

UNIVERSITÀ DEGLI STUDI DI SALERNO

DIEM - Dipartimento di Ingegneria dell'Informazione ed Elettrica e Matematica
Applicata



Corso di Laurea Triennale in Ingegneria Informatica

Progetto - Tecnologie Web **Outer Space**

Gruppo 05:
Alessio Leo - 0612707279
Emanuele Tocci - 0612707488
Rossella Pale - 0612707284
Claudia Montefusco - 0612707404

ANNO ACCADEMICO 2024/2025

Indice

I Executive Summary	3
1 Tema del progetto	3
2 Obiettivo del progetto	3
II Progettazione	4
3 Organizzazione dei file	4
4 Scelte di design	4
5 Wireframe	5
6 Configurazione del Database	5
7 Considerazioni sui redirect (.htaccess ed index.php)	6
8 Fogli di stile esterni	6
III Descrizione del sito	7
9 Pagine principali	7
9.1 Homepage	7
9.2 Catalogue	8
9.3 Pagina per il corpo celeste	9
9.4 History Orders	10
9.5 Ticket	11
9.6 Space History	12
9.7 SignUp	13
9.8 About Us	14
9.9 Contact	14
9.10 Error e Conferma	15
10 Componenti Principali	15
10.1 Login	15
10.2 Cart	15
10.3 Trip Dates	16
10.4 Header	17
10.5 Footer	18
IV Partecipazione al progetto	18
V Istruzioni generali	20

INDICE

11 Installazione delle dipendenze	20
11.1 Installare Stripe	20
11.2 Installare PHPdotenv	20
12 Configurazione della Chiave Segreta di Stripe	20
12.1 Soluzione alternativa senza usare PHPdotenv	21
13 Simulare il pagamento	21

Parte I

Executive Summary

Outer Space è un sito web che simula un tour operator interspaziale, permettendo agli utenti di esplorare un catalogo cosmico e prenotare viaggi verso pianeti, galassie, lune e altri corpi celesti della Via Lattea e dello spazio profondo.

1 Tema del progetto

Il tema assegnato per questo progetto è quello dell'**esplorazione spaziale** e del **turismo interstellare**. Outer Space si propone come una piattaforma per simulare l'acquisto di viaggi verso spazio, offrendo agli utenti la possibilità di *scoprire le meraviglie dell'universo* attraverso un'esperienza virtuale coinvolgente e di *informarsi su dati astronomici* e scoperte scientifiche recenti.

2 Obiettivo del progetto

L'obiettivo principale del nostro sito è quello di ispirare e affascinare gli utenti, accendendo la loro passione per l'**esplorazione dello spazio**. Attraverso un design e contenuti interattivi, il sito vuole educare sulle meraviglie dell'universo, promuovere la conoscenza scientifica e offrire un'esperienza di navigazione, capace di trasportare gli utenti in un viaggio tra le stelle.

L'utente ha la possibilità di *prenotare biglietti per viaggi nello spazio* utilizzando un sistema di pagamento **virtuale** e simulato (stripe), accedere a contenuti esclusivi sullo spazio, come curiosità, e storie di esplorazioni spaziali.

Parte II

Progettazione

3 Organizzazione dei file

Il codice sorgente é stato organizzato su diversi livelli al fine di garantire la **massima modularitá** possibile. La suddivisione dei file segue la seguente gerarchia:

```
root
├── db ..... Database
├── dependencies ..... Dipendenze PHP
├── docs ..... Documentazione
├── src ..... Codice sorgente
│   ├── assets ..... Risorse condivise
│   ├── backend ..... Gestione backend
│   ├── components ..... Sezioni modulari incorporabili tramite PHP
│   ├── pages ..... Pagine del sito
│   └── .env ..... Ambiente dotenv
└── .htaccess
└── index.php
```

In ogni sottocartella é stato inserito un file `README.md` per chiarirne il contenuto.

4 Scelte di design

Il progetto ha preso forma con la creazione dei **wireframe**, utilizzati come bozza iniziale per la struttura del sito. Grazie a **Figma**, sono stati realizzati **mockup** dettagliati che hanno delineato la struttura visiva e l'esperienza utente delle varie pagine. Una volta perfezionati questi prototipi, sono stati collegati tra loro per assicurare un flusso di navigazione coerente e intuitivo.

Per definire chiaramente l'identità del progetto, è stato scelto un **titolo** suggestivo, Outer Space, che incarna perfettamente il tema esplorativo e futuristico del sito. È stata selezionata una **paletta cromatica** dominata da tonalità di viola e nero, ispirate alla vastità e al mistero dello spazio.

