

**UNIVERSITÀ
DEGLI STUDI
DI PADOVA**

**DIPARTIMENTO DI FILOSOFIA, SOCIOLOGIA, PEDAGOGIA E PSICOLOGIA
APPLICATA**

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN COMUNICAZIONE

**Data visualization per supportare lo studio del
discorso su scienza e tecnologia nei quotidiani
online**

Relatore:

Prof. Emanuele Di Buccio

Laureando:

Sergio Brancato

2032341

Anno accademico 2023/2024

Abstract

La presente tesi è focalizzata sulla realizzazione di una homepage per la piattaforma web realizzata nell'ambito del progetto TIPS. Tale piattaforma è finalizzata all'analisi del discorso scientifico e tecnologico nei media digitali contemporanei. Il progetto TIPS nasce dalla necessità di comprendere come la scienza e la tecnologia vengano rappresentate e discusse nei quotidiani online, proponendo una chiave di lettura sociologica del rapporto tra media, scienza e società e offrendo uno strumento che trasforma grandi quantità di dati testuali in visualizzazioni intuitive e significative, consentendo l'esplorazione di tendenze e pattern nei contenuti mediatici. Il progetto documentato in questa tesi è stato realizzato sfruttando moderne tecnologie web per garantire funzionalità ed efficacia visiva. La tesi, inoltre, si concentra sull'utilizzo combinato di metodi di data visualization e di media monitoring, per dimostrare come delle efficaci rappresentazioni visive possano rendere più accessibili informazioni particolarmente complesse a diverse tipologie di utenti: dai ricercatori accademici agli utenti comuni. Il progetto TIPS si propone, quindi, di offrire una piattaforma che permette di automatizzare e facilitare il monitoraggio e l'analisi dei media presenti online, offrendo anche nuove prospettive sulla comunicazione scientifica. L'obiettivo finale della tesi è proprio quello di gettare le basi per un nuovo design front end della piattaforma, che faciliti l'implementazione di componenti di User Interface efficaci.

Indice

Abstract	2
Indice	3
Elenco delle figure	4
Elenco dei codici	6
1 Introduzione	7
2 Scienza e Tecnologia nei Media	9
2.1 Media monitoring	9
2.2 Data visualization	11
2.3 Progetto TIPS	12
3 TIPS Homepage	15
3.1 Obiettivi delle tesi	15
3.2 Tecnologie utilizzate	17
3.2.1 Next.js	17
3.2.2 Tailwind CSS	21
3.2.3 D3.js	23
3.2.4 Nivo	24
3.3 Realizzazione della Homepage	28
3.3.1 Header	29
3.3.2 Navigation sidebar	30
3.3.3 Scroll indicator	31
3.3.4 Home section	36
3.3.5 About section	39
3.3.6 Platform	40
3.3.7 People	57
3.3.8 Publications	60
3.3.9 Annual reports	65
3.3.10 Contacts	74
4 Conclusioni e sviluppi futuri	79
Ringraziamenti	81
Bibliografia	83
Sitografia	84

Elenco delle figure

Figura 1: screenshot sito su IphoneSE _____	16
Figura 2: architettura e processi del client-side-rendering _____	19
Figura 3: architettura e processi del server-side-rendering _____	20
Figura 4: rappresentazione visiva del DOM _____	23
Figura 5: Grafico di esempio costruito utilizzando Nivo _____	25
Figura 6: Grafico di esempio costruito utilizzando D3.js _____	27
Figura 7: nuova versione dell' header della homepage _____	29
Figura 8: vecchia versione dell'header della homepage _____	29
Figura 9: Navigation sidebar, visualizzazione desktop _____	30
Figura 10: confronto della visualizzazione mobile della barra di navigazione nelle due versioni _____	31
Figura 11: confronto dello scroll indicator nelle due versioni _____	32
Figura 12: home section nella vecchia versione _____	37
Figura 13: home section nella nuova versione _____	37
Figura 14: confronto tra le due versioni di home su dispositivo mobile Samsung Galaxy S20 Ultra _____	38
Figura 15: about section nella vecchia versione _____	39
Figura 16: about section nella nuova versione _____	40
Figura 17: platform section nella vecchia versione _____	41
Figura 18: scheda testuale della sezione platform della nuova versione _____	42
Figura 19: line chart costruito con Nivo _____	43
Figura 20: grafico general framing creato con D3.js _____	45
Figura 21: radar chart realizzato con la libreria Nivo _____	50
Figura 22: radar chart visualizzato su Pixel 7 _____	54
Figura 23: sezione people nella vecchia versione _____	57
Figura 24: sezione people nella nuova versione _____	57
Figura 25: confronto della visualizzazione mobile della sezione people _____	58
Figura 26: sezione publications nella vecchia versione _____	61
Figura 27: sezione publications nella nuova versione _____	61
Figura 28: confronto delle versioni della sezione publications visualizzata su mobile _____	65
Figura 29: sezione annual reports nella vecchia versione _____	66
Figura 30: sezione annual reports nella nuova versione _____	66
Figura 31: confronto delle versioni della sezione annual reports visualizzata su mobile _____	70
Figura 32: pagina annual reports nella vecchia versione _____	73

Figura 33: pagina annual reports nella nuova versione	73
Figura 34: sezione contacts nella vecchia versione	74
Figura 35: sezione contacts nella nuova versione	75

Elenco dei codici

Codice 1: esempio di codice sorgente di un'applicazione React	19
Codice 2: esempio di codice sorgente completo consegnato all'utente dal server	20
Codice 3: esempio di codice per visualizzare un'intestazione utilizzando vanilla CSS	22
Codice 4: esempio di codice per visualizzare un'intestazione utilizzando Tailwind CSS	22
Codice 5: esempio di codice di un grafico a barre utilizzando la libreria Nivo	25
Codice 6: esempio di codice di un grafico a barre utilizzando la libreria D3.js	27
Codice 7: rappresentazione della logica dello scroll indicator nella vecchia versione	33
Codice 8: componente scrollindicator.js	34
Codice 9: componente linechart.js	43
Codice 10: snippet di codice per calcolare le lunghezze delle barre in general framing	46
Codice 11: componente D3Graph.js	47
Codice 12: componente radarchart.js	51
Codice 13: funzione per ridimensionare il testo nei grafici Nivo	52
Codice 14: componente platform.js	54
Codice 15: componente people.js	59
Codice 16: componente publications.js	62
Codice 17: componente report.js	67
Codice 18: layout completo della pagina	71
Codice 19: codice completo di page.js	75

1 Introduzione

L'attuale contesto storico è caratterizzato da un rapido e incessante sviluppo scientifico e tecnologico in cui si assiste ad una trasformazione continua e profonda delle dinamiche sociali, economiche e culturali. Le innovazioni mediche, tecniche e scientifiche stanno ridefinendo costantemente il modo di vivere degli individui.

