



# Pronti, Partenza, Lancio: Creare un Sito Statico con GitHub Pages

# Ship and manage your web projects faster

Deploy your projects on Google Cloud Platform's top tier infrastructure. You'll get 25+ data centers to choose from, 24/7/365 expert support, and advanced security with DDoS protection.

Try for free

■ Data centers (25) ■ CDN locations (200)

Con tutti gli strumenti e i framework di sviluppo web che esistono, creare un sito web sta diventando sempre più complicato. A volte, però, non c'è bisogno di molta interattività sul sito. Se l'obiettivo è far arrivare le informazioni a chi le cerca e non c'è bisogno di avere funzionalità complesse, un sito statico potrebbe essere la soluzione migliore.

In questo tutorial scoprirete cos'è un sito statico, quali vantaggi offre, quali sono i limiti e come creare e distribuire gratuitamente un semplice sito personale creato con HTML e Bootstrap utilizzando GitHub Pages.

## Cosa Sono le GitHub Pages?

[GitHub](#) è una piattaforma web che ospita i repository Git e permette di collaborare a progetti software. Offre strumenti per la condivisione e il tracciamento delle modifiche al codice, la gestione e la revisione del codice e la possibilità di aprire e rivedere le richieste di pull.

Non confondete [Git e GitHub](#)! GitHub è un servizio di hosting che permette la collaborazione tra developer, mentre Git è il software di controllo di versione locale che usate per salvare le istantanee dello stato del vostro progetto software.

[GitHub Pages](#) è una delle migliori funzioni di GitHub. È un servizio che vi permette di ospitare un [sito web statico](#) direttamente da un repository GitHub. Ciò significa che potete usare il vostro repository per archiviare il codice e i file del vostro sito web e GitHub li pubblicherà automaticamente come sito web accessibile online.

In sintesi, GitHub Pages è un modo semplice e veloce per rendere operativo il vostro sito ed è particolarmente utile per mostrare il vostro [portfolio](#), i vostri progetti open-source o altri contenuti statici.

## Siti Web Statici e Dinamici

Come abbiamo visto, GitHub Pages offre un modo per distribuire [siti web statici](#). Ma qual è la differenza tra un sito web statico e un sito web dinamico?

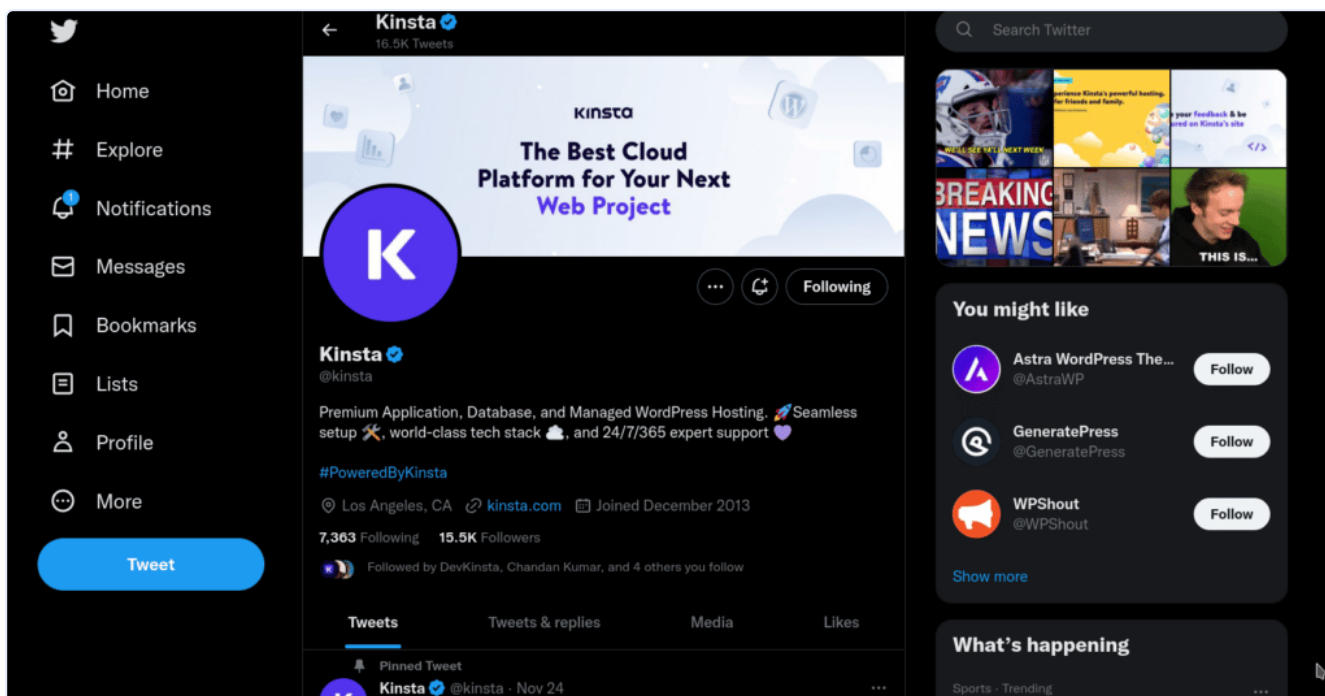
Iniziamo a parlare dei contenuti di questi siti.

Per contenuti statici si intendono elementi del sito web che rimangono invariati per tutti gli utenti, indipendentemente da chi siano o da quali azioni compiano sul sito. Si tratta di elementi come il testo, le immagini e il layout del sito, oltre al codice e ai file sottostanti che lo compongono. Un sito statico viene consegnato all'utente esattamente come è stato memorizzato.

Al contrario, i contenuti dinamici sono quelli che cambiano in base alle azioni dell'utente – come l'accesso, l'interazione con un paywall o i commenti a un articolo – o ad altri fattori, come l'ora o la posizione attuale.

Per esempio, un sito web che mostra un'immagine fissa di un prodotto mostrerà sempre la stessa immagine a ogni utente (statico). D'altro canto, un sito web che mostra l'ora corrente mostrerà a ogni utente un'ora diversa in base alla sua posizione (dinamica).

In generale, possiamo dire che un sito web è statico se contiene solo HTML, CSS e [JavaScript](#) sul [frontend](#), senza tecnologie server come [PHP o Python](#) che interagiscono con un database.



— Twitter è un sito dinamico.

Sebbene non sia possibile creare siti web dinamici utilizzando GitHub Pages, potete creare facilmente il vostro utilizzando un CMS come WordPress o generatori di siti statici come Gatsby o Hugo.

## Caratteristiche Principali di GitHub Pages

Vediamo i punti di forza di GitHub Pages come servizio di hosting:

1. **Facilità di configurazione e pubblicazione:** GitHub Pages vi permette di iniziare facilmente con pochi semplici passi. Potete attivare GitHub Pages per il vostro repository e specificare la fonte dei file del vostro sito web; GitHub pubblicherà automaticamente il vostro sito web e lo renderà disponibile a un URL basato sul vostro nome utente e sul nome del repository.
2. **Domini personalizzati:** Con GitHub Pages, potete usare un nome di dominio personalizzato per il vostro sito web invece dell'URL predefinito fornito da GitHub. Questo vi permette di utilizzare il vostro marchio e di rendere più facile per gli utenti

trovare e accedere al vostro sito web.