5 Wireframe

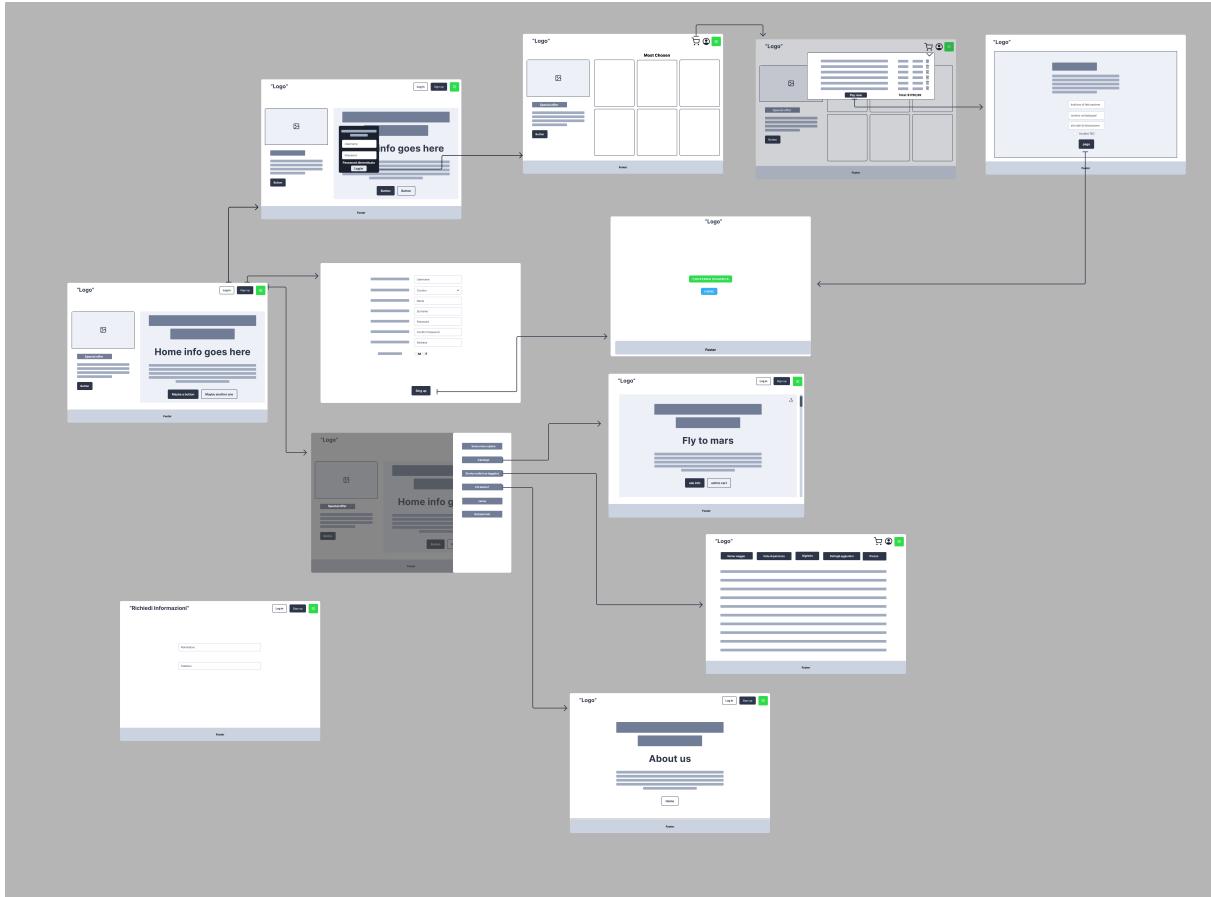


Figura 1: Wireframe-Wireflow

6 Configurazione del Database

Per la configurazione del database, utilizziamo le seguenti credenziali:

- **Porta:** 5432
- **Username:** www
- **Password:** tw2024
- **Nome del Database:** gruppo05

Inoltre, è stato creato anche un utente già inserito nel database con ordini effettuati:

- **Username:** ale
- **Password:** 30

7 Considerazioni sui redirect (.htaccess ed index.php)

Il file `.htaccess` si occupa di effettuare il redirect ad una pagina di errore nel caso in cui l'utente richieda una pagina o una risorsa non esistente. Il file `index.php` invece, effettua il redirect verso la homepage del sito.

Bisogna tuttavia considerare il fatto che, il tipo di redirect effettuato, è dipendente dalla configurazione del webserver nonché dalla lunghezza dell'URL cercato.

I link utilizzati in entrambi i file infatti sono assoluti e fanno riferimento alla `DocumentRoot` configurata nel file `httpd.conf` di **Apache**.

Il redirect è stato testato con una configurazione del genere:

```
DocumentRoot "/Users/emanueleletocci/public_html"  
<Directory "/Users/emanueleletocci/public_html">
```

E la cartella di questo specifico progetto "tw-project", inserita direttamente all'interno della `public_html`:

```
public_html/  
└ tw-project/  
    └ assets/  
    └ db/  
    └ src/  
    └ .htaccess  
    └ index.php
```

Se si dovesse rinominare la cartella o inserirla in una cartella già presente nella `public_html` è necessario modificare anche i percorsi assoluti presenti nei file `.htaccess` ed `index.php`.

8 Fogli di stile esterni

All'interno della directory `src/assets/css` sono presenti 2 fogli di stile globali: `textGlobal.css` e `CssReset.css` che vengono importati da tutti gli altri stili css presenti sul sito. Il primo definisce le **impostazioni generali** dell'intero progetto, a partire dai fonts usati fino ad arrivare ai colori. Presenta anche alcune **utilities** che facilitano la programmazione. Il **reset** invece viene usato per uniformare lo stile su tutti i browser andando a sovrascrivere lo stile di default di essi.

Parte III

Descrizione del sito

9 Pagine principali

Il sito si compone delle seguenti pagine principali:

- **Homepage:** pagina di 'benvenuto'.
- **Catalogue:** lista delle destinazioni disponibili.
- **Pagine per il corpo celeste:** dettagli sulle destinazioni.
- **History Orders:** cronologia degli acquisti.
- **Spatial History:** storia dell'esplorazione spaziale.
- **Ticket:** gestione dei biglietti acquistati.
- **SignUp:** gestione la registrazione dell'utente.
- **About Us:** informazioni sull'azienda.
- **Contact:** gestisce le richieste di supporto.
- **Error e Conferma:** pagine di errore e conferma operazioni.

9.1 Homepage

L'HomePage è il '**benvenuto**' ed **invito agli utenti ad esplorare l'Universo**. I punti salienti di questa pagina si concentrano sul focus, sull'avventura e sull'esperienza unica di esplorare l'universo con viaggi personalizzati. Tramite un **FAQ** si cerca di chiarire dubbi e rassicurare i potenziali clienti.

La Homepage mostra contenuti diversi a seconda del tipo di utente, come per la gestione del coupon:

- Se l'utente **non è loggato** viene mostrato del contenuto che invita a non perdere l'offerta per il primo acquisto con un button con value 'Get Deal!'
- Se l'utente **è loggato** e:
 - **non ha effettuato acquisti**, viene mostrato un button con codice sconto.
 - **ha effettuato almeno un acquisto**, viene mostrato contenuto diverso che invita l'utente a iscriversi alle newsletter.

9.2 Catalogue

La pagina è suddivisa in due sezioni principali. La prima sezione permette agli utenti di cercare destinazioni interstellari utilizzando vari filtri come "Where", "Budget" e "Type". La seconda sezione mostra il catalogo delle destinazioni, suddivise in categorie come pianeti, galassie, lune e nebulose, con immagini, nomi e prezzi.

Utilizza PHP (in particolare lo script getAllUniqueTrips.php) per recuperare i dati del viaggio dal database e generare l'HTML in modo dinamico.