Tuttavia il pubblico potrebbe riscontrare serie difficoltà nel rimanere aggiornato e nel comprendere appieno la complessità delle informazioni scientifiche diffuse quotidianamente nei media di massa. Per gli studiosi risulta quindi particolarmente importante capire come le persone percepiscono e interpretano le informazioni.

In questo contesto, i media, e in particolare i quotidiani online, rappresentano una delle principali fonti di informazione, contribuendo in maniera determinante a plasmare la percezione pubblica, rendendo particolarmente importante comprendere il modo in cui i media digitali rappresentano determinate tematiche per capire l'influenza che esercitano sull'opinione pubblica. Tuttavia, l'enorme quantità di articoli, opinioni e dibattiti prodotti rende difficile un'analisi esaustiva e approfondita del discorso mediatico.

Per risolvere questo problema bisogna congiungere l'applicazione di due campi altamente complementari tra loro: il media monitoring e la data visualization, che, se vengono utilizzati insieme, forniscono degli strumenti potenti che consentono di analizzare il discorso mediatico, rendendo accessibili grandi quantità di dati ai consumatori.

Abbinando infatti gli approcci del media monitoring e della data visualization, viene migliorata l'efficacia del monitoraggio del discorso scientifico e tecnologico nei media online e viene rilevato come la scienza e la tecnologia vengano rappresentate nel panorama mediatico. Entrambe gli approcci offrono, inoltre, un contributo essenziale all'elaborazione di strategie comunicative e politiche, fornendo una base empirica solida per comprendere come il pubblico acquisisce e interpreta le informazioni scientifiche.

Queste tematiche sono al centro del progetto TIPS, acronimo di *Technoscientific Issues in the Public Sphere*. TIPS è un'iniziativa di ricerca interdisciplinare che coinvolge studiosi provenienti da molteplici ambiti disciplinari, tra cui informatica, ingegneria, sociologia,

statistica, psicologia e linguistica. L'obiettivo primario del progetto TIPS è l'analisi sistematica della presenza e della rappresentazione della tecnoscienza nei media di massa, considerati come componente fondamentale dell'opinione pubblica contemporanea.

Il sistema TIPS si avvale di metodologie automatizzate avanzate per assistere utenti esperti - inclusi ricercatori, giornalisti e policy maker - nel monitoraggio e nell'analisi dell'opinione pubblica su vasti corpus di dati eterogenei. L'eterogeneità è dovuta al fatto che i dati provengono da una molteplicità di fonti, tra cui blog, pubblicazioni scientifiche e quotidiani. Una delle sfide metodologiche principali affrontate dal progetto risiede nella gestione e nell'integrazione di dati provenienti da tipologie di fonti così diversificate.

Sebbene il sistema TIPS sia stato concepito e sviluppato per affrontare questa sfida di eterogeneità dei dati, l'obiettivo centrale del progetto va oltre la mera gestione tecnica. L'interesse principale risiede, infatti, nel fornire agli utenti esperti un set di strumenti analitici sofisticati ma di semplice utilizzo che supportino l'implementazione di metodologie di indagine personalizzate finalizzate all'attività di ricerca sul discorso scientifico e sulla sua rappresentazione nella sfera pubblica.

L'intento della tesi è proprio quello di realizzare una landing page moderna e responsive che gli utenti possano consultare per accedere alla piattaforma, offrendo un contesto applicativo concreto per l'integrazione di tecniche di media monitoring e data visualization che verranno analizzate più nel dettaglio.

Nel corso della tesi verranno introdotti i concetti di media monitoring e data visualization, spiegandone la loro importanza, successivamente verrà presentato il progetto TIPS e i suoi obiettivi. Verrà poi commentato il processo di creazione della nuova homepage, chiarendo prima le tecnologie utilizzate, e poi confrontando i risultati ottenuti con la vecchia versione, spiegando la logica di base di alcuni componenti. Infine il tutto sarà concluso da una sintesi del lavoro svolto e da considerazioni personali.

2 Scienza e Tecnologia nei Media

In un contesto mediatico come quello attuale diventa sempre più importante saper comprendere ed interpretare al meglio le informazioni presenti nei media di massa. Tuttavia, la raccolta dati seguita da analisi come quelle basate sull'analisi di occorrenze di specifici termini, potrebbe non essere sufficiente per avere una visione completa delle informazioni presenti online. L'esplosione dei contenuti digitali ha creato una vera e propria valanga informativa: ogni secondo, migliaia di nuovi post, articoli e commenti inondano il web, ponendo sfide significative per chi deve monitorare e analizzare questi flussi di informazione. L'obiettivo del progetto TIPS è proprio quello di utilizzare il media monitoring e la data visualization, avvalendosi di metodi computazionali consolidati o innovativi, per permettere a ricercatori e professionisti di comprendere al meglio il panorama mediatico. Uno degli approcci utilizzati per conseguire tale obiettivo è la definizione di indicatori, resi accessibili tramite una piattaforma facilmente utilizzabile anche da utenti non esperti in ambito informatico, e in grado di supportare ricerche ed analisi in maniera automatizzata su un'ampia base di dati.

2.1 Media monitoring

Il media monitoring consiste nell'osservazione, raccolta e analisi dei contenuti presenti nei media di massa.