3. **Supporto HTTPS:** GitHub Pages supporta l'HTTPS, che consente di effettuare connessioni sicure al vostro sito web. Questo è fondamentale per aumentare la fiducia nei confronti del vostro sito.
4. **Supporto per Jekyll:** GitHub Pages supporta Jekyll, un generatore di siti statici che vi permette di creare siti web sofisticati utilizzando template e altre funzioni. In questo modo è facile creare siti web dall'aspetto professionale senza dover scrivere tutto il codice da zero.

## Limitazioni

Come già detto, potete creare siti statici usando solo GitHub Pages. Se volete creare un progetto complesso con molte funzionalità, dovete ricorrere ad altri [servizi di hosting](#). Dovete anche tenere presente che non potete usare GitHub Pages per scopi commerciali, come la gestione di un'attività online o di un [ecommerce](#).

GitHub Pages non permette al vostro sito di essere più grande di 1 GB, il che significa che i file del vostro repository non possono superare quella quantità di memoria. Nella maggior parte dei casi, 1 GB è più che sufficiente per un sito statico; assicuratevi sempre di [comprimere le immagini](#).

Inoltre ha un limite di larghezza di banda di 100 GB al mese. Questa quantità di larghezza di banda è sufficiente per distribuire il vostro sito web a qualche migliaio di persone al mese, quindi, a meno che non abbiate un pubblico enorme, sarete a posto.

## Creare e Distribuire con GitHub Pages: Guida Passo Passo

La creazione di una pagina GitHub è un processo semplice e diretto per ospitare un sito statico. Tenete presente che se avete bisogno di un qualche tipo di connessione al database, dovete avere un [provider di database](#) esterno.

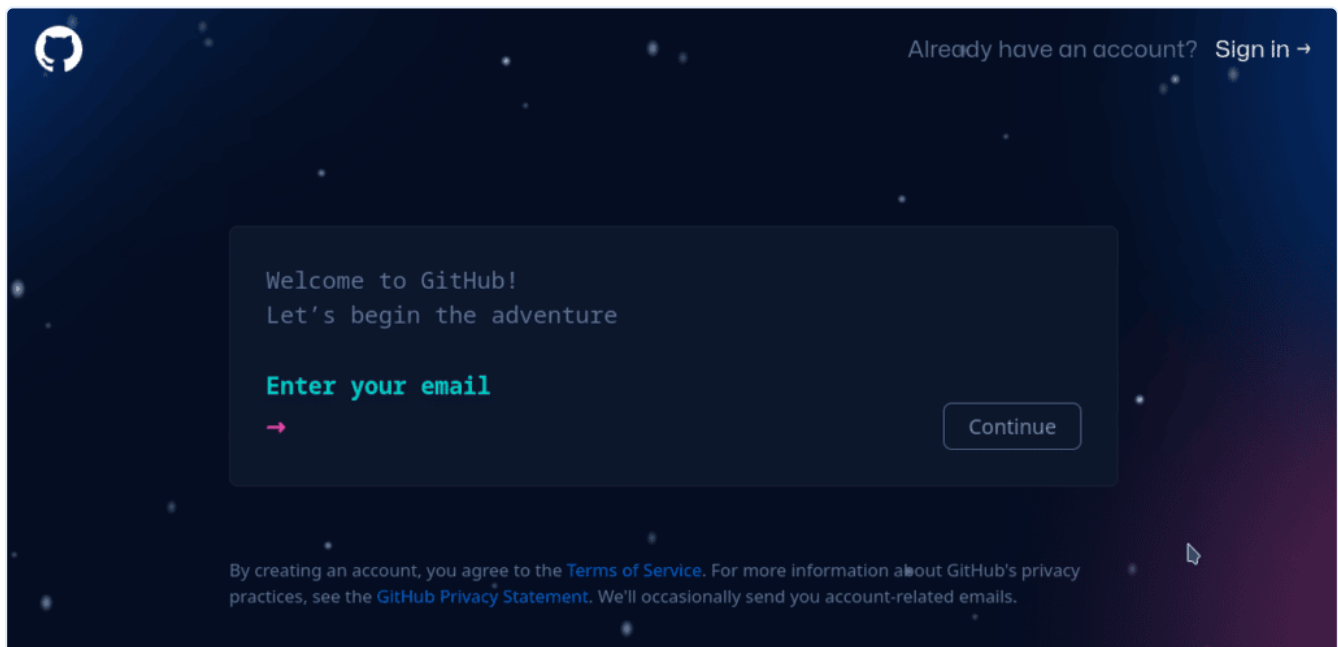
Nella seguente guida scoprirete come creare una pagina GitHub da zero. Questo include la creazione di un repository remoto, la creazione di un [sito web personale responsive con](#)

[l'HTML](#) e la distribuzione sul web con GitHub.

Entriamo nel vivo dell'argomento!

## 1. Iscriverti a GitHub

Per iniziare, dovete avere un account GitHub attivo. Se non ce l'avete, andate alla [pagina di registrazione](#). La compilazione del modulo non dovrebbe richiedere più di un paio di minuti.



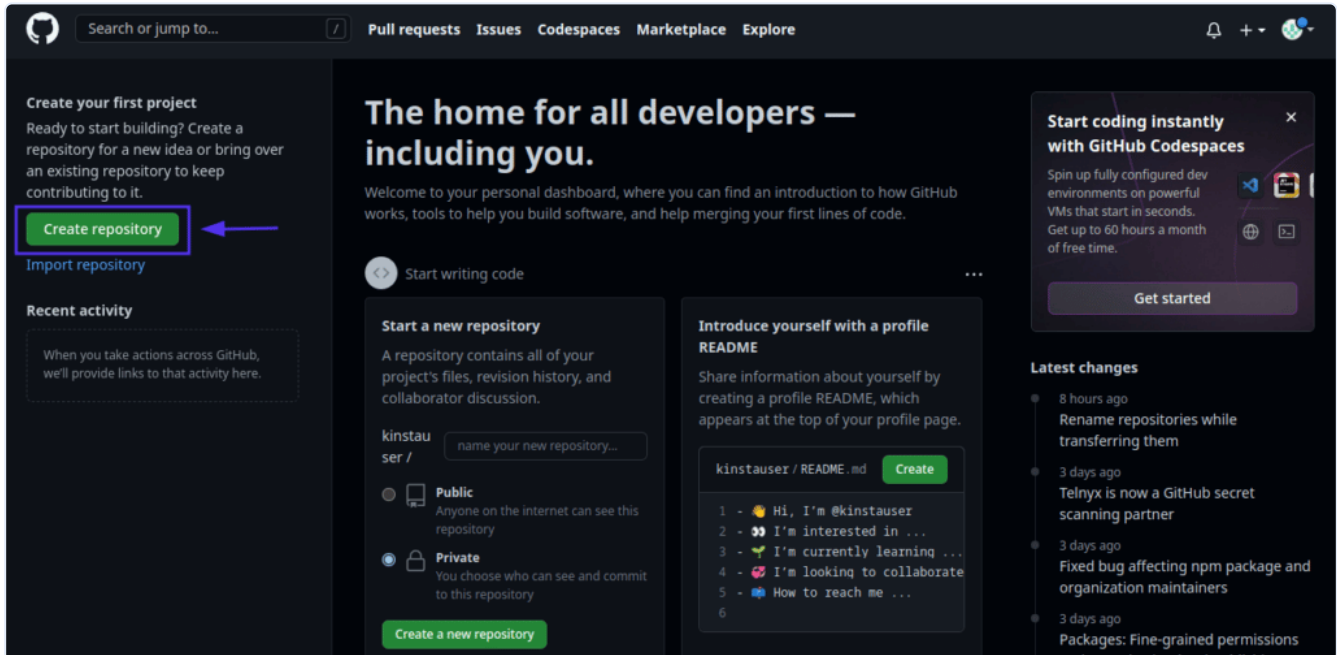
— Pagina di registrazione di GitHub.