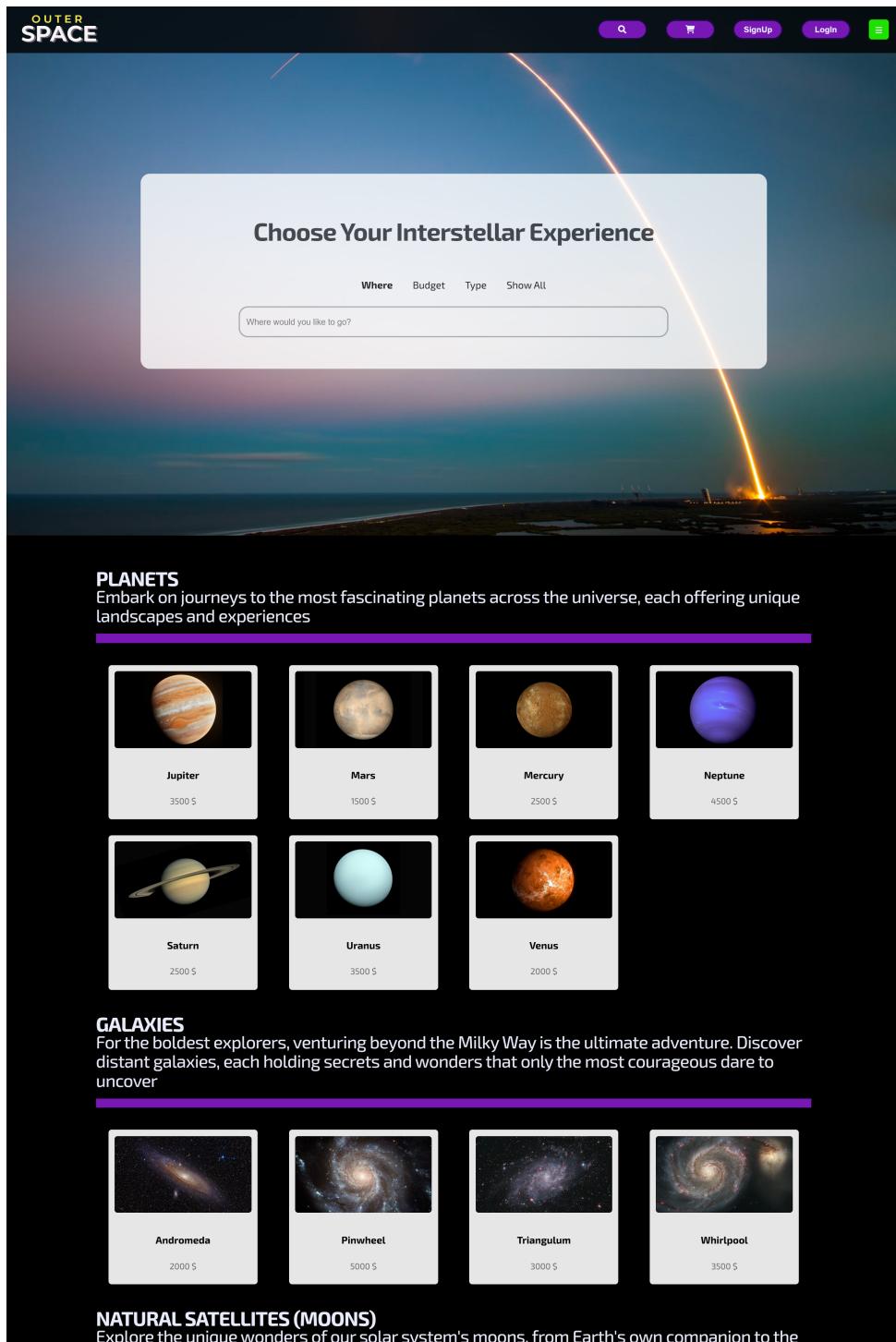


Figura 2: Catalogue

Il tutto è gestito con **AJAX**.

9.3 Pagina per il corpo celeste

Quando un utente clicca su una delle destinazioni nel catalogo, viene reindirizzato a una pagina che mostra le informazioni dettagliate sul corpo celeste selezionato. In particolare, per i pianeti e le lune viene utilizzato l'**API Solar System OpenData**.¹

Ciascuna pagina include il file **tripdates.php**, che implementa un popup per visualizzare le date disponibili per il viaggio selezionato. Il popup mostra una tabella con le **date di partenza, ritorno, la posizione del portospaziale** e un pulsante per **aggiungere il viaggio al carrello**. JavaScript gestisce l'apertura e chiusura del popup, e la logica per aggiungere viaggi al carrello tramite **AJAX**. In questo modo, l'utente può facilmente visualizzare le date disponibili e aggiungere viaggi al carrello senza ricaricare la pagina.

Nella pagina inoltre vengono visualizzati **suggerimenti** per altre destinazioni correlate in un carosello nella parte inferiore della pagina, per incentivare ulteriori esplorazioni da parte dell'utente.

¹Fornita da le-systeme-solaire.net. Questa API consente di accedere a dati scientifici completi relativi a pianeti, lune, e altri corpi celesti del sistema solare. Le informazioni recuperate includono caratteristiche fisiche come massa, raggio, orbita e altre proprietà rilevanti. Le chiamate API vengono gestite tramite richieste AJAX