Il panorama mediatico contemporaneo sta attraversando una trasformazione senza precedenti e adesso la complessità del media monitoring non risiede solamente nel volume massiccio dei dati da elaborare. La vera sfida emerge dalla necessità di interpretare correttamente questa moltitudine di voci, opinioni e prospettive che si intrecciano attraverso piattaforme diverse. Istituzioni e aziende si trovano davanti alla necessità di sviluppare strategie sempre più sofisticate per gestire questa complessità di dati da analizzare.

L'analisi dei contenuti mediatici richiede una particolare attenzione metodologica. Gli approcci tradizionali erano basati sul campionamento di sottoinsiemi di articoli correlati a determinati argomenti, che sebbene possano risultare ancora utili in determinate circostanze, mostrano limiti significativi nell'era dei big data. La selezione di

sottoinsiemi di contenuti rischia infatti di fornire una visione parziale e potenzialmente distorta del fenomeno studiato.

Un esempio emblematico riguarda l'analisi dell'attenzione mediatica nel tempo. L'aumento del numero di articoli su un determinato argomento non può essere interpretato isolatamente, ma deve essere contestualizzato nell'evoluzione complessiva della produzione editoriale. Solo considerando il quadro generale è possibile trarre conclusioni significative sulle reali dinamiche dell'attenzione mediatica, (Neresini, 2017).

Questo aspetto diventa particolarmente rilevante nell'ambito della comunicazione scientifica e tecnologica. Limitare l'analisi alle sole sezioni tematiche specializzate, (come ad esempio le sezioni "Scienze" o "Tecnologia"), significherebbe perdere di vista le numerose interconnessioni che questi temi sviluppano con altri ambiti del dibattito pubblico. La tecnologia e la scienza permeano infatti molteplici aspetti della vita sociale, economica e culturale.

Monitorare sistematicamente i contenuti presenti nei media di massa, si rivela quindi uno strumento essenziale per la ricerca sulla comunicazione scientifica. Il vantaggio principale è che fornisce una base empirica solida per comprendere come il pubblico percepisce e interpreta le questioni tecnoscientifiche, facilitando lo sviluppo di un dialogo costruttivo tra comunità scientifica e società civile.

Il monitoraggio sistematico di fonti eterogenee, ad esempio articoli di quotidiani pubblicati online o post di social media, richiede di affrontare una serie di problemi legati alla memorizzazione, all'accesso ed all'analisi di dati non strutturati. Un esempio sono gli approcci per identificare automaticamente le tematiche trattate in un articolo o in un insieme di articoli, e la classificazione automatica degli articoli in base ai contenuti trattati; tale classificazione può complementare quella fatta nei quotidiani, ad esempio "l'assegnazione" di un articolo a determinate sezioni dei quotidiani. Un articolo rilevante a tematiche tecnoscientifiche potrebbe essere anche pubblicato in una sezione diversa da "Scienze" o "Tecnologia", ad esempio in una sezione relativa all'economia in quanto l'articolo tratta dell'impatto di nuove scoperte o tecnologie in quell'ambito.

La sfida del media monitoring nell'era digitale richiede dunque un approccio metodologico rigoroso, strumenti analitici avanzati e una visione ampia che sappia

cogliere la complessità delle dinamiche comunicative contemporanee. Solo attraverso questa prospettiva integrata sarà possibile sviluppare una comprensione profonda del ruolo dei media nella costruzione del dibattito pubblico su scienza e tecnologia, obiettivo perseguito dal progetto TIPS tramite il suo portale che mira ad analizzare in maniera specifica e completa tutto il panorama dei quotidiani online.

Bisogna però tenere a mente che il media monitoring da solo potrebbe non essere sufficiente per supportare efficacemente attività cruciali per la comunità scientifica: le analisi svolte devono essere visualizzate in maniera chiara e intuitiva; avvalendosi quindi, di tecniche di data visualization.

2.2 Data visualization

La data visualization consiste in una serie di metodi per rappresentare visivamente informazioni tramite grafici in maniera chiara e facilmente comprensibile. Secondo Kieran Healy (2018), la visualizzazione dei dati è fondamentale non solo per l'analisi quantitativa ma anche per la comunicazione efficace dei risultati, poiché "buone visualizzazioni facilitano la comunicazione delle idee e delle scoperte".

La data visualization gioca un ruolo fondamentale in un contesto storico in cui la quantità di dati generati cresce esponenzialmente ogni anno. Con questa mole imponente di dati, risulta fondamentale per il successo in vari settori, essere capaci di sintetizzare le informazioni per comunicarle efficacemente.

Gli strumenti utili alla data visualization possono essere divisi in tre macro categorie: fogli di calcolo, software di visualizzazione e librerie di programmazione.

I fogli di calcolo, come Microsoft Excel e Google Sheets, rappresentano una delle soluzioni più comuni in vari settori per la visualizzazione e gestione dei dati. Questi strumenti offrono funzionalità di base per creare grafici semplici, come grafici a linee e a dispersione, che sono spesso sufficienti per esigenze meno complesse.

Il software di visualizzazione di dati è invece progettato specificamente per consentire un'analisi visiva più avanzata. Piattaforme come Tableau e PowerBI permettono delle analisi approfondite, rispondendo solitamente a esigenze analitiche più complesse. Avendo un costo maggiore rispetto ai normali fogli di calcolo, sono spesso adottate da realtà che necessitano di tool intuitivi per effettuare e automatizzare determinate analisi.

Per questa tesi sono state usate due librerie di Javascript, ovvero D3.js e Nivo, che verranno successivamente approfondite. Le librerie offrono un approccio più flessibile e personalizzabile per la visualizzazione dei dati, richiedendo però, delle competenze tecniche precise. Rispetto agli altri strumenti, le librerie consentono di gestire completamente le modalità di visualizzazione dei dati, questo approccio permette di condurre analisi specifiche e mostrarne i risultati su piattaforme web e software proprietari.