Dopo aver effettuato l'accesso, dovrete essere in grado di creare un repository remoto.

## 2. Creare un Repository Remoto

Un repository è una directory in cui viene salvato tutto il codice relativo a un determinato progetto.

Dalla home page di GitHub, fate clic sul pulsante “New” o “Create repository” situato nel pannello sinistro del sito. In questo modo verrete reindirizzati a un modulo in cui dovrete inserire le informazioni relative al vostro repository.



— Creare un repository GitHub.

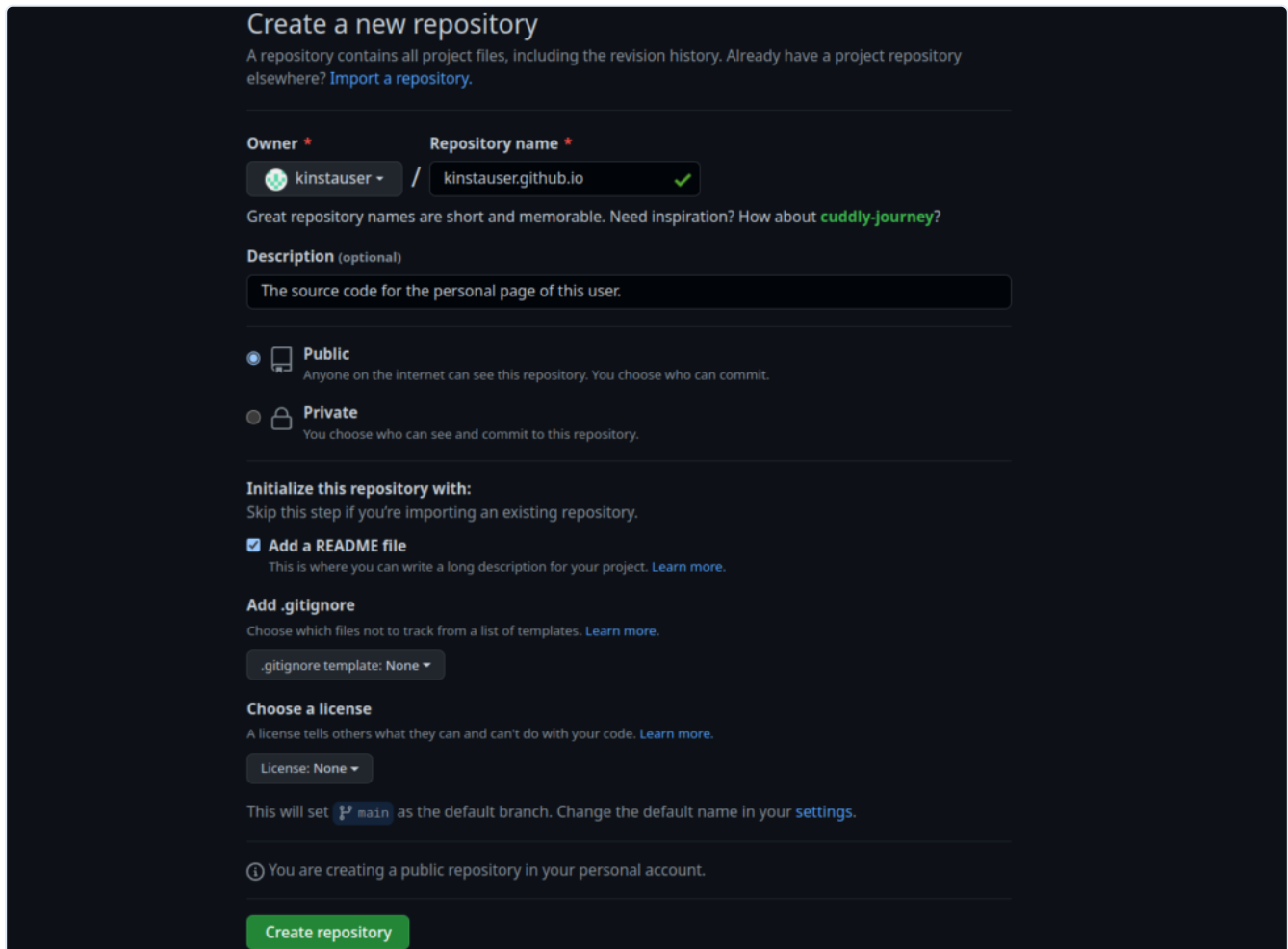
I prossimi passi sono fondamentali:

1. Impostate il nome del vostro repository su "yourusername".github.io.
2. Selezionate il pulsante “Public”. Dovete impostare il repository su **Public** per pubblicare il vostro sito.
3. Aggiungete un README.

Potete avere un repository solo per un determinato account personale o organizzazione. Ecco perché il nome del repository è il vostro nome utente e il dominio `github.io`. Più avanti vedremo come creare un sito da un repository.

A meno che non abbiate GitHub Pro, potete pubblicare una pagina GitHub solo se il repository è pubblico. Tenetelo presente se non volete condividere pubblicamente il codice sorgente del vostro sito.

A questo punto, dovrete ottenere qualcosa di simile a quanto segue:



The screenshot shows the GitHub 'Create a new repository' page. At the top, it says 'Create a new repository' and provides a brief explanation of what a repository is. Below this, there are two main sections: 'Owner' and 'Repository name'. The 'Owner' is set to 'kinstauser' and the 'Repository name' is 'kinstauser.github.io', which is marked as valid with a green checkmark. A suggestion for a repository name, 'cuddly-journey', is shown below. The 'Description' field is optional and contains the text 'The source code for the personal page of this user.' The 'Visibility' section has two options: 'Public' (selected) and 'Private'. The 'Initialize this repository with' section has three checkboxes: 'Add a README file' (checked), 'Add .gitignore' (unchecked), and 'Choose a license' (unchecked). The 'Add a README file' checkbox is checked, and the 'Add .gitignore' checkbox is unchecked. The 'Choose a license' checkbox is unchecked. At the bottom, there is a green 'Create repository' button. A note at the bottom states: 'You are creating a public repository in your personal account.'