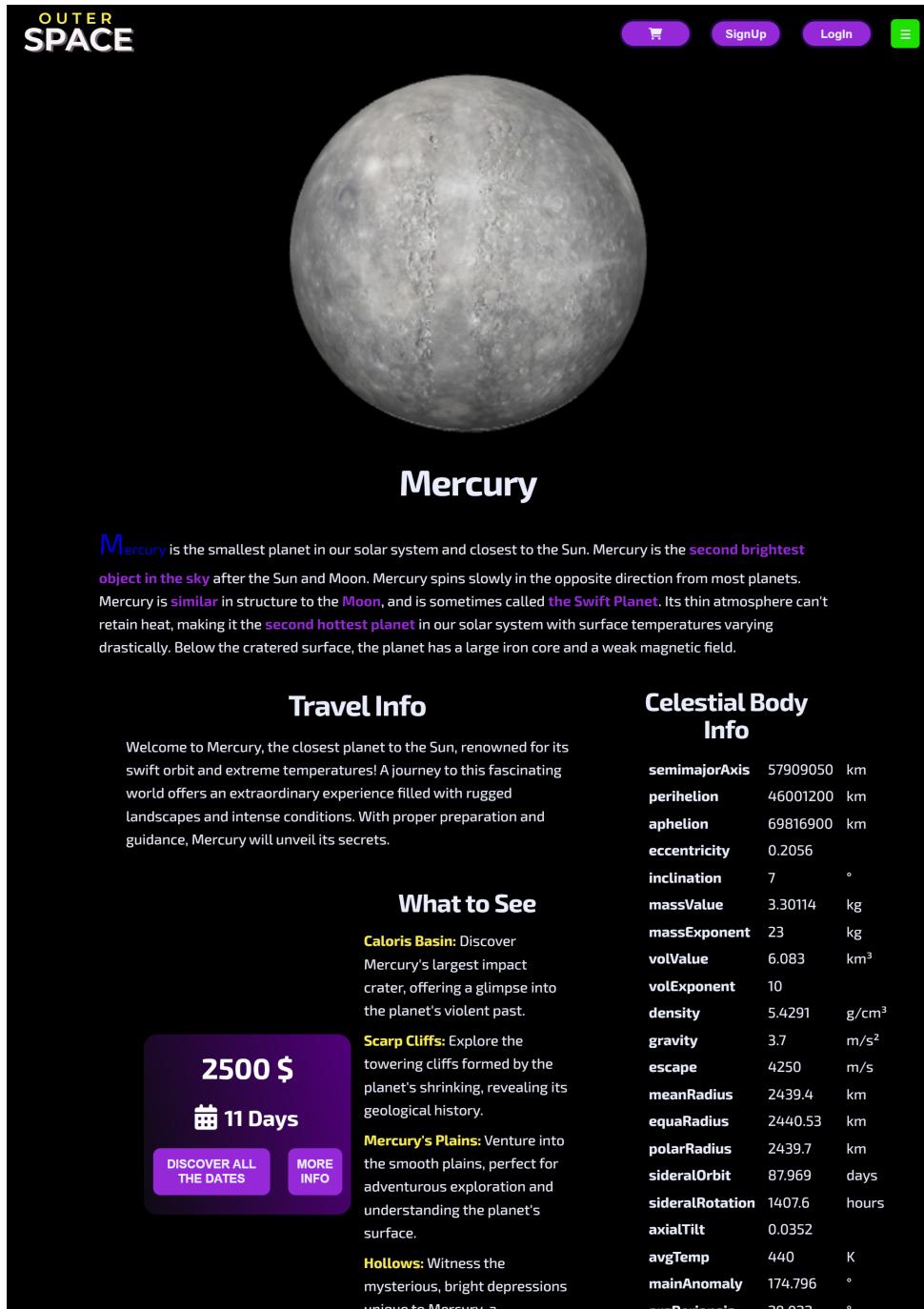


Figura 3: Pagina per il corpo celeste Mercurio

9.4 History Orders

La pagina History Orders permette agli utenti loggati di **visualizzare la cronologia dei propri ordini**. Solo una volta effettuato l'accesso nel menu laterale viene visualizzato il collegamento alla pagina. L'utente può vedere un elenco dettagliato degli acquisti passati. Accanto a ogni ordine, viene fornito un link per visualizzare un biglietto dettagliato relativo all'acquisto.

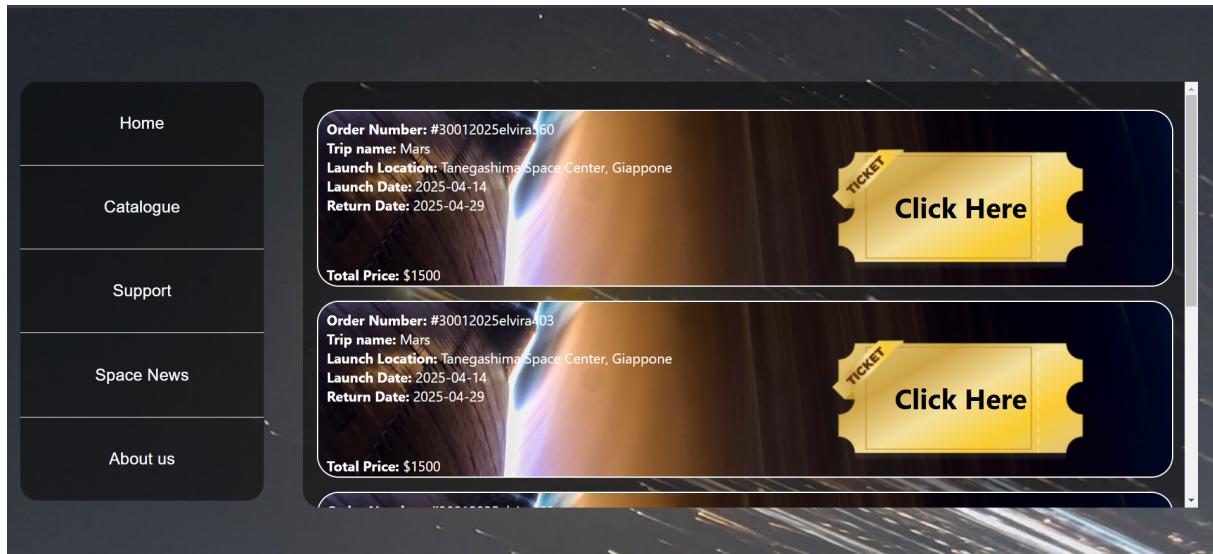


Figura 4: Pagina History Orders

La pagina utilizza PHP per gestire le sessioni utente, recuperare i dati degli ordini dal database e generare contenuto dinamico. Le query SQL, effettuate nello script GetInfo.php, vengono utilizzate per interrogare il database e recuperare i dati degli ordini.

9.5 Ticket

La 'Your floppy ticket' permette agli utenti di vedere i **dettagli del loro viaggio selezionato**.

Gli utenti vengono indirizzati a questa pagina dalla sezione History Orders quando cliccano su uno dei biglietti visualizzati.

Utilizzando PHP, la pagina recupera i dati dell'ordine da una richiesta GET e gestisce le sessioni utente.

