Corso di Programmazione Web e Mobile
A.A. 2021-2022, Sito di meteo

« Gabriele, Bellini, 988546 »
gabriele.bellini@studenti.unimi.it

2022-07-20

1 Introduzione

L'applicazione web presentata si occupa di fornire informazioni meteo a livello mondiale, è reperibile nel repository github https://github.com/g7240/pwbProject.git e raggiungibile online nella sua forma funzionante attraverso i link:

- https://pwb-project.vercel.app
- https://pwb-project-git-main-g7240.vercel.app
- https://pwb-project-g7240.vercel.app

La pagina web, oltre a fornire dati, si colora con le immagini Google del luogo selezionato e dà la possibilità di usufruire di telecamere Windy disposte sul territorio in tutto il modo.

1.1 Destinatari

1.1.1 Capacità e possibilità tecniche.

Gli utenti che accedono alla pagina sono consapevoli di che cosa stanno cercando. Il sito di meteo raccoglie nel suo corpo principale un carosello con le foto del luogo, dei riferimenti con immagini ad altre località, e i dati della giornata corrente. Questi elementi hanno un layout ben distinto e semplice da identificare , non sono necessarie esperienze pregresse per l'utilizzo della pagina che, pertanto, è idonea a un vasto pubblico, data anche la progettazione responsiva che consente l'accesso da ogni tipo di dispositivo, fisso o mobile che sia.

La quantità di banda utilizzata per accedere al sito è in media quella necessaria per caricare dalle 6 alle 13 immagini più qualche Kb per i documenti css e js di bootstrap per una media misurata di 20Mb.

Per quanto concerne i dispositivi mobili, la velocità ottimale di connessione per accedere in modo naturale alla pagina è il 4G, sebbene, essendo la pagina costruita in maniera modulare e non bloccante, sia possibile accedere anche tramite 3G, infatti ogni servizio è caricato separatamente appena possibile, tutto ciò con priorità per i dati meteo.

Quindi un eventuale fruizione della pagina con connessioni più lente è possibile e permette di visualizzare agevolmente la tabella delle previsioni del tempo, che necessita solo di qualche Kb per essere caricata; penalizzato invece risulta l'aspetto grafico: eccetto per l'impaginazione, non si ha un accesso immediato ai contenuti multimediali grafici come immagini e video-meteo in diretta; in ogni caso la logica dell'applicazione rimane funzionante e si riesce a fruire della lista località cercate precedentemente e dei dati meteo di quella corrente.

1.1.2 Linguaggio.

L'applicazione, sebbene abbia potenzialità per un utilizzo internazionale, per il momento è sviluppata nella sola lingua italiana. Notiamo però che le molte

immagini e il layout semplice potrebbero permettere una fruizione del sito anche senza leggere.

Il linguaggio grafico è molto immediato, si avvale per lo più dell'uso di bottoni, toggle switch, search bar, menù a tendina, barra di navigazione e di un carosello: tutti elementi molto comuni nelle pagine web moderne. I bottoni responsivi e i loro cambi di stile, che però non stravolgono il layout della pagina, vogliono guidare l'utente nell'utilizzo delle funzionalità che il sito mette a disposizione.

1.1.3 Motivazione.

La ragione per cui un utente decide di usufruire del sito è la necessità di poter accedere ad informazioni aggiornate sul meteo, ciò in una web app che non solo fornisce dati ma anche immagini in diretta e che è perfetta per organizzare un viaggio quando si deve scegliere tra più mete visto che essa conserva una memoria delle ultime visite e può, pertanto, aiutarci a tenere sott'occhio le località che più abbiamo cercato.

Il sito cerca di soddisfare le esigenze di ogni tipologia di utente dai più consapevoli e motivati, per cui è presente la search bar delle località, ai meno, per i quali nel momento dell'accesso viene servita in automatico una risorsa il più adatta possibile che si basa sulla posizione tracciata per mezzo del loro ip; è poi il layout e la dinamicità della pagina che dovrebbe guidare e intrattenere l'utente che vi accede. Per gli utenti attivi nella ricerca ma inconsapevoli sono fornite delle località di esempio di cui cercare il meteo che possono essere consultate tramite il browsing nella pagina. Esse vengono anche usate come strumento di monitoring per rispondere alle esigenze di quegli utenti interessati al meteo ma passivi nella ricerca, a cui vengono fornite delle località cercate nei precedenti accessi e monitorate nel local storage del browser.

Tabella 1: Modello di Bates

	Active	Passive
Directed	Searching (ok)	Monitoring (ok)
Undirected	Browsing (ok)	Being Aware (ok)

1.2 Modello di valore

Il valore dell'applicazione web sta nella sua capacità di attrarre utenti, da cui si può trarre profitto tramite l'advertising. Il sito offre in maniera aggregata più servizi e questa completezza crea un plusvalore rispetto alle singole API interpellate per prendere i dati con cui è renderizzata la pagina.

Il carico di utenti presente, misurabile real-time, e futuro, stimabile basandosi su statistiche fatte sugli accessi passati, rappresenta una vero e proprio valore quantificabile e scambiabile in favore di inserzionisti.

1.3 Flusso dei dati

1.3.1 Ottenere i contenuti

I contenuti della pagina devono essere reperiti online, avvalendosi di altri servizi che li mettono a disposizione, in quanto produrli significherebbe dover avere un sistema di sensori e videocamere distribuite globalmente e gestite in maniera centralizzata, con elevati costi di gestione e manutenzione.

Per il momento non sono presenti alcun tipo di costo per l'uso dei servizi esterni data l'ottica didattica e il basso numero di accessi all'applicazione web. In caso di un utilizzo commerciale del sito dovrebbero essere acquistati i servizi:

- Windy (9990€ all'anno)
- Google Costume Search ($5 per mille richieste)
- OpenWeatherMap (0.0014€ per ogni API call)

1.3.2 Archiviare e organizzare i contenuti

I contenuti della pagina sono tutti serviti dinamicamente con una pagina che cambia da accesso ad accesso. I dati vengono caricati per mezzo di API a servizi web.