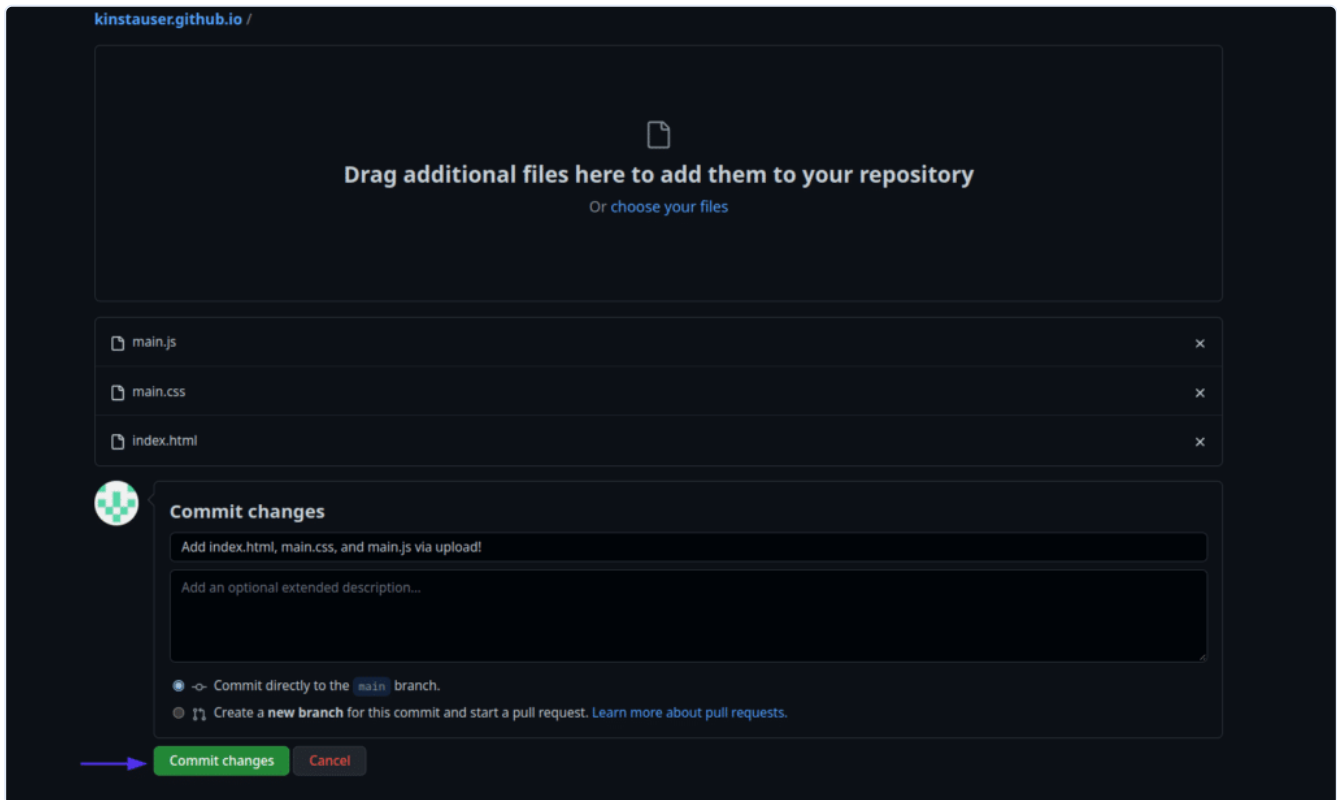
— Forma del repository GitHub.

Se il codice sorgente del vostro sito è già funzionante, potete saltare il comune [workflow di Git](#) e inserire i file direttamente nella repo.

Per farlo, fate clic sul menu `Add file` nel vostro repository e selezionate l'opzione **Upload files**. Quindi selezionate i file del vostro sito web, con il file HTML principale chiamato `index.html`. Ricordatevi di inserire nel repository anche tutti i file CSS e JavaScript.



Infine, premete il pulsante **Commit changes**.



— Caricare i file su GitHub.

Nella sezione seguente realizzeremo un semplice sito web personale con HTML e Bootstrap. Poi lo caricheremo su GitHub e lo imposteremo come pagina pubblica di GitHub con un dominio personalizzato.

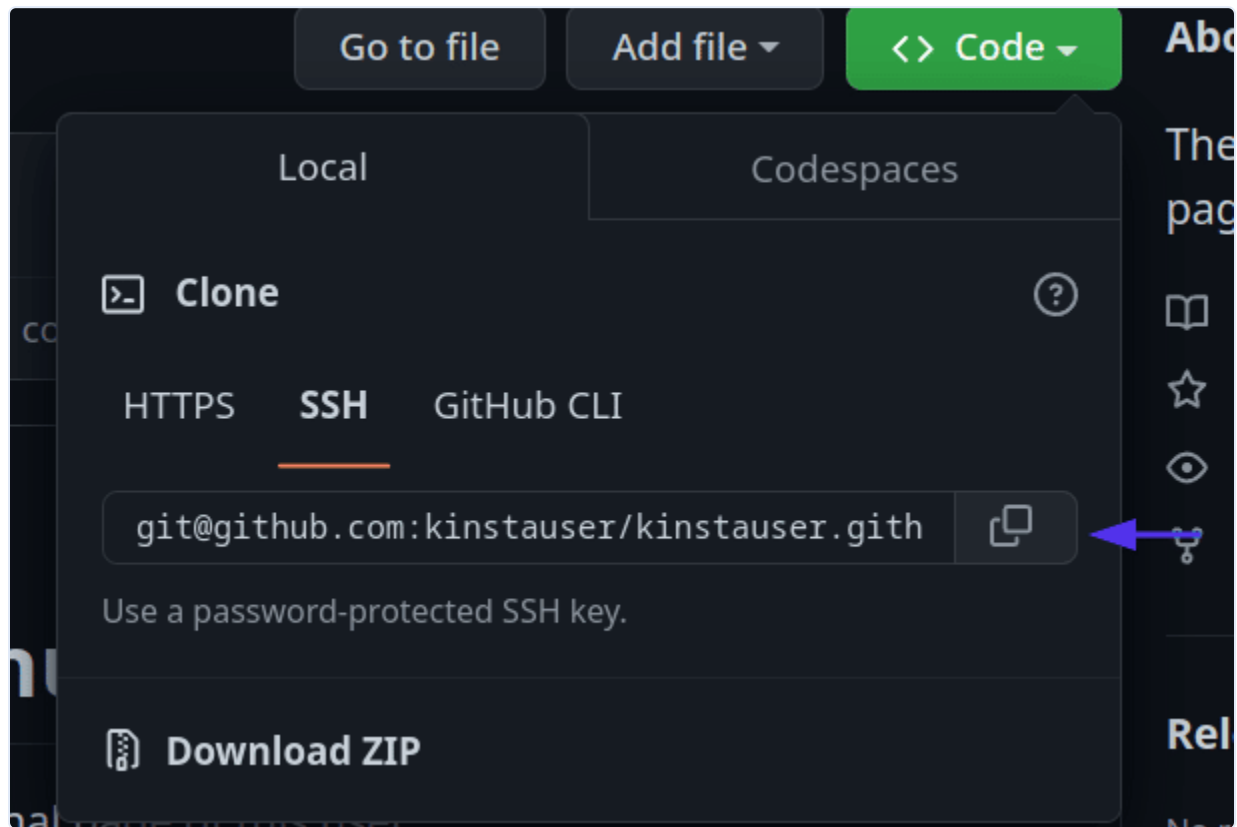
### 3. Creare un Sito Personale

Inizieremo clonando il repo GitHub appena creato. Per farlo, assicuratevi di avere il [client Git](#) già installato sul computer. (Se non ce l'avete, date un'occhiata al nostro tutorial su [Git e GitHub](#))