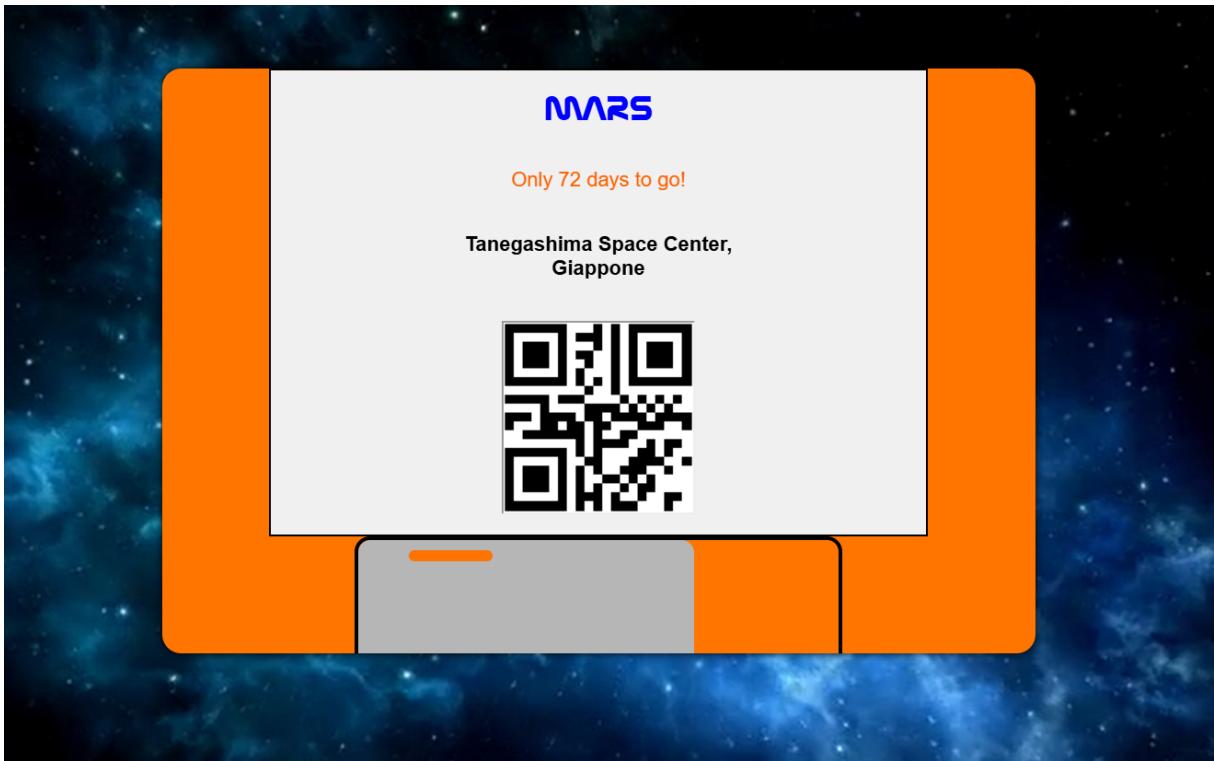


Figura 5: Pagina Ticket

La pagina genera un **codice QR** per il numero dell'ordine utilizzando l'API `QRServer`. Questo permette agli utenti di avere un rapido accesso al proprio ordine scansionando il codice QR. Il codice QR viene generato costruendo un URL che include i dati del numero dell'ordine e specifica le dimensioni del codice QR. Questo URL viene poi inserito in un elemento `iframe` che carica l'immagine del codice QR generato dall'API.²

9.6 Space History

Questa pagina offre una **panoramica sui pianeti** con un coinvolgente slider fotografico, fornendo dettagliate informazioni sui **pionieri dello spazio** e notizie storiche sui **voli spaziali**.

L'utente ha la possibilità di visualizzare le **notizie spaziali più recenti alla data inserita**. Dopo l'inserimento nel form della data, l'API di `Spaceflight News` viene chiamata tramite richieste `AJAX` per recuperare articoli storici sui voli spaziali.³ La pagina presenta un form per l'iscrizioni alle **newsletter**. Per ciascun email inserita ne viene verificata la validità e controllata la sua esistenza nel database prima di inserirla.

²Questa API ci consente di generare dinamicamente un codice QR che contiene il numero dell'ordine.

³l'API `Spaceflight News` consente di ottenere articoli pubblicati dopo una data specifica, ordinati per data di pubblicazione. Le informazioni recuperate includono il titolo dell'articolo, la data di pubblicazione, il contenuto e il link all'articolo completo.

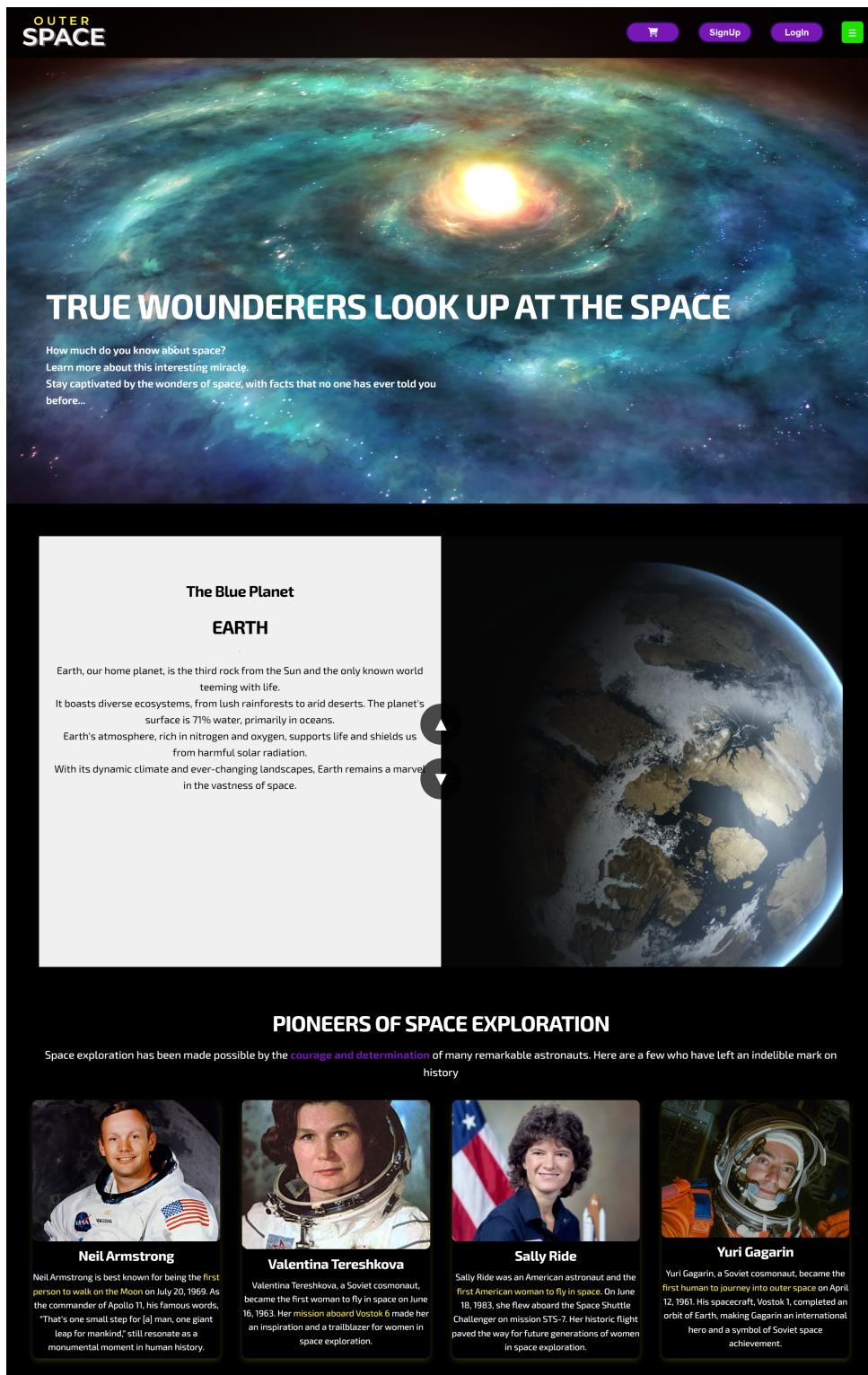


Figura 6: Pagina Space History

9.7 SignUp

La pagina gestisce la registrazione degli utenti.