Lato server, almeno che la località di cui è richiesto il meteo non sia presente nella richiesta, viene effettuata una geo-localizzazione tramite IP, da cui vengono estratti paese e nome della città che vengono sia usati per renderizzare il documento html da servire e anche dal server per richiedere gli url delle immagini del luogo che a loro volta vengono renderizzate nel file inviato al client.

Lato client invece sono usati un web worker per caricare dal web le immagini i cui url sono stati passati dal server; uno script, che usa nome e paese, per caricare la tabella dei dati del meteo e lat-long ritornati dalle API meteo per caricare (con le API di windy) il video in diretta del luogo; in più dal local storage vengono recuperati i dati delle passate ricerche, le cui immagini vengono poi caricate dal web tramite il web worker.

1.3.3 Pubblicare i contenuti

I contenuti derivano tutti da servizi esterni che aggiornano i loro dati periodicamente nell'arco di decine di minuti o ore, per tanto non deve esserci alcun inserimento di nuovi dati da parte della web app stessa, la quale si occupa solo di gestire le richieste, talvolta usando alcune risposte come query per altre API, talvolta selezionando tra i dati ricevuti i contenuti da visualizzare davvero nella pagina. Un esempio tra tanti: di tutte le possibili informazioni che si possono richiedere con le api di google search si è deciso di richiedere solamente delle immagini, senza neanche la descrizione, le parole chiave o altro e tra esse selezionare solo quelle che rispettano un adeguato livello di qualità grafica.

I contenuti pubblicati, essendo tutti risultato di richieste ad altri servizi, non nascono per essere condivisi singolarmente o per fornire dati ad altri servizi

in quanto, oltre a provocare un aumento della latenza delle richieste, potrebbe provocare dei problemi di licenza con i fornitori delle API. La condivisione preferibile del sito è quella naturale in cui l'utente, tramite gli strumenti del browser, inoltra con email o social media la pagina web per intero copiando il suo url.

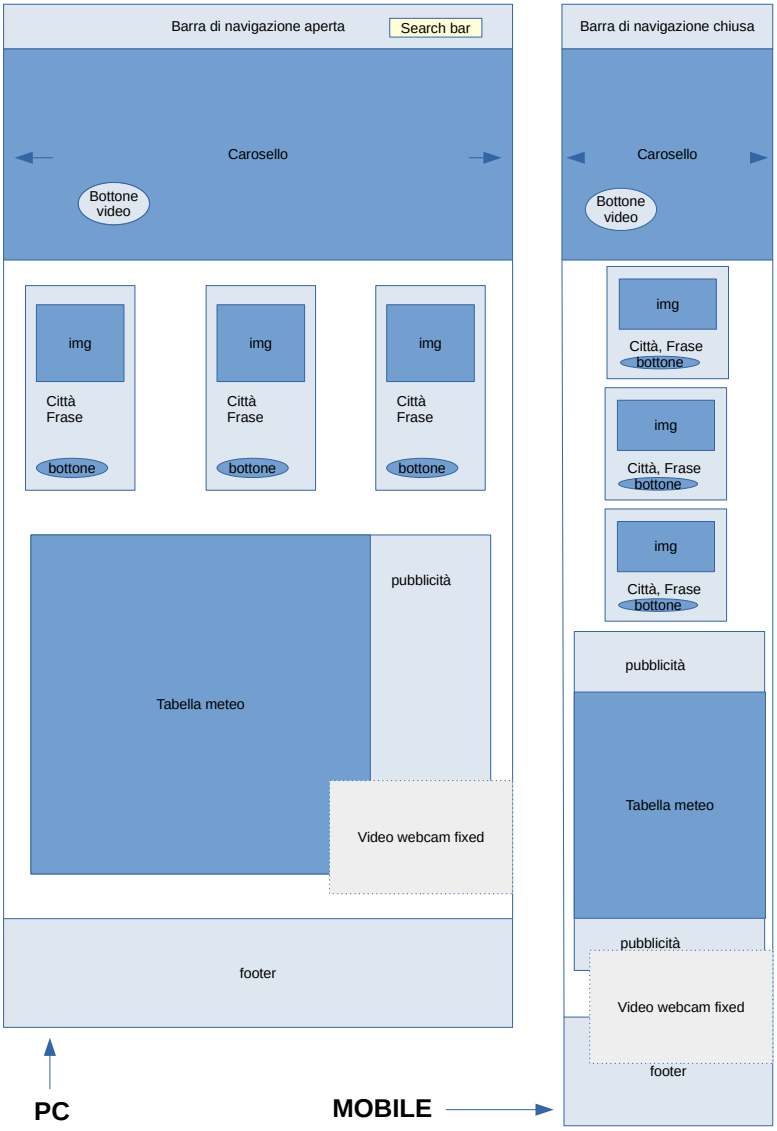
1.4 Aspetti tecnologici

L'applicazione segue il modello MVC; il controller è il cuore javascript dell'applicazione, è esso infatti che interagisce con il model (database dei 4 siti che forniscono le API) e aggiorna la view: struttura, creata con HTML 5, CSS3 e l'ausilio del framework bootstrap, il quale consente di definire una layout responsivo e flessibile. La comunicazione tra il controller e il model avviene utilizzando il formato JSON, che consente una trasmissione dei dati strutturata e facile da gestire.