2.3 Progetto TIPS

Il progetto di ricerca TIPS (Technoscientific Issues in the Public Sphere), è parte delle attività di ricerca del gruppo Pa.S.T.I.S. e rappresenta una risposta concreta alle sempre più complesse sfide lanciate al campo della comunicazione scientifica nell'era digitale. Il fine ultimo a cui tende il progetto è lo sviluppo, l'implementazione e la sperimentazione di metodologie automatizzate per la gestione e l'analisi dei contenuti digitali che supportano il monitoraggio dell'evoluzione dei temi scientifici e tecnologici all'interno del dibattito pubblico. Il conseguimento di tale obiettivo avviene tramite l'integrazione di diverse prospettive teoriche e metodologiche.

Il Public Communication of Science and Technology (PCTS), rappresenta un campo accademico e professionale dedicato alla comunicazione efficace della scienza e della tecnologia nei confronti del grande pubblico. Si concentra su come le informazioni scientifiche vengono trasmesse e comprese dalla società.

Parallelamente gli Studi Sociali della Scienza (STS), sono un campo interdisciplinare che esamina come la scienza e la tecnologia influenzano e sono influenzate dalla società. Questo ambito di studio analizza le pratiche scientifiche, le istituzioni, i processi di innovazione e le implicazioni sociali delle tecnologie emergenti.

Proprio la maggiore forza del progetto è rappresentata dalla capacità di integrare competenze diverse, ossia dall'impegno di proporre le ricerche ad un team di ricercatori con un background eterogeneo e specifiche competenze legate alle discipline coinvolte nel progetto.

Le applicazioni di questo sistema di monitoraggio spaziano dall'analisi delle tendenze comunicative all'identificazione di pattern emergenti nel dibattito pubblico, fornendo così strumenti preziosi per la ricerca accademica. La rilevanza del progetto TIPS si

manifesta anche nella sua capacità di adattarsi all'evoluzione continua del panorama mediatico, offrendo soluzioni innovative per affrontare le sfide della comunicazione scientifica contemporanea attraverso l'integrazione di tecnologie avanzate e metodologie di ricerca consolidate. Ciò è reso possibile anche grazie ai *Digital Methods*, che rappresentano un approccio innovativo nell'analisi dei dati e nella ricerca sociale, specialmente nel contesto della comunicazione scientifica. Questi metodi si concentrano sull'uso di strumenti digitali per raccogliere, analizzare e visualizzare dati provenienti dalle interazioni online, come quelle sui social media, forum e altre fonti digitali.

Il progetto TIPS si configura come un modello pionieristico per lo studio della comunicazione scientifica nell'era digitale, contribuendo significativamente alla comprensione delle dinamiche comunicative e alla gestione delle sfide sempre più complesse poste sulle questioni tecnoscientifiche del dibattito pubblico; la natura interdisciplinare del progetto unita alla sua capacità di integrare diverse prospettive metodologiche, lo rende uno strumento essenziale per il monitoraggio e l'analisi della comunicazione scientifica contemporanea, aprendo nuove prospettive per la ricerca futura e offrendo contributi significativi sia alla comunità accademica sia a professionisti che hanno a che fare con la comunicazione pubblica della scienza; il tutto, tra l'altro, non richiede alcun tipo di competenza informatica specifica da parte degli utenti, che potranno svolgere svariate attività di ricerca mentre dietro le quinte vengono adottati approcci all'avanguardia nell'analisi dei contenuti, come recenti tecniche proposte nell'ambito dell'elaborazione del linguaggio naturale (Natural Language Processing, NLP).

Tecniche proposte nell'ambito del NLP e del reperimento dell'informazione (Information Retrieval, IR) sono state adottate in TIPS per definire e calcolare automaticamente degli indicatori basati sul contenuto (testo) degli articoli. Alcuni di tali indicatori, come quello relativo al *rischio*, hanno l'obiettivo di misurare il grado in cui un concetto (ad esempio quello di rischio) è presente in un documento o in un insieme di documenti. In questo modo, è possibile caratterizzare ciascun articolo non solo in base alla sua pertinenza ad una specifica tematica, ad esempio la scienza, ma anche in base al grado in cui il suo contenuto "evoca" il concetto di rischio. L'indicatore rischio è solo uno degli indicatori disponibili in TIPS. Un altro indicatore è la *salienza* che indica la proporzione di articoli che trattano tematiche tecnoscientifiche, sul totale di articoli pubblicati in un determinato periodo temporale. Analizzando il trend della salienza è

possibile comprendere se c'è una maggior attenzione rispetto a tematiche tecnoscientifiche in determinati periodi storici.

3 TIPS Homepage

In questo capitolo verrà descritta la piattaforma progettata e sviluppata per il progetto TIPS, focalizzandosi sia sui motivi che hanno portato alla sua realizzazione, sia sugli “strumenti” – ad esempio librerie software – utilizzati per implementarla.

3.1 Obiettivi delle tesi

L’obiettivo della tesi è stato quello di contribuire al progetto TIPS attraverso la realizzazione, tramite tecnologie all’avanguardia, di una homepage funzionale e moderna che integrasse visualizzazioni di dati efficaci ed utili per fini di ricerca. L’interfaccia del sito è stata proprio progettata per essere intuitiva e responsiva, permettendo a utenti diversi, di navigare facilmente tra i contenuti e accedere alle funzionalità principali del progetto. Questi contributi hanno permesso di arricchire il progetto TIPS con una piattaforma che unisce estetica, funzionalità e capacità analitiche; rappresentando così un esempio concreto dell’applicazione delle tecnologie digitali nello studio della comunicazione su scienza e tecnologia.

Questo lavoro si inserisce nel contesto delle attività del gruppo di ricerca Pa.S.T.I.S.,¹ che indaga le dinamiche comunicative legate alla scienza e alla tecnologia nella società contemporanea.

Il progetto TIPS disponeva già di una pagina web che fungeva da home page. Tuttavia è stata realizzata utilizzando componenti di Angular (un framework Javascript del 2016) ormai datati, risultando in un sito non eccessivamente gradevole alla vista, non molto curato nei dettagli soprattutto per quanto riguarda la visualizzazione su dispositivi mobili con schermi più piccoli.