Questa pagina presenta un **sticky form** di registrazione per l'utente. Gli utenti possono inserire i loro dettagli personali, come **nome**, **cognome**, **email**, **username**, **password**, **data di nascita**, **paese** e **genere**.

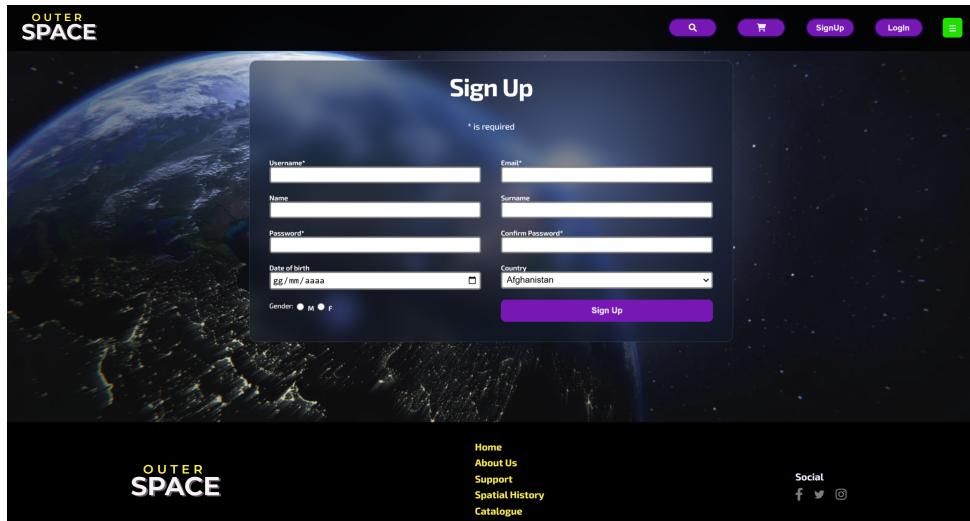


Figura 7: Pagina SignUp

Il form raccoglie i dati dell’utente e li invia al server (**POST**), il quale si occupa del **controllo di validità**. In caso di successo, l’utente viene reindirizzato a una pagina di conferma, in caso di errore viene mostrato un banner che invita l’utente a ricompilare il form.

9.8 About Us

About Us è la pagina volta alla descrizione degli **obiettivi** e degli **scopi** del gruppo nell’intento di realizzare al meglio le **pre condizioni** predisposte da noi membri, senza mai togliere l’attenzione al **contesto** su cui si basa il nostro progetto. Una particolare attenzione nel ricordare che il sito non solo vuole **informare**, ma anche *ispirare* gli utenti riguardo alla possibilità di esplorare lo spazio.

9.9 Contact

Contact è la pagina volta alla possibilità di **rivolgersi domande** relative ai viaggi, prenotazioni, segnalazione di problemi e di ricevere supporto agli utenti che hanno bisogno di assistenza.

Il form raccoglie l’*oggetto* e il *messaggio* del supporto, quindi apre il client di posta elettronica predefinito con i campi ”Oggetto” e ”Corpo” già compilati. Dopo l’invio dell’email, la pagina viene ricaricata per garantire che il modulo sia pronto per un nuovo utilizzo.

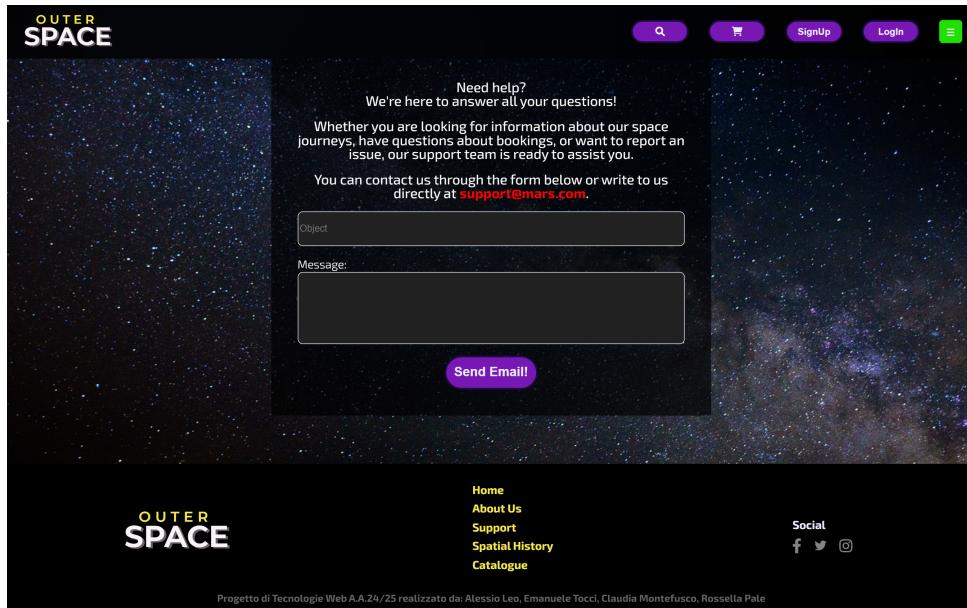


Figura 8: Pagina Contact

9.10 Error e Conferma

Il sito presenta una pagina per la **conferma dinamica**, che visualizza un messaggio di conferma personalizzato in base al tipo di operazione completata (registrazione o checkout), e una pagina di errore nel caso di **errato pagamento**.

La pagina `Conferma.php` consulta i parametri nell'URL (`confirmsignup` per la registrazione e `confirmcheckout` per l'acquisto) e mostra un messaggio di conferma personalizzato. Nel caso di conferma di acquisto viene inoltre svuotato il carrello tramite lo script `gestioneAcquisti.php` che è incluso in `Conferma.php`.

10 Componenti Principali

Oltre alle pagine principali, il sito include diversi pop-up interattivi e componenti modulari come il footer e l'header.