2 Interfacce

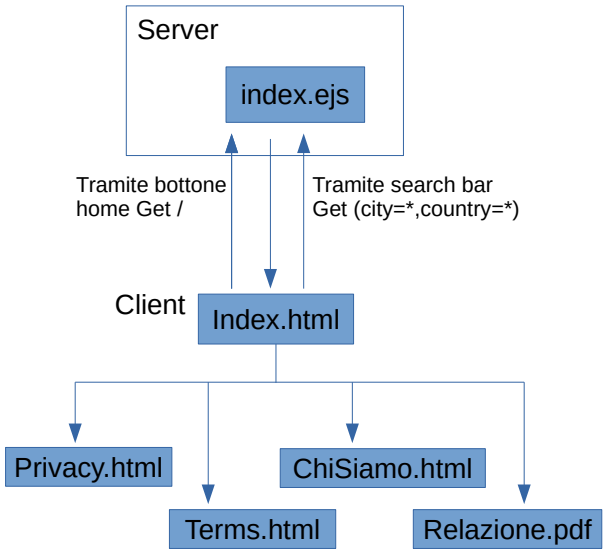
Il sito presenta un unica interfaccia principale, responsiva il cui codice può essere eseguito su dispositivi di ogni dimensione. I principali elementi della pagina sono cinque:

- la barra di navigazione, che può mostrarsi sia in versione completa da pc, in cui si riescono a osservare i collegamenti ad altre pagine, lo switch toggle della luminosità e la search bar, sia in versione compressa, nei dispositivi mobile, in cui tutti i vari elementi caratteristici sono nascosti;
- il carosello, che mantiene la sua dinamicità e tutte le sue immagini sia in versione desktop che mobile, in cui l'unica differenza sta nel modo in cui vengono adattate e ritagliate le immagini;
- tre carte che fanno riferimento ex-novo a 3 città nel mondo e che registrano le passate ricerche sostituendo le città di default. Il loro layout cambia da orizzontale a verticale nel passaggio da desktop a mobile;
- Un corpo che contiene una tabella con i dati del meteo circondata da banner pubblicitari, disposti a seconda delle dimensioni dello schermo
- in più si può osservare, a discrezione dell'utente, e in base alle disponibilità della località, un video in diretta o in timelapse che appare sul fondo della pagina in modalità fixed e dimensioni fissate, ma che viene rimpicciolito al restringersi della pagina.


Barra di navigazione aperta
Search bar
Carosello
Bottone video
img
Città
Frase
bottone
Tabella meteo
pubblicità
Video webcam fixed
footer
PC
Barra di navigazione chiusa
Città, Frase
MOBILE

3 Architettura

3.1 Diagramma dell'ordine gerarchico delle risorseDescrizione delle risorse



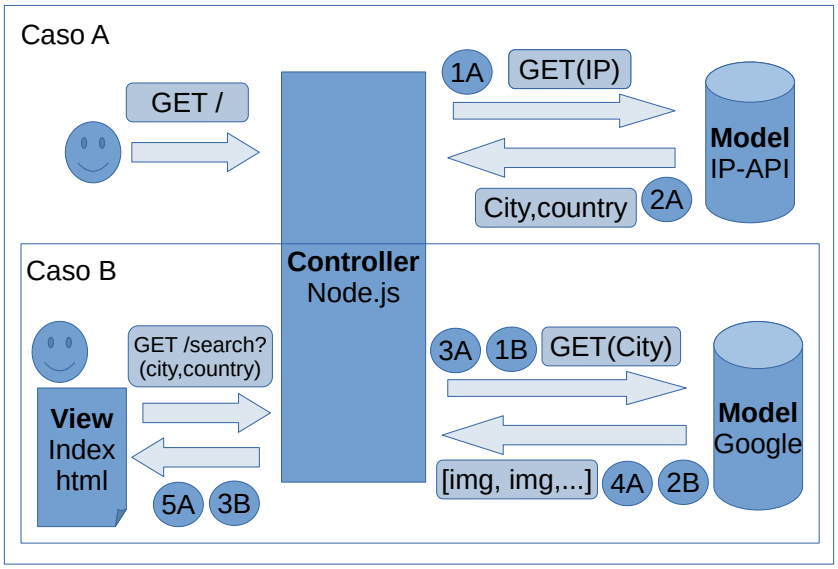
3.1.1 Cosa avviene lato server

caso A viene richiesta la pagina / allora

1. il server fa una get ad ip-api API e ottiene città e paese come risposta
2. il server usa la risposta per richiedere a se stesso la pagina /search?city=...&country=...

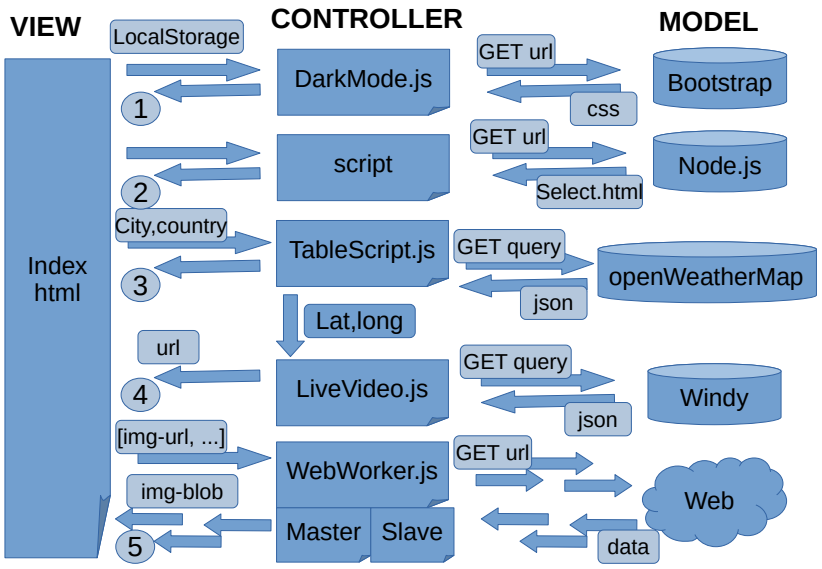
caso B, viene richiesta la pagina /search?city=...&country=... allora

1. il server fa una get a googleImages API e ottiene una lista di immagini
2. il server risponde al client con index.html, cioè index.ejs renderizzato con nome città, nome paese e immagini



3.1.2 Cosa avviene lato client

1. DarkModeScript.js carica il css corretto di boostrap e aggiorna la view
2. uno script embedded carica il form dei paesi dal server e aggiorna la view
3. TableScript.js fa una get ad openWeatherMap per prende i dati meteo e crea la tabella
4. latitudine e longitudine vengono usati per fare una get a windy e prendere l'indirizzo della telecamera da aggiungere alla pagina
5. WorkerSlave.js fa dei fetch delle immagini e il web worker master aggiorna la view



4 Prestazioni

Rispetto a una prima versione funzionante è stato fatto un efficientamento, il cui impatto sulle prestazioni può essere valutato nelle pagine che seguono.