¹ <https://www.pastis-research.eu/>



Figura 1: screenshot sito su IphoneSE

Nella schermata riportata si può notare un problema relativo alla sovrapposizione di vari elementi se viene visualizzato il sito su un dispositivo mobile (in questo caso specifico si tratta di un iPhone SE ma potrebbe essere sostituito con un qualsiasi dispositivo con la stessa risoluzione schermo).

Uno dei problemi di un design obsoleto o datato è che potrebbe risultare poco professionale e scoraggiare utenti e ricercatori a utilizzare la piattaforma nonostante disponga di validissime funzionalità; bisogna quindi tenere a mente l'importanza della user experience quando si crea un sito, ovvero l'interazione complessiva dell'utente con un portale online.

Considerando inoltre che nel 2024 più del 60% del traffico globale è generato da dispositivi mobili si può intuire quanto sia importante per un sito essere *responsive* (GilPress 2024), soprattutto considerando che questa percentuale è destinata ad aumentare considerevolmente dato che appena dieci anni fa, nel 2014, la percentuale di traffico gestito da smartphone e tablet era solo del 29%.

Parallelamente sono aumentate le qualità e le prestazioni delle connessioni, portando quindi a utenti sempre più esigenti per quanto riguarda la velocità di caricamento di un qualsiasi applicativo presente online.

Per tutti questi problemi verranno presentate soluzioni con l'obiettivo di creare una nuova home page funzionale e visivamente gradevole.

3.2 Tecnologie utilizzate

Per risolvere le problematiche discusse nel capitolo precedente sono state applicate varie tecnologie moderne che permettessero di trovare soluzioni facilmente implementabili e personalizzabili nel tempo. Tuttavia prima di parlare dei vari tool impiegati nella realizzazione della homepage è necessario chiarire cosa si intende per libreria e cosa si intende per framework.

Una libreria è un insieme di programmi progettati, scritti e resi disponibili da altri sviluppatori. Una libreria permette di eseguire compiti specifici in maniera semplice senza dover implementare nuovamente da zero una funzione. Quando si usa una libreria lo sviluppatore ha pieno controllo su come strutturare l'applicazione e su come e quando richiamare le funzioni della libreria.

Un framework, dall'altra parte, è una vera e propria infrastruttura di base su cui sviluppare la propria applicazione, ne stabilisce l'architettura controllando il flusso di esecuzione; lo sviluppatore si inserisce quindi all'interno di questa struttura per creare il proprio applicativo. In linea generale, un framework ha un maggior livello di complessità di una "semplice" libreria.

Il punto chiave è che mentre quando si usa una libreria è il programmatore che chiama indipendentemente le funzioni, il framework, invece, richiama il codice che lo sviluppatore inserisce in punti specifici del software nei momenti appropriati, gestendo autonomamente il flusso di esecuzione dell'applicazione. Questa differenza è definita come *inversion of control* o "IoC", perché viene invertito il ruolo tradizionale che si ha con una libreria, dove è il programmatore a controllare tutto il flusso..

3.2.1 Next.js

Sicuramente l'innovazione più importante apportata alla precedente versione della homepage del progetto TIPS è stata quella di utilizzare il framework Next.js.²

² <https://nextjs.org/>

Next.js è un framework basato su React per la creazione di applicazioni web full-stack. React.js³ è la libreria Javascript più utilizzata tra gli sviluppatori, permette di creare interfacce utente dinamiche e interattive tramite l'utilizzo di *components*, ovvero componenti riutilizzabili costituiti da blocchi di codice autonomi. Questo approccio modulare permette di costruire applicazioni web complesse mantenendo il codice organizzato e facilmente manutenibile.

Next.js quindi, estende significativamente le funzionalità di React, rendendolo un potente e versatile strumento supportato da un'ampia community di sviluppatori in tutto il mondo; non a caso questo framework viene utilizzato da aziende come Spotify, OpenAI, Nike, LG, Vodafone e molte altre per la creazione delle loro web-app.

Una delle principali innovazioni che apporta Next.js è la capacità di supportare sia il *Server-Side-Rendering* (SSR), che il *Client-Side-Rendering* (CSR). Sono entrambi validi modi che permettono all'utente finale di visualizzare correttamente la pagina. La differenza tra i due sta nel modo in cui questo avviene: mentre il Server-Side-Rendering renderizza le pagine direttamente sul server prima di inviarle, già complete, al browser dell'utente, il Client-Side-Rendering si basa invece sul browser stesso che, eseguendo localmente il codice Javascript, restituisce la pagina finale all'utente