10.1 Login

Il pop-up permette agli utenti, che si sono registrati almeno una volta, di **accedere al sito così da ottenere contenuti personalizzati**. Il login **dipende dalla posizione geografica dell'utente...** in particolare è limitato ai soli paesi ammessi tramite l'utilizzo della **geolocalizzazione di HTML5**.

10.2 Cart

Il pop-up cart permette di visualizzare tutti gli elementi aggiunti al **carrello**. Per ogni viaggio viene visualizzato la **destinazione**, la data di **partenza** e il **costo** del viaggio, oltre al pulsante per la **rimozione**. Solo alla fine è mostrato il prezzo totale e il pulsante di acquisto nel caso di utente loggato.

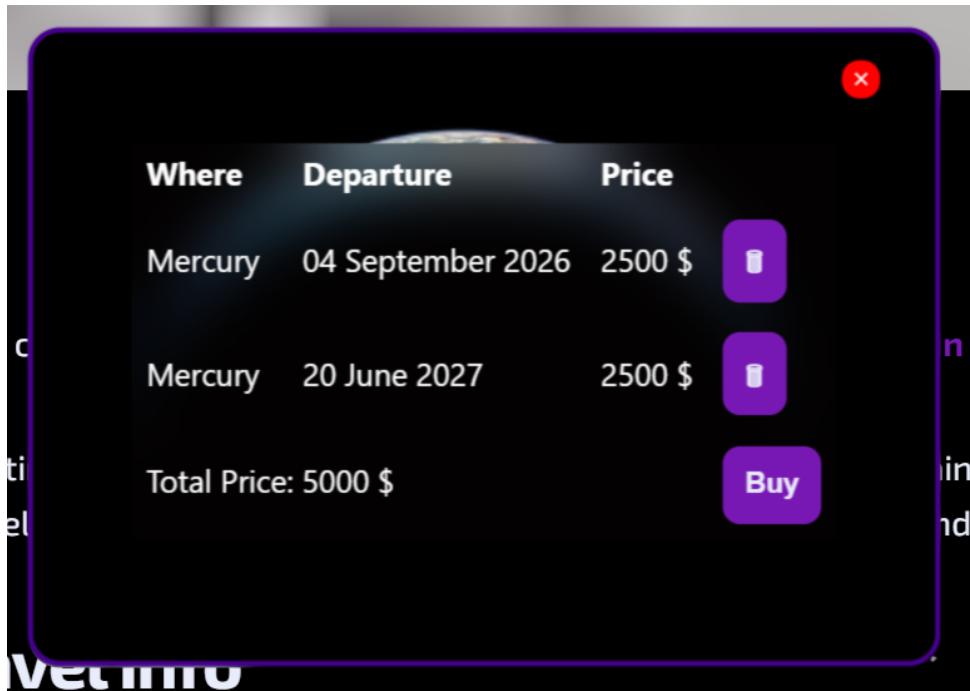


Figura 9: Pop-up Cart

L'utilizzo dei **cookies** per la memorizzazione dei dati del carrello, consente un aggiornamento dinamico della lista degli articoli attraverso chiamate **AJAX**, senza ricaricare la pagina. Il totale del carrello viene costantemente aggiornato, e l'utente può procedere al pagamento una volta soddisfatto con il contenuto del carrello solo se **loggato**.

Quando l'utente clicca su "Buy", il carrello viene passato al file **Checkout.php**, che gestisce la creazione di una sessione di pagamento con **Stripe**. I dati del carrello vengono codificati in **JSON**, inviati a Stripe, che poi reindirizza l'utente all'URL di checkout. Se il pagamento va a buon fine, l'utente viene indirizzato alla pagina di conferma del pagamento; in caso contrario, alla pagina di errore.

10.3 Trip Dates

Questo componente permette di visualizzare le date disponibili per il corpo celeste selezionato, sotto forma di pop-up.

L'aggiunta al carrello viene gestita tramite **AJAX** così da evitare il reload della pagina, e tramite **JAVASCRIPT** si aggiorna il frontend sostituendo il pulsante di aggiunta con la scritta 'ADDED'.

10 COMPONENTI PRINCIPALI

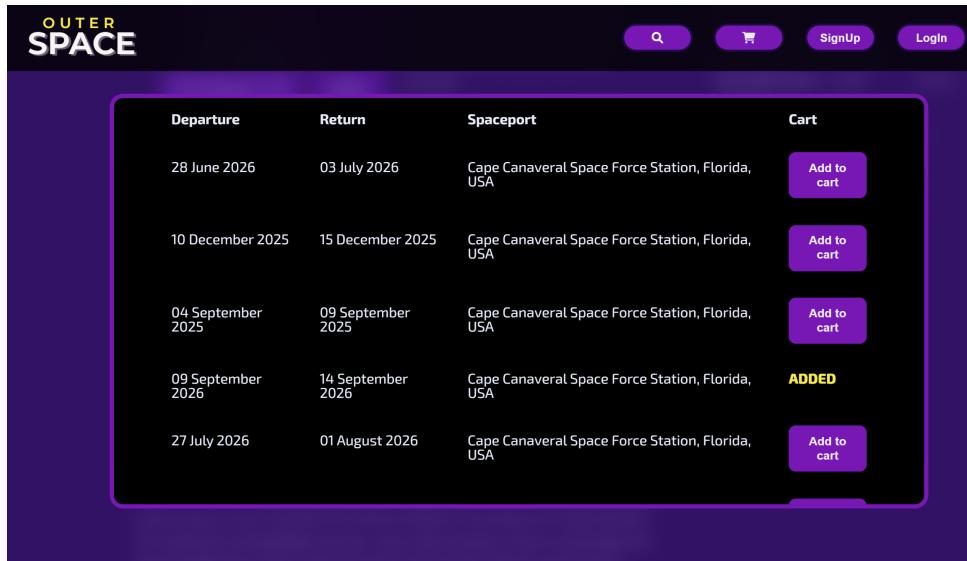


Figura 10: Pop-up Trip Dates

10.4 Header

L'header è presente in tutte le pagine principali e include:

- **Logo:** Il logo del sito, che funge anche da link alla homepage.
- **Slider-menu:** Un menu di navigazione con collegamenti alle sezioni principali del sito.
- **Icône di accesso:** Icône per l'accesso al login, al carrello, signup, logout.

L'intestazione mostra icône di accesso diverse a seconda del tipo di utente:

- *Utente non loggato:* Vengono mostrati i button per la registrazione e il login.
- *Utente loggato:* Vengono mostrati button per il logout, lo username del utente e in caso ci siano anche il nome e cognome. Nello slider menu viene aggiunto il collegamento per la pagina History Orders.