Le principali strategie adottate per il miglioramento della pagina sono state l'uso di web worker per caricare le immagini in modo non bloccante e la riduzione del numero di accessi di rete fatti tramite le API, salvando direttamente tutti i dati del meteo direttamente nel DOM, senza doverli ricaricare ogni volta che venissero richieste nuove informazioni rispetto a quelle presentate o il meteo di giorni differenti. Tutto questo, per quanto possa sembrare un appesantimento del codice HTML, produce un miglioramento delle prestazioni e non impatta negativamente sulla pagina grazie, anche, all'utilizzo delle page visibility API di HTML5.

PRIMA

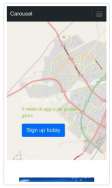
Diagnose performance issues

64

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100



METRICS — Collapse view

- First Contentful Paint — 1.8 s — First Contentful Paint marks the time at which the first text or image is painted. Learn more.
- Time to Interactive — 1.8 s — Time to interactive is the amount of time it takes for the page to become fully interactive. Learn more.
- Speed Index — 2.1 s — Speed Index shows how quickly the contents of a page are visibly populated. Learn more.
- Total Blocking Time — 0 ms — Sum of all time periods between FCP and Time to Interactive, when task length exceeded 50ms, expressed in milliseconds. Learn more.
- Largest Contentful Paint — 5.8 s — Largest Contentful Paint marks the time at which the largest text or image is painted. Learn more
- Cumulative Layout Shift — 0.667 — Cumulative Layout Shift measures the movement of visible elements within the viewport. Learn more.

Captured at Jul 7, 2022, 10:11 AM GMT+2 · Initial page load · Emulated Moto G4 with Lighthouse 9.6.2 · Slow 4G throttling · Single page load · Using HeadlessChromium 102.0.5005.115 with lr

View Treemap

Show audits relevant to: All FCP TBT LCP CLS

OPPORTUNITIES

Opportunity	Estimated Savings
Serve images in next-gen formats	51 s
Efficiently encode images	38.25 s
Properly size images	0.75 s
Reduce unused CSS	0.3 s
Enable text compression	0.15 s

These suggestions can help your page load faster. They don't directly affect the Performance score.

DIAGNOSTICS

- Avoid enormous network payloads — Total size was 16,334 KiB
- Serve static assets with an efficient cache policy — 8 resources found
- Image elements do not have explicit `width` and `height`
- Avoid an excessive DOM size — 1,069 elements
- First Contentful Paint (3G) — 3403 ms
- Avoid chaining critical requests — 5 chains found
- Keep request counts low and transfer sizes small — 18 requests • 16,334 KiB
- Largest Contentful Paint element — 1 element found
- Avoid large layout shifts — 5 elements found

More information about the performance of your application. These numbers don't directly affect the Performance score.

PASSED AUDITS (26) — Hide

- Eliminate render-blocking resources — Potential savings of 0 ms
- Defer offscreen images
- Minify CSS
- Minify JavaScript
- Reduce unused JavaScript
- Preconnect to required origins
- Initial server response time was short — Root document took 380 ms
- Avoid multiple page redirects
- Preload key requests
- Use video formats for animated content
- Remove duplicate modules in JavaScript bundles
- Avoid serving legacy JavaScript to modern browsers
- Preload Largest Contentful Paint image
- User Timing marks and measures
- JavaScript execution time — 0.1 s
- Minimizes main-thread work — 0.6 s
- All text remains visible during webfont loads
- Minimize third-party usage — Third-party code blocked the main thread for 0 ms
- Largest Contentful Paint image was not lazily loaded
- Uses passive listeners to improve scrolling performance
- Avoids `document.write()`
- Avoid long main-thread tasks
- Avoid non-composited animations

Mobile Desktop

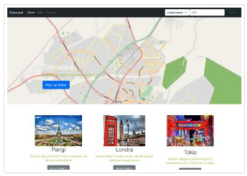
Diagnose performance issues

87

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100

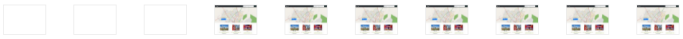


METRICS — Expand view

- First Contentful Paint — 0.5 s
- Time to Interactive — 0.5 s
- Speed Index — 0.8 s
- Total Blocking Time — 0 ms
- Largest Contentful Paint — 1.3 s
- Cumulative Layout Shift — 0.331

Captured at Jul 7, 2022, 10:11 AM GMT+2 · Initial page load · Emulated Desktop with Lighthouse 9.6.2 · Custom throttling · Single page load · Using HeadlessChromium 102.0.5005.115 with lr

View Treemap



Show audits relevant to: All FCP TBT LCP CLS

OPPORTUNITIES

Opportunity	Estimated Savings
Serve images in next-gen formats	8.16 s
Efficiently encode images	6.12 s
Properly size images	0.52 s

These suggestions can help your page load faster. They don't directly affect the Performance score.

DIAGNOSTICS

- Avoid enormous network payloads — Total size was 16,334 KiB
- Serve static assets with an efficient cache policy — 8 resources found
- Image elements do not have explicit `width` and `height`
- Avoid an excessive DOM size — 1,069 elements
- Avoid chaining critical requests — 5 chains found
- Keep request counts low and transfer sizes small — 18 requests • 16,334 KiB
- Largest Contentful Paint element — 1 element found
- Avoid large layout shifts — 5 elements found

More information about the performance of your application. These numbers don't directly affect the Performance score.

