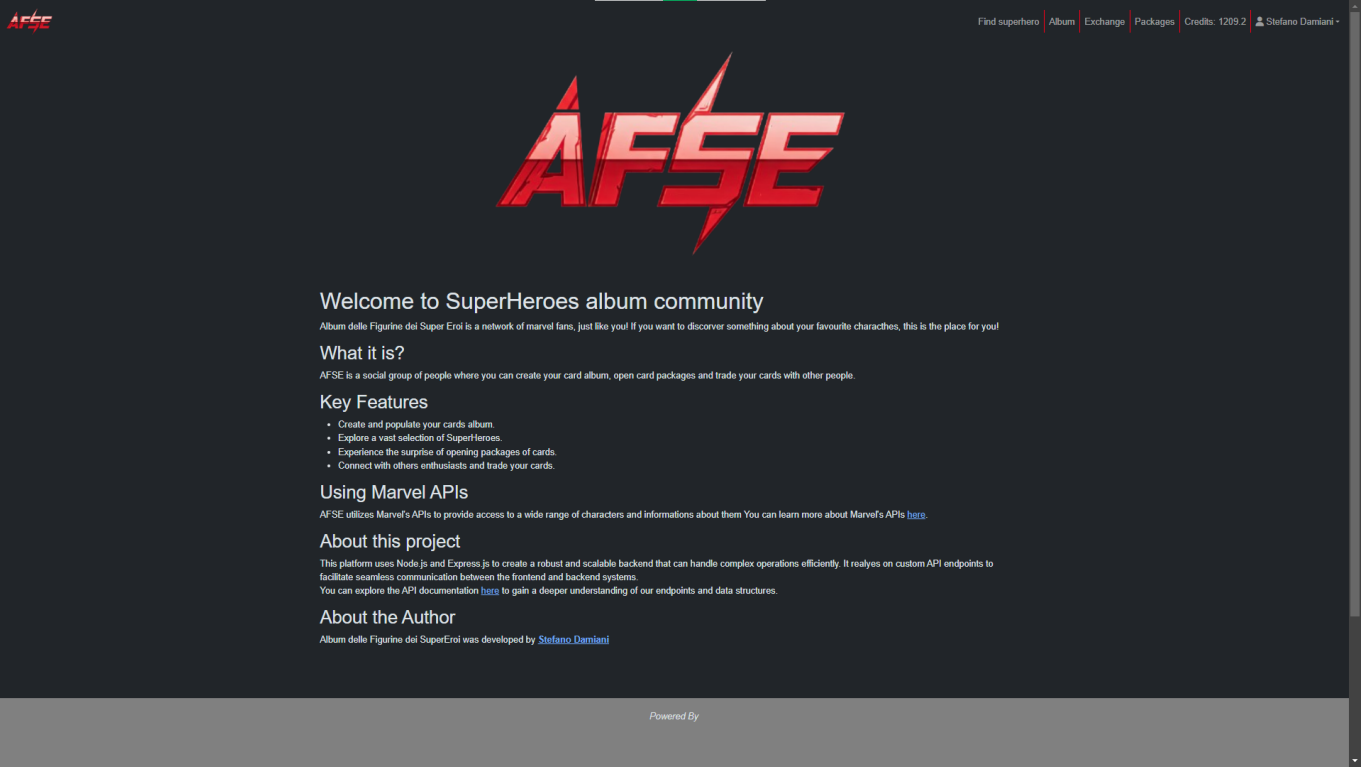
MARVEL

Express

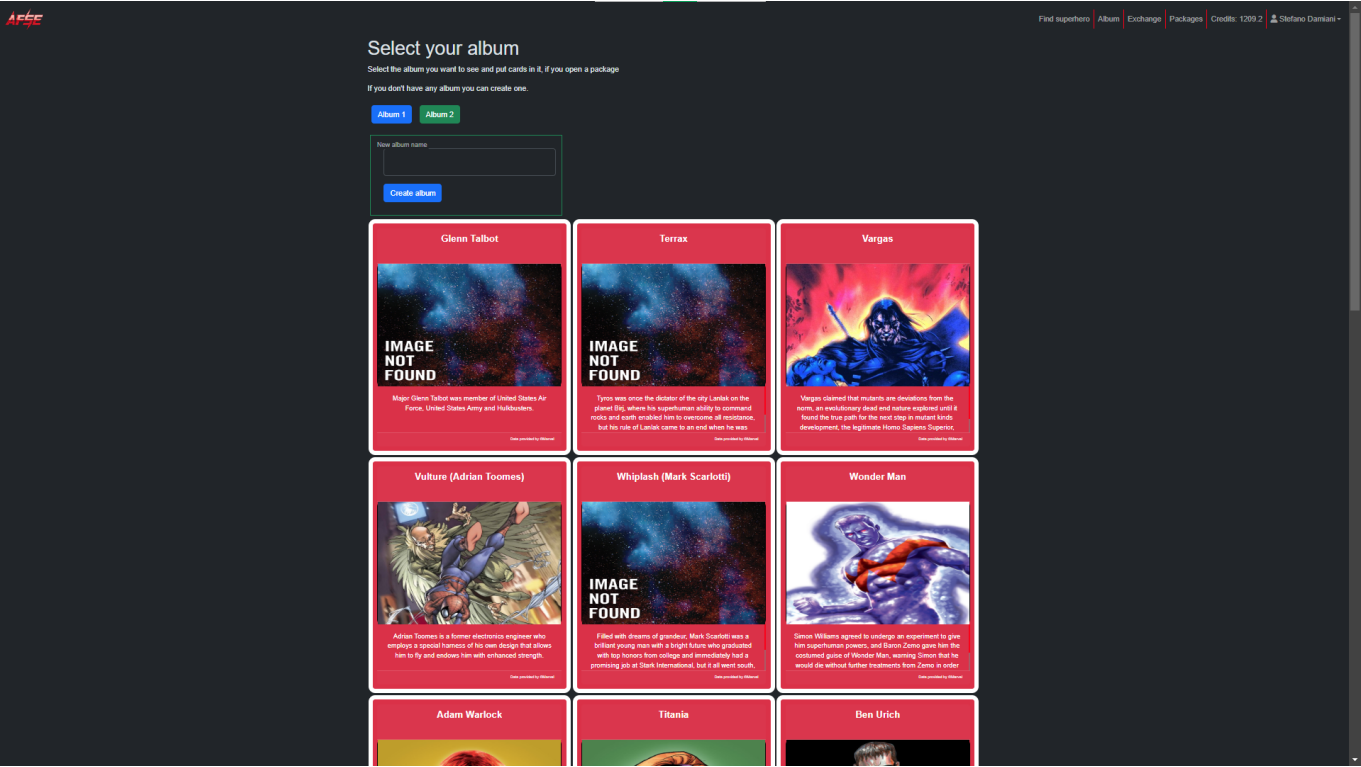
MongoDB.

15 / 17

Tema scuro - Pagina iniziale



Tema scuro - Album



Personaggi Marvel

I personaggi Marvel, per essere utilizzati all'interno dell'applicazione devono essere validati. Per essere validi essi devono obbligatoriamente avere un nome, una descrizione e un'immagine, che può essere quella di default. Tutti i personaggi non validi non vengono presi in considerazione dall'applicazione.

Lingua

La scelta di utilizzare la lingua inglese, come standard di programmazione, è ampiamente diffusa nell'industria del software ed è guidata principalmente dal desiderio di aderire allo standard internazionale. Questo standard è anche noto nella community di programmatori come **"English-based programming"**.

Adottare questa convenzione ha numerosi vantaggi, in quanto rende il codice più leggibile e comprensibile per un pubblico globale di sviluppatori. La scelta è derivata anche dalla decisione di rendere il progetto, dopo la conclusione, un'applicazione open-source. La scelta della licenza, che è stata come MIT, rispecchia questa scelta.

Tema

L'applicazione è disponibile sia con un tema chiaro e sia con un tema scuro. L'applicazione si adatta automaticamente alle preferenze dell'utente, come visto sopra.