## Info

Dovrete impostare l'autenticazione SSH per il vostro account GitHub. Questa operazione può sembrare un po' complessa, ma abbiamo un articolo completo sulle [chiavi SSH per GitHub](#) che vi spiega tutto.

Andate alla scheda `code` del vostro repository e copiate l'URL SSH nell'opzione **SSH**.



— URL SSH del repository.

Quindi, avviate il terminale o una riga di comando. Sulla maggior parte delle distribuzioni Linux e macOS, potete usare la scorciatoia `Ctrl + Alt + T` o cercare **Terminale** nel menu delle applicazioni del vostro sistema. Su Windows, potete usare il [BASH Git](#) installato di default con Git, il CMD, PowerShell o un client GUI.

Nel vostro terminale, digitate `git clone` e l'URL che avete copiato. Questo scaricherà e creerà una copia del repository remoto sul vostro computer locale, così potrete costruire il vostro sito web.

```
MEGA/my-git/github
> git clone git@github.com:kinstauser/kinstauser.github.io.git
Cloning into 'kinstauser.github.io'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
Receiving objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
```

— Comando Git clone.

Poi entrate nella nuova cartella denominata **yourusername.github.io** con `cd` e `ls`. Dovreste vedere la cartella **.git**, che contiene la configurazione e i metadati del vostro progetto, oltre al file **README.md** se ne avete creato uno.

Aprire il vostro [editor di testo](#) preferito (per esempio [Sublime Text](#)) e iniziate a creare il vostro sito web.

Nella root del repository, create un file chiamato **index.html** (questo nome è richiesto da GitHub Pages) e inserite la tipica struttura del codice HTML:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Kinsta User</title>
</head>
<body>

</body>
</html>
```

Come abbiamo detto in precedenza, useremo [Bootstrap 5.0](#), un framework CSS open-source che ci aiuta a costruire siti web responsive più facilmente. Come vedrete, non dovremo usare CSS personalizzati per questo sito in particolare.

Per inserire Bootstrap nella nostra pagina, dovremo includere il CSS e il JavaScript compilati tramite un [CDN](#). Incollate il seguente codice all'interno dell'HTML `<head>` per poter usare il CSS di Bootstrap:

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet">
```

Allo stesso modo, includeremo anche il CDN [Devicon](#) per poter utilizzare senza problemi le icone SVG dei linguaggi di programmazione e delle tecnologie più diffuse:

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/gh/devicons/devicon"
```

Ora, per includere il codice JavaScript, inserite il seguente codice proprio sopra la fine del tag `</body>`:

```
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap"
```

Inizieremo creando un header per il nostro sito web. Si tratterà di un header scuro, con link alla nostra pagina indice e ad altre due pagine – “Projects” e “Reading log” – che potrete creare in seguito:

```
<nav class="navbar navbar-dark navbar-expand-lg bg-dark ">
  <div class="container-fluid">
    <div class="mx-4">
      <a class="navbar-brand" href="#">Kinsta User</a>
    </div>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
      data-bs-target="#navbarTogglerDemo02" aria-controls="navbarTogglerDemo02"
      aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarTogglerDemo02">
      <ul class="navbar-nav me-auto mb-2 mb-lg-0">
```

```

        <li class="nav-item">
            <a class="nav-link" href="#">Projects</a>
        </li>
        <li class="nav-item">
            <a class="nav-link" href="#">Reading Log</a>
        </li>
    </ul>
</div>
</div>
</nav>

```

Usiamo i wrapper di Bootstrap `navbar` e `navbar-expand-lg` per creare un contenitore responsive che collassa quando la larghezza del display è inferiore a 992px. Questo avviene grazie all'opzione `griglia lg`. Se volete saperne di più sulle opzioni della griglia, date un'occhiata alla [pagina di Bootstrap dedicata al layout](#).

Ora creiamo due colonne responsive all'interno di un contenitore: una per un'[immagine gratuita](#) di [Unsplash](#) e l'altra per una nostra descrizione:

```

<div class="container my-4 ">
    <div class="row justify-content-center">
        <div class="col-lg mb-lg-4">
            
        </div>
        <div class="col-lg mx-2 align-self-center">
            <div class="my-3">
                <h1 class="text-center">I'm a Kinsta User</h1>
                <p>As a passionate software developer, I am deeply enthusias
                    developing software applications. I am constantly learni
                    technologies and approaches, and I have a strong desire
                    solutions to complex problems. I am driven by my curiosi
                    I
                    am committed to producing high-quality, well-designed so

```

```
        users.  
    </p>  
  </div>  
</div>  
</div>  
</div>
```

Come potete vedere, l'immagine proviene da un file locale, quindi deve essere presente nel repository quando applichiamo le modifiche al repo di GitHub.

Infine, all'interno del nostro contenitore Bootstrap, useremo le icone SVG di Devicon e un po' di CSS interno per far risaltare le nostre competenze:

```
<div class="my-4">  
  <div class="text-center mb-4">  
    <h1>My Skills</h1>  
  </div>  
  <div class="row ">  
    <style>  
      I {  
        font-size: 4em;  
      }  
    </style>  
  
    <div class="col">  
      <div class="text-center">  
        <h4>WordPress</h4>  
        <i class="devicon-wordpress-plain"></i>  
      </div>  
    </div>  
    <div class="col">  
      <div class="text-center">
```