PASSED AUDITS (28) — Hide

- Eliminate render-blocking resources — Potential savings of 0 ms
- Defer offscreen images
- Minify CSS
- Minify JavaScript
- Reduce unused CSS — Potential savings of 24 KiB
- Reduce unused JavaScript
- Enable text compression — Potential savings of 13 KiB
- Preconnect to required origins
- Initial server response time was short — Root document took 380 ms
- Avoid multiple page redirects
- Preload key requests
- Use video formats for animated content
- Remove duplicate modules in JavaScript bundles
- Avoid serving legacy JavaScript to modern browsers
- Preload Largest Contentful Paint image
- User Timing marks and measures
- JavaScript execution time — 0.0 s
- Minimizes main-thread work — 0.2 s
- All text remains visible during webfont loads
- Minimize third-party usage — Third-party code blocked the main thread for 0 ms
- Lazy load third-party resources with facades
- Largest Contentful Paint image was not lazily loaded
- Uses passive listeners to improve scrolling performance
- Avoids `document.write()`
- Avoid non-composited animations
- Has a `<meta name="viewport">` tag with `width` or `initial-scale`
- Avoids `unload` event listeners

Mobile Desktop

https://pwb-project-git-main-g7240.vercel.app/search?city=Gaborone&country=BW

Analyze

DOPO

Discover what your real users are experiencing

No Data

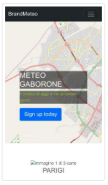
Diagnose performance issues

77

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

0–49 50–89 90–100



METRICS Expand view

First Contentful Paint 1.7 s	Time to Interactive 1.7 s
Speed Index 6.3 s	Total Blocking Time 0 ms
Largest Contentful Paint 2.5 s	Cumulative Layout Shift 0.724

Captured at Jul 11, 2022, 3:14 PM GMT+2 · Initial page load · Emulated Moto G4 with Lighthouse 9.6.2 · Slow 4G throttling · Single page load · Using HeadlessChromium 102.0.5005.115 with lr

View Treemap

Show audits relevant to: All FCP TBT LCP CLS

OPPORTUNITIES

Opportunity — Estimated Savings

Reduce unused CSS 0.45 s

Reduce unused rules from stylesheets and defer CSS not used for above-the-fold content to decrease bytes consumed by network activity. Learn more. FCP LCP

URL	Transfer Size	Potential Savings
…css/bootstrap.min.css (pwb-project-git-main-g7240.vercel.app)	26.4 KiB	24.6 KiB

Enable text compression 0.3 s

Text-based resources should be served with compression (gzip, deflate or brotli) to minimize total network bytes. Learn more. FCP LCP

URL	Transfer Size	Potential Savings
…2.5/forecast?q=GABORONE,BW&appid=c7b2c14…&lang=it&units=metric (api.openweathermap.org)	15.3 KiB	13.3 KiB

These suggestions can help your page load faster. They don't directly affect the Performance score.

DIAGNOSTICS

Image elements do not have explicit `width` and `height`

Set an explicit width and height on image elements to reduce layout shifts and improve CLS. Learn more CLS

Show 3rd-party resources (4)

URL
icona Clear id 800 `<img class="ring" src="https://openweathermap.org/img/wn/01d@2x.png" alt="icona Clear id 800">` …wn/01d@2x.png (openweathermap.org)
icona Clear id 800 `<img class="ring" src="https://openweathermap.org/img/wn/01n@2x.png" alt="icona Clear id 800">` …wn/01n@2x.png (openweathermap.org)
icona Clear id 800 `<img class="ring" src="https://openweathermap.org/img/wn/01n@2x.png" alt="icona Clear id 800">` …wn/01n@2x.png (openweathermap.org)
icona Clear id 800 `<img class="ring" src="https://openweathermap.org/img/wn/01n@2x.png" alt="icona Clear id 800">` …wn/01n@2x.png (openweathermap.org)
Immagine 1 di 3 carte `<img id="img_1" class="card-img-top" alt="Immagine 1 di 3 carte" style="max-height: 180px; width: auto" data-src="https://www.turistafaidate.it/wp-content/uploads/2016/10/Parigi-696x462.jpg">`
Immagine 2 di 3 carte `<img id="img_2" class="card-img-top" alt="Immagine 2 di 3 carte" style="max-height: 180px; width: auto" data-src="https://www.viaggilondra.it/wp-content/uploads/2016/04/meteo-londra.jpg">`
Immagine 3 di 3 carte `<img id="img_3" class="card-img-top" alt="Immagine 3 di 3 carte" style="max-height: 180px; width: auto" data-src="https://media-assets.vanityfair.it/photos/614d9c858292c15b12d9d1ad/3:2/w_1…">`

Avoid an excessive DOM size — 1,043 elements

A large DOM will increase memory usage, cause longer style calculations, and produce costly layout reflows. Learn more. TBT

Statistic	Element	Value
Total DOM Elements		1,043
Maximum DOM Depth	(*C) `<small class="text-muted">`	10
Maximum Child Elements	scegli paese Afghanistan Aland Islands Albania Algeria … `<select class="form-select" aria-label=".form-select-sm example" form="formid" name="country">`	253

Mobile Desktop

Avoid chaining critical requests — 10 chains found

Keep request counts low and transfer sizes small — 15 requests • 84 KiB

Largest Contentful Paint element — 1 element found

Avoid large layout shifts — 5 elements found

More information about the performance of your application. These numbers don't directly affect the Performance score.

https://pwb-project-git-main-g7240.vercel.app/search?city=Gaborone&country=BW

Analyze

Discover what your real users are experiencing

No Data

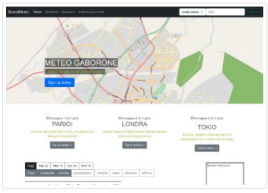
Diagnose performance issues

91

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

0–49 50–89 90–100



METRICS Expand view

First Contentful Paint 0.5 s	Time to Interactive 0.5 s
Speed Index 0.9 s	Total Blocking Time 0 ms
Largest Contentful Paint 0.6 s	Cumulative Layout Shift 0.299