³ <https://react.dev/>

CLIENT-SIDE RENDERING

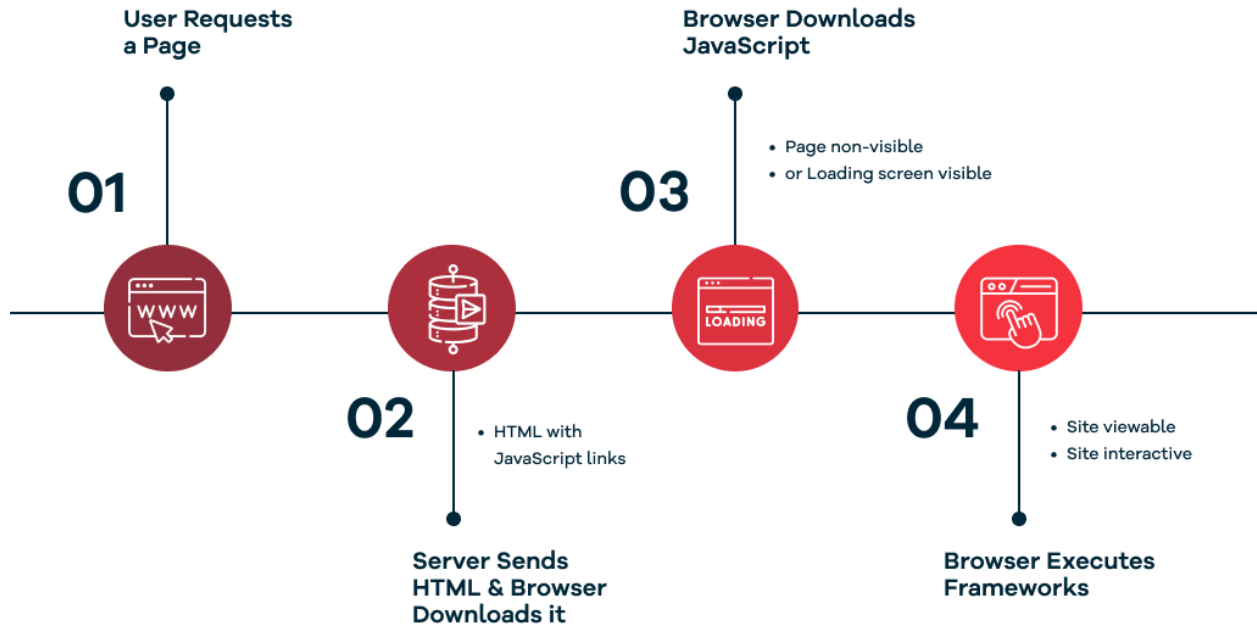


Figura 2: architettura e processi del client-side-rendering

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Pagina caricata tramite CSR</title>
</head>
<body>
  <div id="root">
    <app></app>
  </div>
  <script src="location/of/app.js" type="text/javascript"></script>
</body>
</html>
```

Codice 1: esempio di codice sorgente di un'applicazione React

Il codice presente nel browser di una web-app scritta utilizzando React avrà la struttura riportata in Codice 1: il browser eseguirà autonomamente il codice Javascript presente all'interno del componente `<app></app>`.

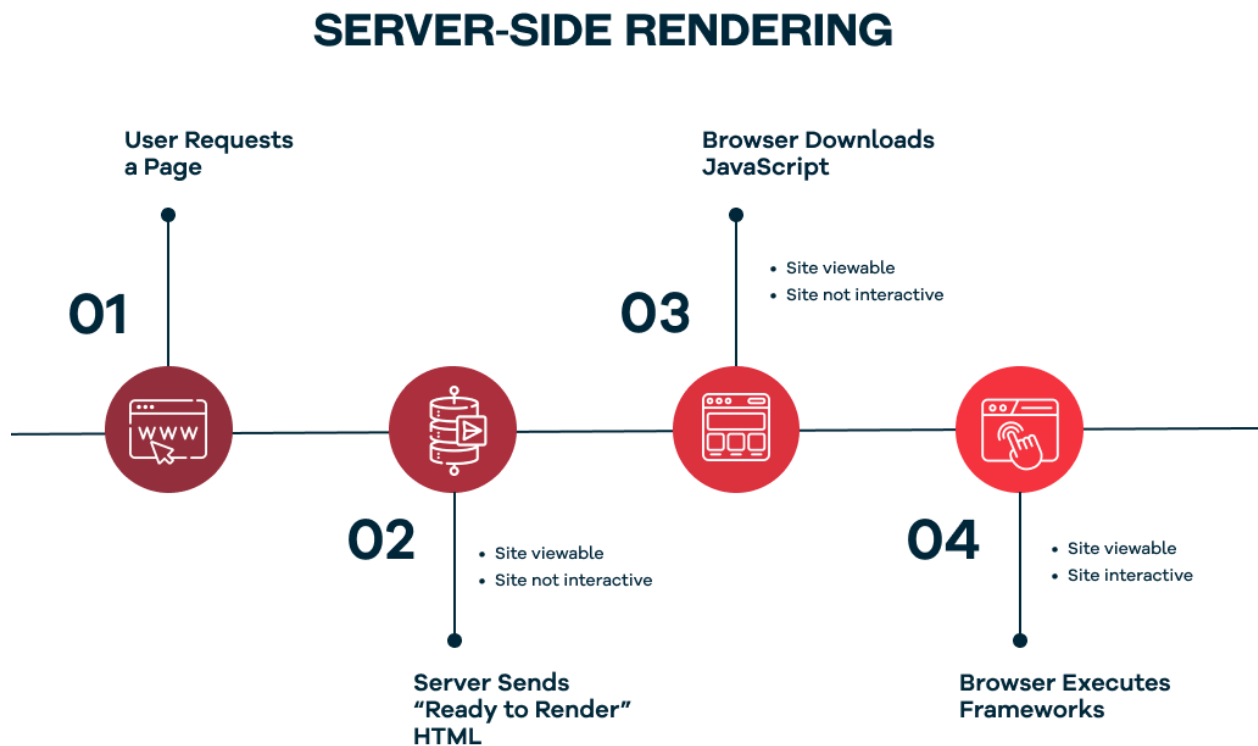


Figura 3: architettura e processi del server-side-rendering

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Pagina caricata tramite SSR</title>
</head>
<body>
<h1>Titolo del paragrafo</h1>
<!-- Intero corpo della pagina -->
</body>
</html>
```

Codice 2: esempio di codice sorgente completo consegnato all'utente dal server

Utilizzando invece il Server-Side-Rendering il codice sorgente della pagina apparirà come una normale pagina HTML, offrendo notevoli vantaggi per l'ottimizzazione nei motori di ricerca (SEO) – si veda il codice riportato nel riquadro Codice 2.

Ci sono comunque pro e contro per entrambi i metodi di rendering. Il Client-Side-Rendering può offrire agli utenti pagine più dinamiche senza bisogno di doverle ricaricare, a discapito di una maggiore dipendenza dal Javascript del client, (che potrebbe essere disattivato, portando a visualizzazioni incomplete della pagina), e una SEO meno efficace, in quanto il codice sorgente della pagina non riporterebbe le informazioni essenziali. D'altra parte il Server-Side-Rendering permette delle prestazioni iniziali migliori, caricando più rapidamente il contenuto già generato del sito, e una SEO migliore e più funzionale all'indicizzazione, peccando però in quanto a interattività della pagina, poiché ogni interazione può richiedere un nuovo caricamento della pagina.

In questa tesi si è scelto di utilizzare Next.js dal momento che questo framework permette di combinare entrambi gli approcci in maniera ottimale così da ottenere le qualità di entrambi, come ad esempio la velocità di caricamento iniziale del Server-Side-Rendering e la fluidità di navigazione del Client-Side-Rendering.