Il pulsante del carrello e della ricerca vengono mostrati in entrambi i casi.

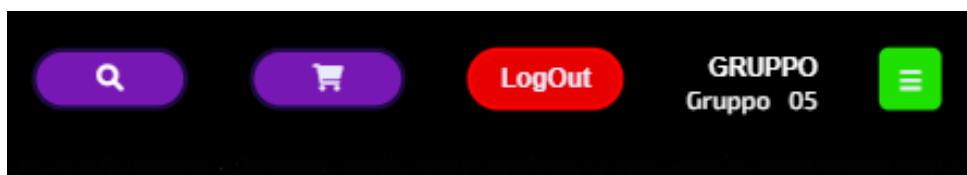


Figura 11: header di un utente loggato

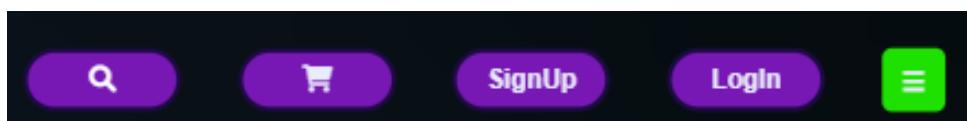


Figura 12: header di un utente non loggato

10.5 Footer

Il footer è presente in tutte le pagine e include:

- **Logo:** Il logo del sito, che funge anche da link alla pagina corrente.
- **Link utili:** Collegamenti alle sezioni principali del nostro sito.
- **Social media:** Icône con link ai profili social media dell'azienda.
- **Autori:** Informazioni sugli autori del progetto.

Parte IV

Partecipazione al progetto

Partecipanti	Contributo alle pagine
Alessio Leo	<ul style="list-style-type: none">• Order History e ticket (Backend + frontend)• Sign Up (backend registrazione)• Header• Signup (frontend+backend)• Logout (backend)• Login (frontend+backend)• Carrello (frontend e backend)• Support (frontend+backend)• Pagina error (404) (frontend)
Claudia Montefusco	<ul style="list-style-type: none">• About Us (css, html, php, js)• Signup (html, css)• Support (html)• Footer• catalogue-items (html,js,php): Jupiter, Mercury, Pluto, Neptune, Uranus, Andromeda, Pinwheel, Triangulum, Whirlpool, Carina, Eagle, Helix, Horsehead, Tarantula

Emanuele Tocci	<ul style="list-style-type: none"> • Homepage (frontend + backend) • Catalogue (frontend + backend) • Catalogue-items (frontend + backend): scrittura di un template generico (venus.php) da poter riutilizzare (e realizzazione di altri item), implementazione API sistema solare, popUp tripDates.php, effetto parallasse • Redirect 404 (.htaccess) ed homepage. (index.php) • headMetadata.html • Ricerca per nome (searchName.php) e tipologia (searchType.php) • Backend: getTripInfo.php, getAllUniqueTrips.php • Global.css, CssReset.css, slideshow.js
Rossella Pale	<ul style="list-style-type: none"> • NewSpace (frontend+backend) • Gestione delle newsletter (frontend+backend) • Gestione del carrello (backend + frontend) • Aggiunta e rimozione degli elementi dal carrello con aggiornamento di cart e TripDates • Footer • Backend: addToCart.php, gestioneAcquisti, gestioneCoupon, RemoveItemFromCart • ConfermaDinamica, errorpagamento • Aggiornamento vista catalogo dopo ricerca • Ricerca per budget • Gestione pagamento con Stripe • Login: solo ajax • <u>Gestione del database</u>

Parte V

Istruzioni generali

Il progetto fa uso di alcune librerie PHP facilmente installabili tramite Composer. È pertanto necessario avere il composer installato. In particolar modo sono state usate le seguenti librerie:

- Stripe: gestione dei pagamenti;
- PHPdotenv: salvataggio sicuro di informazioni sensibili

Dal momento che è stato usato **Github** come piattaforma di collaborazione, abbiamo preferito utilizzare **PHPdotenv** per salvare in modo sicuro la chiave segreta fornita da Stripe. In seguito si mostra un'alternativa qualora non si volesse installare PHPdotenv.

11 Installazione delle dipendenze

L'installazione delle **dipendenze** deve avvenire all'interno della cartella `dependencies`. Dal momento che sono stati forniti i file `composer.json` e `composer.lock`, l'installazione di tutte le dipendenze necessarie si limita all'esecuzione del seguente comando da terminale (nella cartella `dependencies`):

```
composer install
```

Questo generico comando si occupa di installare tutte le dipendenze specificate nei file `composer.*`, ossia **Stripe** e **PHPdotenv**. Nella cartella dovrebbe comparire una sottocartella chiamata `vendor`:

```
dependencies
  +-- vendor/
  |   +-- composer.json
  |   +-- composer.lock
```

Volendo è comunque possibile installare singolarmente le dipendenze:

11.1 Installare Stripe

```
composer require stripe/stripe-php
```

11.2 Installare PHPdotenv

```
composer require vlucas/phpdotenv
```

12 Configurazione della Chiave Segreta di Stripe

L'utilizzo di **PHPdotenv** implica la creazione di un file chiamato `.env` (file nascosto) all'interno del quale salvare la chiave privata di Stripe.

All'interno del file `.zip` fornito è già presente il file in questione per cui non dovrebbe essere necessario crearlo da capo.

12.1 Soluzione alternativa senza usare PHPdotenv

Se non si vuole usare il pacchetto PHPdotenv, é necessario modificare il file `src/components/Stripe/Checkout.php` ed inserire esplicitamente la chiave. Trovare il seguente blocco di codice:

```
/**  
 * Impostare la chiave segreta di Stripe.  
 */  
  
// Commentare se non si usa PHPdotenv  
$dotenv = Dotenv\Dotenv::createImmutable('.../.../...');  
$dotenv->load();  
$stripe_secret_key = $_ENV['STRIPE_SECRET_KEY'];  
  
// Decommentare ed inserire key se non si usa PHPdotenv  
// $stripe_secret_key = '';//
```

La chiave da inserire si trova nel file `.env` ed inizia con `sk_test`.

1. Commentare tutto il blocco relativo a `phpdotenv`
2. Decommentare l'ultima riga ed inserire la chiave fornita tra gli apici

13 Simulare il pagamento

Per la simulazione del pagamento Stripe é possibile utilizzare una qualunque carta di credito specificata dalla wiki ufficiale. Per esempio:

- NUMBER: 5555555555554444
- CVV: 123
- DATE: 01/28

I dati restanti possono anche essere inventati.