Captured at Jul 11, 2022, 3:14 PM GMT+2 · Initial page load · Emulated Desktop with Lighthouse 9.6.2 · Custom throttling · Single page load · Using HeadlessChromium 102.0.5005.115 with lr

View Treemap

Show audits relevant to: All FCP TBT LCP CLS

DIAGNOSTICS

Image elements do not have explicit `width` and `height`

Set an explicit width and height on image elements to reduce layout shifts and improve CLS. Learn more CLS

Show 3rd-party resources (4)

URL
icona Clear id 800 `<img class="ring" src="https://openweathermap.org/img/wn/01d@2x.png" alt="icona Clear id 800">` …wn/01d@2x.png (openweathermap.org)
icona Clear id 800 `<img class="ring" src="https://openweathermap.org/img/wn/01n@2x.png" alt="icona Clear id 800">` …wn/01n@2x.png (openweathermap.org)
icona Clear id 800 `<img class="ring" src="https://openweathermap.org/img/wn/01n@2x.png" alt="icona Clear id 800">` …wn/01n@2x.png (openweathermap.org)
icona Clear id 800 `<img class="ring" src="https://openweathermap.org/img/wn/01n@2x.png" alt="icona Clear id 800">` …wn/01n@2x.png (openweathermap.org)
Immagine 1 di 3 carte `<img id="img_1" class="card-img-top" alt="Immagine 1 di 3 carte" style="max-height: 180px; width: auto" data-src="https://www.turistafaidate.it/wp-content/uploads/2016/10/Parigi-696x462.jpg">`
Immagine 2 di 3 carte `<img id="img_2" class="card-img-top" alt="Immagine 2 di 3 carte" style="max-height: 180px; width: auto" data-src="https://www.viaggilondra.it/wp-content/uploads/2016/04/meteo-londra.jpg">`
Immagine 3 di 3 carte `<img id="img_3" class="card-img-top" alt="Immagine 3 di 3 carte" style="max-height: 180px; width: auto" data-src="https://media-assets.vanityfair.it/photos/614d9c858292c15b12d9d1ad/3:2/w_1…">`

Avoid an excessive DOM size — 1,043 elements

A large DOM will increase memory usage, cause longer style calculations, and produce costly layout reflows. Learn more. TBT

Statistic	Element	Value
Total DOM Elements		1,043
Maximum DOM Depth	(*C) `<small class="text-muted">`	10
Maximum Child Elements	scegli paese Afghanistan Aland Islands Albania Algeria American Samoa Andorra A… `<select class="form-select" aria-label=".form-select-sm example" form="formid" name="country">`	253

Mobile Desktop

Largest Contentful Paint element — 1 element found

Avoid large layout shifts — 5 elements found

More information about the performance of your application. These numbers don't directly affect the Performance score.

PASSED AUDITS (33) Show

5 Codice

5.1 HTML

Figura 1: Scheletro del carosello che viene completato a runtime dal javascript

```
<div id="myCarousel" class="carousel slide" data-bs-ride="carousel">
    <div id="carousel-indicators" class="carousel-indicators">
      <!-- completato tramite js -->
    </div>
    <div id="carousel-inner" class="carousel-inner">
      <!-- completato tramite js -->
    </div>
    <button class="carousel-control-prev" type="button"
            data-bs-target="#myCarousel" data-bs-slide="prev">
      <span class="carousel-control-prev-icon" aria-hidden="true"></span>
      <span class="visually-hidden">Previous</span>
    </button>
    <button class="carousel-control-next" type="button"
            data-bs-target="#myCarousel" data-bs-slide="next">
      <span class="carousel-control-next-icon" aria-hidden="true"></span>
      <span class="visually-hidden">Next</span>
    </button>
  </div>
```

Figura 2: Uso della fivisione della pagina in griglie tramite bootstrap per definire tre carte dalla posizione dinamica e responsiva

```
<div class="container marketing">
 <!-- Three columns of text below the carousel -->
 <div class="row">
   <div class="col-lg-4">
     <img id="img_1" class="card-img-top" alt="Card image cap">
     <h2 id="h2_name1">Heading</h2>
     <p>Cerca in ogni posto del mondo, a qualsiasi ora del giorno, senza limiti</p>
     <p><a id="a_url1" class="btn btn-secondary" href="#">Vai al meteo &raquo;</a></p
   </div><!-- /.col-lg-4 -->
   <div class="col-lg-4">
     <img id="img_2" class="card-img-top" alt="Card image cap">
     <h2 id="h2_name2">Heading</h2>
     <p>Nessun posto è troppo lontano, lasciati guidare dalla tua immaginazione</p>
     <p><a id="a_url2" class="btn btn-secondary" href="#">Vai al meteo &raquo;</a></p
   </div><!-- /.col-lg-4 -->
   <div class="col-lg-4">
     <img id="img_3" class="card-img-top" alt="Card image cap">
     <div class="card-body">
       <h2 id="h2_name3">Heading</h2>
       <p>Esplora, viaggia e pianifica ogni tuo spostamento con il nostro sito di met
       <p><a id="a_url3" class="btn btn-secondary" href="#">Vai al meteo &raquo;</a><
     </div>
   </div><!-- /.col-lg-4 -->
 </div><!-- /.row -->
</div><!-- /.container -->
```

5.2 CSS

Figura 3: Codice per la gestione e il display delle tabelle

```
table{margin-top: 25px;}
table.B, table.C, table.D, table.E{ display: none; }
.precipitazioni, .visibil, .vento, .direzione, .raffiche{ /* .Tperc, .prob,  .umidit,*/
    visibility: collapse;
}
img.ring{
    border-radius: 50%;
    background: white;
    width: 1.5cm;
    height: auto;
}
```

Figura 4: Gestione dinamica delle dimensioni immagini delle carte anche quando lo scermo viene ristretto di molto

```
@media (min-width:350px){
    .card-img-top{
        height: 180px;
        width: auto
    }
}
@media (max-width: 350px){
    .card-img-top{
        max-height: 180px;
        object-fit: contain;
    }
}
```