3.2.2 Tailwind CSS

Tailwind CSS⁴ è un framework CSS che, grazie al suo approccio *utility first*, consente di creare interfacce utente in modo rapido direttamente all'interno del file HTML. A differenza di altri framework CSS come Bootstrap⁵, che forniscono componenti precedentemente realizzati da altri sviluppatori, Tailwind offre un insieme di classi che possono essere combinate tra loro permettendo di arricchire e migliorare vari aspetti del design, come colori, grandezze, padding e tipografia. Questa metodologia consente agli sviluppatori di poter costruire design unici senza dover scrivere manualmente regole CSS in fogli di stile separati. La differenza tra i due framework sta proprio nella libertà che concedono, Bootstrap permette l'utilizzo di design finiti che devono essere semplicemente posizionati all'interno del codice HTML tramite delle classi specifiche, Tailwind invece, mette a disposizione dei programmatori solo delle classi brevi e immediate che vanno semplicemente a sostituire le regole CSS; tutte le componenti

⁴ <https://tailwindcss.com/>

⁵ <https://getbootstrap.com/>

della pagina quindi, devono essere create da zero, ma la rapidità di utilizzo e l'immediatezza dell'implementazione del framework facilitano di gran lunga il flusso di lavoro senza dover rinunciare alla personalizzazione.

Tailwind inoltre, si integra perfettamente con Next.js, contribuendo in maniera positiva alla produttività durante lo sviluppo. Tailwind infatti, permette di applicare le singole classi del framework a qualsiasi componente dell'applicazione direttamente all'interno del file *JS* o *TSX*. Questo approccio consente di modificare rapidamente il layout e il design della pagina senza la necessità di dover importare singoli fogli di stile CSS. Tailwind permette inoltre, di creare design responsivi in maniera facile e intuitiva data dalla logica *mobile first* adottata dal framework, e dai breakpoint prestabiliti sempre sotto forma di classi che possono essere applicati a ogni componente, rendendolo a sua volta responsive.

```
//all'interno del file html
<h1 class="intestazione">Testo scritto in vanilla CSS</h1>

//all'interno del file CSS
.intestazione{
  font-size: 2em;
  text-align: center;
}
```

Codice 3: esempio di codice per visualizzare un'intestazione utilizzando vanilla CSS

```
<h1 class="text-3xl text-center">Testo scritto con Tailwind CSS</h1>
```

Codice 4: esempio di codice per visualizzare un'intestazione utilizzando Tailwind CSS

In entrambi gli esempi riportati - Codice 3 e Codice 4 - si vuole visualizzare un'intestazione più grande del testo dei paragrafi al centro della pagina; è evidente come sia molto più immediato scrivere lo stesso codice utilizzando il framework di Tailwind CSS, scrivendo una sola riga di codice in un unico file, rispetto a due file distinti e cinque linee di codice.

Considerando quindi la facilità di utilizzo, di implementazione e di manutenibilità del codice, ci si accorge della perfetta integrazione che questi due framework possiedono, assieme ad una documentazione dettagliata e completa per il loro utilizzo simultaneo.

3.2.3 D3.js

D3.js è una libreria Javascript open source, è un potente strumento per creare visualizzazioni di dati personalizzate e d’impatto, manipolando elementi SVG e CSS all’interno dei file HTML della pagina. D3 è un acronimo che sta per *Data Driven Documents*. D3.js permette infatti di controllare con precisione quale rappresentazione grafica dovrebbe avere un insieme di dati. D3 è in grado di gestire in maniera efficiente grandi set di dati che possono essere analizzati e visualizzati in forma di mappe, tabelle e grafici; incorporandoli facilmente nelle applicazioni web senza dover rinunciare alle prestazioni.

Uno dei vantaggi principali di D3.js è la capacità di data binding, ovvero la capacità di associare i dati da visualizzare agli elementi del DOM.

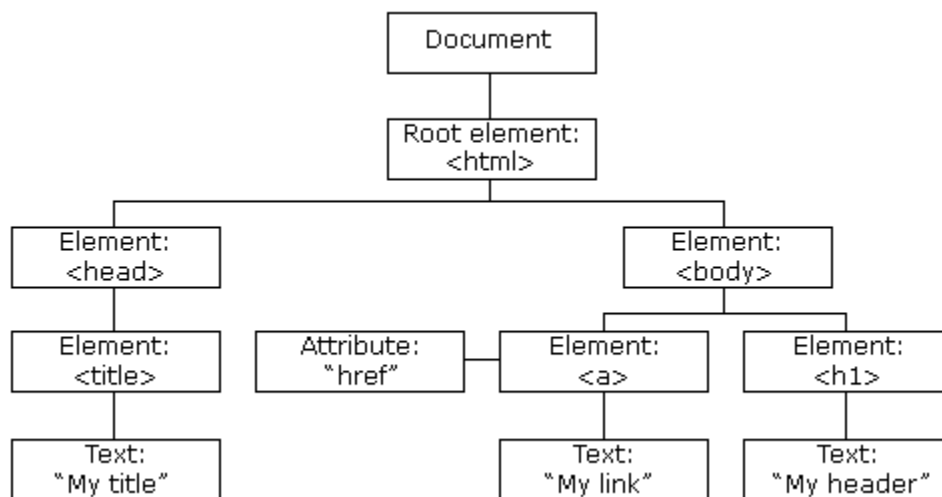


Figura 4: rappresentazione visiva del DOM

DOM sta per *Document Object Model*, ed è un’interfaccia che rappresenta e interagisce con i documenti HTML sul web. Ogni pagina HTML è rappresentata come un albero di nodi, in cui ogni nodo rappresenta una parte del documento. Nella figura viene rappresentata una semplice pagina in cui è possibile notare che il tag `<html></html>` sia l’elemento *genitore* di tutti gli altri, e come poi si divida in due nodi distinti, in cui `<head></head>` è responsabile dei metadati della pagina, come il titolo o la descrizione per i motori di ricerca; e invece `<body></body>` corrisponde a tutto ciò che vede l’utente, come ad esempio titoli, paragrafi e immagini.