```

        <h4>Django</h4>
        <i class="devicon-django-plain"></i>
    </div>
</div>
<div class="col">
    <div class="text-center">
        <h4>Python</h4>
        <i class="devicon-python-plain"></i>
    </div>
</div>
<div class="col">
    <div class="text-center">
        <h4>GitHub</h4>
        <i class="devicon-github-original" ></i>
    </div>
</div>
</div>
</div>

```

Visto che usiamo il tag HTML `<i>`, possiamo trattarlo come un font. Quindi impostiamo la dimensione dei nostri loghi su 4 em dichiarandola nel tag `style`.

Ecco il risultato finale di questo semplice sito web personale:





## I'm a Kinsta User

As a passionate software developer, I am deeply enthusiastic about creating and developing software applications. I am constantly learning and experimenting with new technologies and approaches, and I have a strong desire to create innovative and effective solutions to complex problems. I am driven by curiosity and love for problem-solving, and I am committed to producing high-quality, well-designed software that meets the needs of users.

## My Skills

WordPress



Django



Python

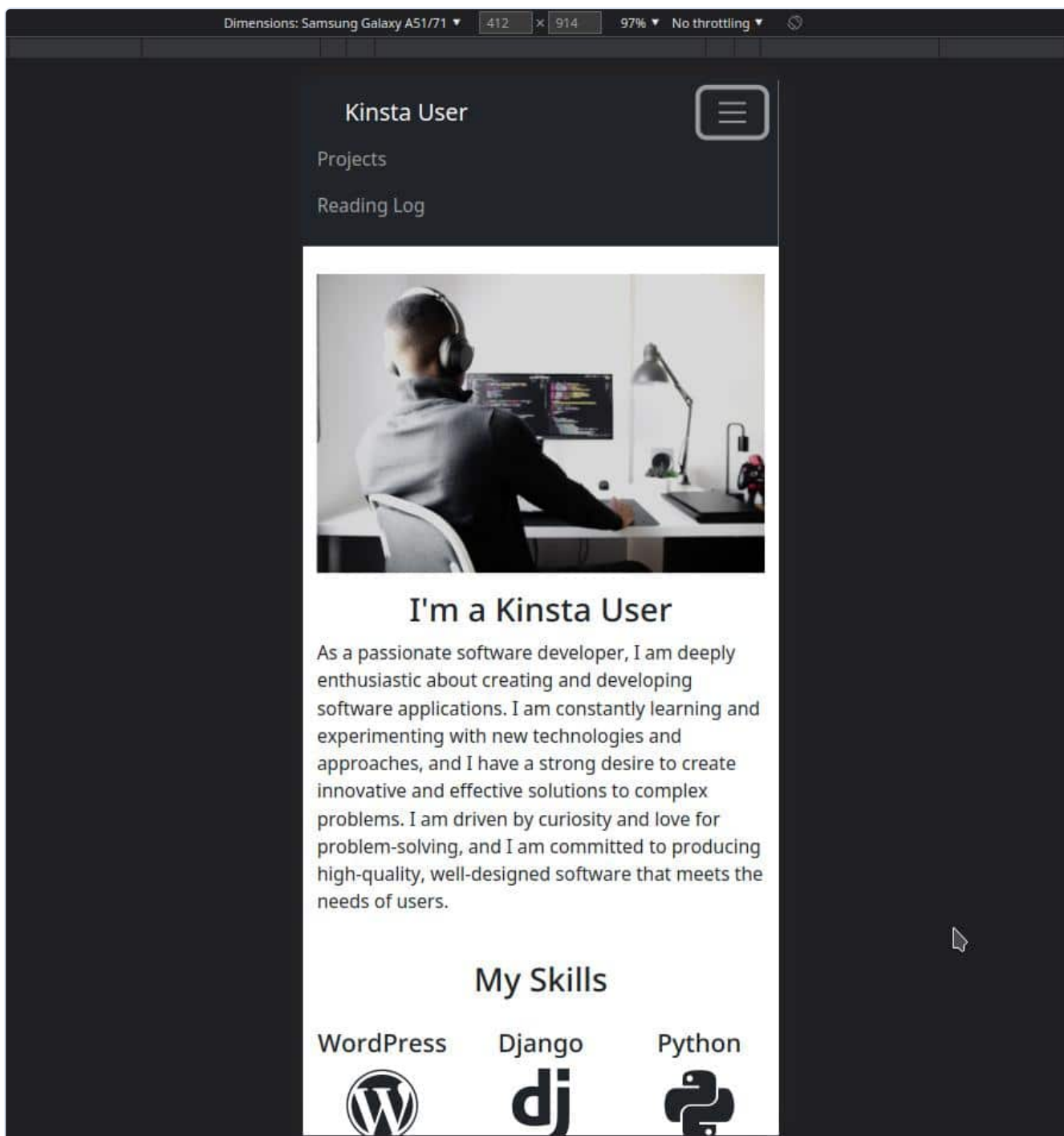


GitHub



— Pagina personale.

Sapevate che oltre il [50% del traffico di un sito web](#) proviene da dispositivi mobili? Grazie all'utilizzo di Bootstrap, abbiamo risparmiato un sacco di codice CSS e abbiamo ottenuto un sito web responsive, come potete vedere qui sotto.



— Pagina reattiva.

Potete personalizzare questo sito quanto volete. Ecco il [codice sorgente completo su GitHub](#), a vostra disposizione.

Potete anche collegare un CMS headless come Ghost utilizzando una delle nostre [soluzioni di Hosting di Applicazioni](#) complete. Potete collegarvi direttamente al vostro repository GitHub attraverso il vostro [cruscotto MyKinsta](#) e avere un nuovo sito funzionante in pochi minuti.

È il momento di fare il push dei vostri file. Per farlo, eseguite i seguenti comandi sul vostro terminale, al livello superiore del vostro progetto.