Figura 5: Codice per gestire l'elemento video-live in maniera tale che sia nascosto, appaia in primo piano con una larghezza di 320px, si restringa in condizioni di spazio ridotto estremo senza mai però rompere l'iframe a cui è associato del php che richiede come dimensioni minime 200x110

```
#iframe{
    min-width: 202px;   /* Questo serve perchè altrimenti non funziona il video */
    min-height: 112px;  /*quindi meglio sproporzionato che rotto */
    max-width: 100%;
    width:320px;
    height: auto;

    border:1px solid black;
    position:fixed;
    right:0;
    bottom: 0;
    z-index: 1;
    display: none
}
```

Figura 6: CSS3 media query per la gestione dei box pubblicitari

```
@media (min-width:992px){
    div.content{
        width: calc(100% - 200px)
    }
    div.right-advertising{
        border: 5px solid gray;
        display: block;
        position: absolute;
        top: 0px;
        right: 0px;
        width: 200px;
        height: 100%;
    }
    div.right-advertising::after{
    content: "Spazio inserzioni";
    }
    div.advertising-box {
        display: none;
    }
}
@media (max-width:992px){
    div.advertising-box {
        display: block;
        width: 250px;
        height: 100px;

        padding: 10px;
        border: 5px solid gray;
        margin: 15px auto;/*center*/
    }
    div.advertising-box::after{content: "Spazio inserzioni";}
    div.right-advertising{
        display: none;
    }
}
```

5.3 JavaScript

Figura 7: Chiamata alle API di OpenWeatherMap e riempimento della tabella nel DOM costruendo riga per riga

```
function TableMenager(cit,coun){
    let url = 'https://api.openweathermap.org/data/2.5/forecast?q='+cit+','+coun+'&appi
    fetch(url).then(res => res.json())
    .then(o =>{
        if(o.cod=="200"){
            let i, j, tmp, data
            let count= 0
            for(i=0;i<o.cnt;i++){
                data= o.list[i].dt_txt.split(' ')[0]
                while(i<o.cnt && (data==o.list[i].dt_txt.split(' ')[0] || o.list[i].dt_t
                    // ... ho rimosso del codice per ragioni di spazio...
                    // inserisco i dati nella riga della tabella
                    let td11= document.createElement("td")
                    td11.textContent = o.list[i].wind.gust.toFixed('1')
                    let td12= document.createElement("td")
                    td12.textContent = o.list[i].main.humidity.toString()+'%'

                    let tr= document.createElement("tr")
                    // ... ho rimosso del codice per ragioni di spazio...
                    tr.appendChild(td9)
                    tr.appendChild(td10)
                    tr.appendChild(td11)
                    document.getElementById("table"+count).children[2].appendChild(tr)
                    i++
                }
                count++;
                if(count>=5){break}
            }
            //Cerco qualche video/immagine live della giornata
            LiveVideoMenager(o.city.coord.lat,o.city.coord.lon)
        }
        else{
            document.getElementById("meteoUnsuccesMessage").innerText= "Errore nel cari
            LiveVideoMenagerError()
        }
    })
    .catch((err) => {
        console.log(err)
        document.getElementById("meteoUnsuccesMessage").innerText= err
        LiveVideoMenagerError()
    });
}
```

5.4 API

Figura 8: Google immages API js object response


```
[
    {
        type: 'image/',
        width: 1000,
        height: 667,
        size: 148943,
        url: 'https://images.unsplash.com/photo-1543429776-2782fc8e1acd?ixlib=rb-1.2.1&i
        thumbnail: {
        url: 'https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTM8_TL1v1RUXrIC8Agxs
        width: 149,
        height: 99
        },
        description: 'Pisa Pictures | Download Free Images on Unsplash',
        parentPage: 'https://unsplash.com/s/photos/pisa'
    },
    {
        type: 'image/',
        width: 1000,
        height: 1339,
        size: 276874,
        url: 'https://images.unsplash.com/photo-1590226053097-e2984346ceaa?ixlib=rb-1.2.
        thumbnail: {
        url: 'https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcRiGdPCV3Z1LPYPbzQIq2
        width: 112,
        height: 150
        },
        description: '999+ Leaning Tower Of Pisa, Pisa, Italy Pictures | Download Free .
        parentPage: 'https://unsplash.com/s/photos/leaning-tower-of-pisa%2C-pisa%2C-ital
    },
    {
        type: 'image/',
        width: 479,
        height: 359,
        size: 92242,
        url: 'https://media.istockphoto.com/photos/leaning-tower-of-pisa-picture-id13666
        thumbnail: {
        url: 'https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcRNho8sfLAJSITHODeJfS
        width: 129,
        height: 97
        },
        description: 'Leaning Tower Of Pisa Pictures | Download Free Images on Unsplash'
        parentPage: 'https://unsplash.com/s/photos/leaning-tower-of-pisa'
    }
]
```

Figura 9: OpenWeatherMap example API json response


```
{
    "cod":"200","message":0,"cnt":40,"list":[
        {
            "main":{
                "temp":301.67,"feels_like":303.14,"humidity":58
            },
            "weather":[
                {
                    "description":"pioggia leggera",
                    "icon":"10d"
                }
            ],
            "clouds":{
                "all":40
            },
            "wind":{
                "speed":0.56,
                "deg":230,
                "gust":2.44
            },
            "visibility":10000,
            "pop":0.74,
            "rain":{
                "3h":1.29
            },
            "dt_txt":"2022-06-23 15:00:00"
        },
        {"dt":1656007200,"main":{"temp":300.39,"feels_like":301.87,"temp_min":297.82,
        "temp_max":300.39,"pressure":1012,"sea_level":1012,"grnd_level":995,
        "humidity":64,"temp_kf":2.57},"weather":[{"id":500,"main":"Rain",
        "description":"pioggia leggera","icon":"10d"}],"clouds":{"all":55},
        "wind":{"speed":0.97,"deg":25,"gust":3.33},"visibility":10000,"pop":0.74,
        "rain":{"3h":1.12},"sys":{"pod":"d"},"dt_txt":"2022-06-23 18:00:00"},
        {"dt":1656018000,"main":{"temp":297.62,"feels_like":298.11,"temp_min":295.59,"te
        {"dt":1656028800,"main":{"temp":293.35,"feels_like":293.78,"temp_min":293.35,"te
        {"dt":1656039600,"main":{"temp":292.96,"feels_like":293.35,"temp_min":292.96,"te
        {"dt":1656050400,"main":{"temp":293.28,"feels_like":293.65,"temp_min":293.28,"te
    ],
    "city":{
        "id":6542283,
        "name":"Comune di Milano",
        "coord":{"lat":45.4642,"lon":9.19},
        "country":"IT","population":0,"timezone":7200,"sunrise":1655955303,"sunset":1656
    }
}
```