```
git add .  
git commit -m "Added website source code and image"  
git push
```

Ora possiamo usare questo sito web per configurare la nostra pagina GitHub.

## 4. Pubblicare una Pagina GitHub Utente

Non appena invierete l'**index.html** al repository remoto con il vostro nome utente, GitHub avvierà automaticamente un processo di workflow per configurare il vostro sito online. Potrebbero volerci un paio di minuti, ma il vostro sito statico sarà attivo e funzionante automaticamente.

L'URL del vostro sito sarà simile al seguente:

<https://kinstauser.github.io/>

Se dopo 10 minuti il vostro sito non è ancora online, potete provare ad apportare una modifica fittizia al vostro codice (per esempio, aggiungendo uno spazio) e fare nuovamente il push per riattivare il processo di creazione di GitHub Pages.

### Important

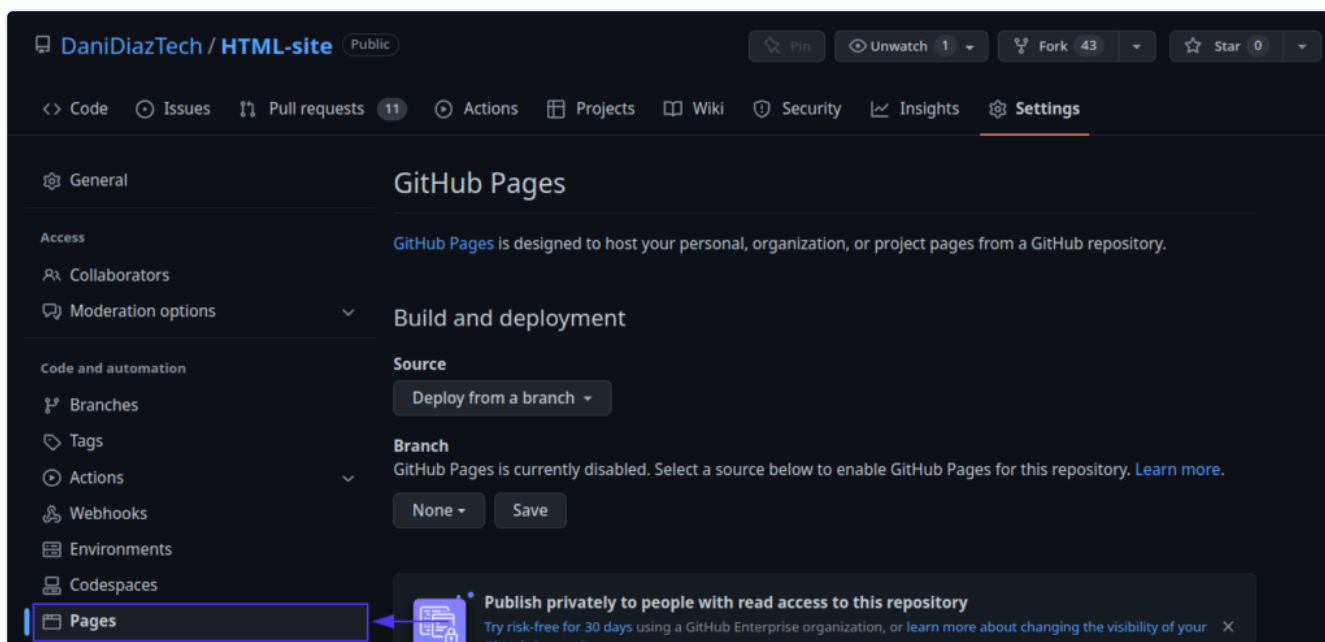
Non è possibile annullare la pubblicazione di una pagina GitHub a livello utente. Se volete farlo, dovrete cambiare il nome del repo.

## 5. Pubblicare una Pagina GitHub di un Repository

Finora abbiamo creato un sito utente, ma se volessimo avere più siti GitHub pubblicati? Allora dobbiamo scegliere un sito di progetto.

Come esempio, useremo il sito HTML tech che abbiamo creato nel tutorial [Git per lo sviluppo web](#).

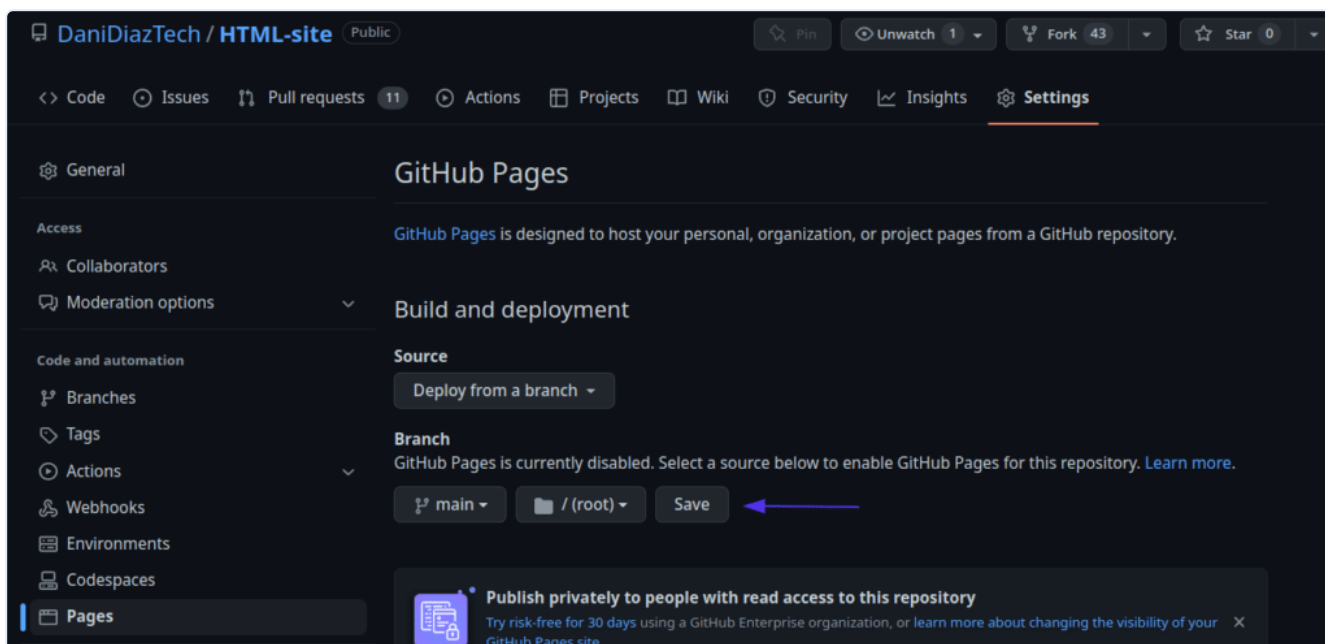
Il modo più semplice è quello di andare nella scheda **Settings** del nostro repository, quindi nell'opzione **Pages** all'interno della sezione "Code and automation".



— Impostazioni del repository.

Selezionate la fonte **Deploy from a branch** e fate clic sulla branch da cui volete distribuire i file. È molto consigliato pubblicare dalla **branch principale**, ma creare funzionalità e correggere bug usando branch ausiliari e poi unirli. In questo modo sarà più difficile introdurre errori nel sito pubblicato.

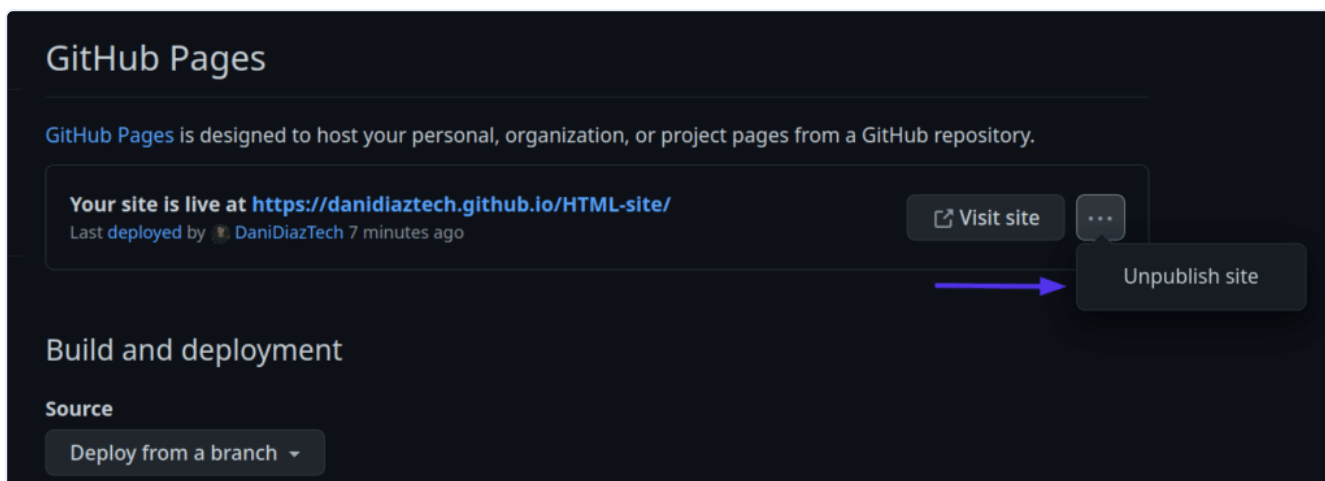
Una volta selezionata la branch, selezionate la cartella da cui volete servire i file – di solito la root (/) – e fate clic su salva.



— Pubblicazione dalla principale.

Anche in questo caso, attendete qualche minuto. Il vostro sito dovrebbe essere disponibile all'indirizzo `"yourusername".github.io/`.

Per disattivare la pubblicazione di un sito di un repository, potete andare su **Settings**, poi su **Pages** e fare clic sull'opzione dei tre puntini. Verrà visualizzato un riquadro con il messaggio **Unpublish site**.



— Rimuovere un sito dalla pubblicazione.

Fantastico! Il sito del vostro progetto open-source è attivo e funzionante, ma il nome del dominio è lungo e non certo facile da ricordare. Vediamo come migliorare questo aspetto.

## 6. Impostare un Dominio Personalizzato

Il modo più semplice per impostare un dominio personalizzato per una pagina GitHub e assicurarsi che funzioni è quello di andare dal vostro provider DNS e creare quattro record A per gli indirizzi IP delle pagine GitHub:

```
185.199.108.153
185.199.109.153
185.199.110.153
185.199.111.153
```

Dovete anche impostare un record CNAME con `yourusername.github.io` come destinazione. Di solito le modifiche ai DNS sono lente, quindi abbiate pazienza, potrebbero

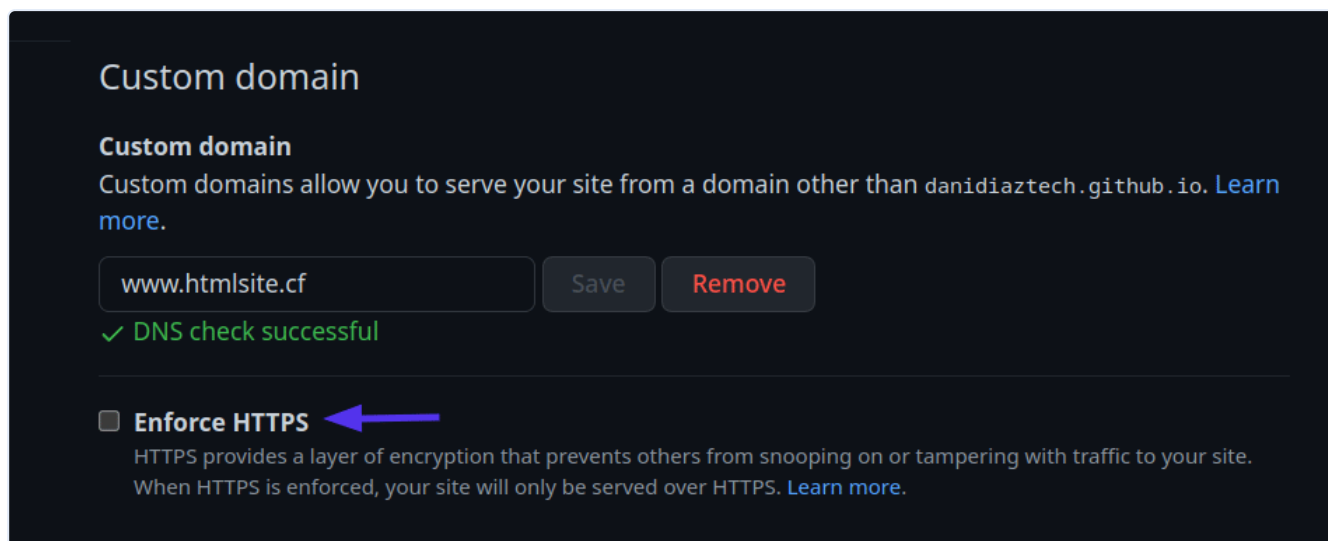
richiedere anche un giorno intero.

### Info

Se avete bisogno di maggiori informazioni su come creare questi record, consultate le guide del vostro provider DNS.

Dopo aver fatto questo, andate alla sezione **Custom domain** nelle impostazioni del vostro repo, inserite il vostro dominio, fate clic sul pulsante **Save** e attendete che GitHub controlli il vostro dominio personalizzato.

Inoltre, se il vostro DNS lo supporta, selezionate la casella **Enforce HTTPS** per servire il vostro sito solo su [HTTPS](#).




Custom domain

**Custom domain**  
Custom domains allow you to serve your site from a domain other than danidiaztech.github.io. [Learn more.](#)

✓ DNS check successful

---

☐ **Enforce HTTPS** 

HTTPS provides a layer of encryption that prevents others from snooping on or tampering with traffic to your site. When HTTPS is enforced, your site will only be served over HTTPS. [Learn more.](#)

— Dominio personalizzato.



Congratulazioni! Se avete seguito le istruzioni fino a questo punto del tutorial, ora avete un sito web statico ospitato su GitHub Pages gratuitamente.

## Migliori Pratiche per GitHub Pages

Prima di lasciarci, ecco alcune buone pratiche che dovrete tenere in considerazione quando pubblicate un sito GitHub:

1. Non fate mai commit diretti alla branch da cui state facendo il deploy. Create modifiche in altri rami e poi create una richiesta di pull.
2. Scegliete un [buon dominio](#) per il vostro sito, se potete permettervelo. È una delle decisioni più importanti per la vostra marca personale o di progetto.
3. Non usate GitHub Pages per scopi commerciali, per esempio per creare un sito di ecommerce.
4. Valutate le vostre esigenze. È bello poter pubblicare un sito statico gratuitamente, ma se vi aspettate molto traffico o volete funzioni complesse, la soluzione migliore è pagare un [ottimo servizio di hosting](#).

## Riepilogo

Lo sviluppo del web diventa ogni giorno più complicato. I siti statici esistono fin dall'avvento di internet e sono un ottimo modo per iniziare a creare progetti.

In questo tutorial avete imparato cosa sono i siti statici e come configurarli online utilizzando GitHub Pages. Avete creato un semplice sito personale utilizzando Bootstrap e lo avete pubblicato come sito utente di GitHub. Avete anche imparato come rendere operativo il sito di un progetto e come annullare la pubblicazione se necessario.

Nel complesso, GitHub Pages è uno strumento comodo e potente per ospitare gratuitamente il vostro sito web statico. Se volete mostrare il vostro portfolio, condividere i vostri progetti open-source o iniziare a creare una presenza online, GitHub Pages è una scelta eccellente. E una volta che avrete ottenuto abbastanza traffico o sarete pronti a creare un [sito full-stack](#), potrete passare senza problemi ad altri [servizi di hosting di applicazioni](#), come quello di Kinsta.