Si tenga presente che le risposte delle API sono state tagliate e semplificate per ragioni di spazio e comprensibilità. In oltre non sono riportate le risposte delle API di Windy e di ip-api, per non appesantire troppo il documento.

5.5 Node.js

Figura 10: Uso di axios per geolocalizzare tramite ip-api API e reindirizzamento all'indirizzo https: //url/search?city=&country=

```
/** REINDIRIZZO DA INDEX A QUERY TRAMITE GEOLOCALIZZAZIONE IP */
app.set('trust proxy', true)
app.get('/', (req, res) => {
console.log("rispondo con tracciamento dell'ip",req.ip,"e reindirizzamento ad hoc")
var url = req.protocol + '://' + req.get('host') + req.originalUrl; //mi da l'url da do
const axios = require("axios");
// API
const URL = "http://ip-api.com/json/"+req.ip;

// Send a GET request to the API
axios.get(URL)
.then((resp) => {
    let cc= resp.data.countryCode
    let cty= resp.data.city
    res.statusCode = 302;
    res.setHeader("Location", url+"search?city="+cty+'&country='+cc);
    //res.send({"indirizzo": url+"/search?city="+cty+'&country='+cc})
    res.end();
})
.catch((error) => {
    console.log(error)
    res.send({"satatus":"IpGeolocAndataMale","risposta":error});//error.data
});

})
```

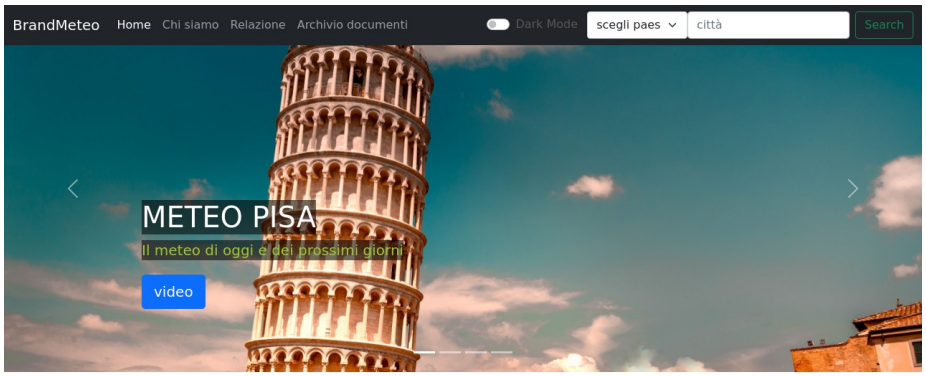
Figura 11: Servo pagine statiche

```
app.use(express.static(__dirname + '/public/'))
app.use('/favicon.ico', express.static(__dirname + '/public/images/immagine.png'));
app.get('/Terms', (req, res) => {
    console.log("rispondo con Terms")
    res.send("<h1>Terms page</h1><p>This page is a placeholder for a possible terms page
})
app.get('/error404', (req, res) => {
    console.log("rispondo con error 404")
    res.sendFile("./public/error404.html",{root:__dirname})
})
app.get('/Relazione', (req, res) => {
    console.log("rispondo con Relazione")
    res.sendFile("./public/Relazione.pdf",{root:__dirname})
})
```

6 Conclusioni

Il progetto in questione, per quanto funzionante e fruibile dal grande pubblico, è ancora migliorabile. Al momento la logica di ricerca si basa tutta sul conoscere città e paese di una località per poi disporne il meteo. Una funzionalità che potrebbe essere integrata è quella della geo-localizzazione lato client tramite il browser e le API HTML5; ciò non richiede alcuno sforzo per essere aggiunta tramite del codice javascript front-end, ma diventa invece dispendiosa la gestione della cosa lato server perché c'è bisogno di definire, in maniera additiva senza cambiare ciò che è stato già fatto fin ora, una logica che si basi sull'uso di latitudine e longitudine e dato l'utilizzo di API in maniera concatenata dove l'output di una, talvolta, è l'ingresso di un'altra, c'è da prestare attenzione. Questi saranno le future linee di sviluppo del progetto.

Ad ora possiamo comunque essere soddisfatti del funzionamento e apprezzare il risultato di una semplice query https://pwb-project.vercel.app/search?city=Pisa&country=IT



NEW YORK CITY

Cerca in ogni posto del mondo, a qualsiasi ora del giorno, senza limiti

Vai al meteo »

LONDRA

Nessun posto è troppo lontano, lasciati guidare dalla tua immaginazione

Vai al meteo »

TOKYO

Esplora, viaggia e pianifica ogni tuo spostamento con il nostro sito di meteo

Vai al meteo »

Oggi | Ven 22 | Sab 23 | Dom 24 | Lun 25

Tperc | probabilità | umidità | precipitazioni | visibilità | vento | direzione | raffiche

		descrizione	T(°C)	Tp(°C)	nubi	prob	umidità
21	●	cielo sereno	27.2	30.5	0%	0%	84%
00	●	cielo sereno	25.1	25.9	0%	0%	85%

Spazio inserzioni

Back to top