Il DOM quindi, consente al codice di esecuzione nel browser di accedere e interagire con i nodi della pagina.

Gli sviluppatori quindi, grazie a D3.js, possono scegliere elementi specifici nel file HTML e modificarli in base ai dati forniti in maniera flessibile.

La libreria D3.js è stata di vitale importanza in questo progetto per capire come visualizzare i dati in maniera ottimale, ha permesso di trasformare un file JSON di dati grezzi in un grafico comprensibile e di facile implementazione all'interno della pagina, grazie anche ai componenti Next.js.

3.2.4 Nivo

D'altra parte, nonostante sia stato utile e istruttivo costruire un grafico da zero utilizzando la libreria di D3.js, è corretto utilizzare le risorse già presenti sul web, se esse permettono di ottenere dei risultati qualitativamente adeguati ed allo stesso tempo aiutano a semplificare lo sviluppo. Nivo⁶ svolge egregiamente questo compito e si è rilevata una soluzione adeguata per lo sviluppo dei grafici presenti nella homepage del progetto TIPS.

Nivo è una libreria che permette di visualizzare dati, progettata specificamente per l'ecosistema React e basata su D3.js; Nivo offre un'ampia gamma di componenti altamente personalizzabili per creare grafici interattivi e visivamente accattivanti.

Una delle principali ragioni per cui Nivo è utile è, oltre alla perfetta integrazione di librerie già presenti nel progetto, la sua facilità d'uso. La libreria fornisce una documentazione ben strutturata e ricca di esempi, che rende l'apprendimento e l'implementazione dei grafici un processo relativamente semplice; fornendo numerosi grafici che possono essere adottati dagli sviluppatori in base alle tipologie di dati da analizzare e visualizzare. Inoltre, Nivo permette di implementare grafici reattivi che si adattano automaticamente alle dimensioni del contenitore, migliorando l'esperienza utente aiutando a costruire un design responsive. Per dimostrare la profonda differenza nell'impiego di Nivo e D3.js verrà fatto un esempio in cui si costruisce un semplice grafico a barre costituito da soli 3 elementi.

```
import { ResponsiveBar } from '@nivo/bar';  
const data = [
```

⁶ <https://nivo.rocks/about/>

```

    { label: "Barra 1", value: 12 },
    { label: "Barra 2", value: 23 },
    { label: "Barra 3", value: 34 },
  ];

  const MyBarChart = () => (
    <ResponsiveBar
      data={data}
      keys={['value']}
      indexBy="label"
      margin={{ top: 50, right: 130, bottom: 50, left: 60 }}
      padding={0.3}
      // Altre configurazioni opzionali
    />
  );

  export default MyBarChart;

```

Codice 5: esempio di codice di un grafico a barre utilizzando la libreria Nivo

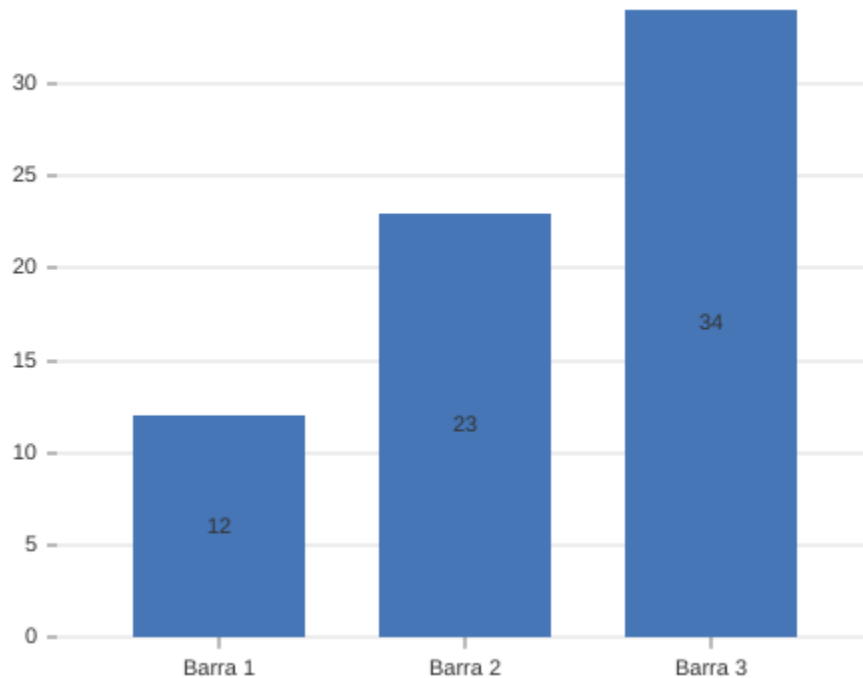


Figura 5: Grafico di esempio costruito utilizzando Nivo

Il codice, riportato nel riquadro Codice 5 ed impiegato per mostrare un grafico a barre utilizzando la libreria Nivo, è semplice, intuitivo e facilmente adattabile ai dati. Tuttavia, nonostante offra un'ampia possibilità di configurazione, non raggiunge il livello di controllo totale che si ottiene costruendo un grafico da zero con D3.js.

```
<!DOCTYPE html>
<html lang="it">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Bar Chart con D3.js</title>
  <style>
    .bar { fill: steelblue; }
  </style>
</head>
<body>
  <svg width="500" height="300"></svg>
  <script src="https://d3js.org/d3.v7.min.js"></script>
  <script>
    const data = [
      { label: 'Barra 1', value: 12 },
      { label: 'Barra 2', value: 23 },
      { label: 'Barra 3', value: 34 },
    ];

    const svg = d3.select("svg"),
      margin = { top: 20, right: 30, bottom: 40, left: 40 },
      width = +svg.attr("width") - margin.left -
margin.right,
      height = +svg.attr("height") - margin.top -
margin.bottom,
      g = svg.append("g").attr("transform",
`translate(${margin.left},${margin.top})`);

    const x = d3.scaleBand().rangeRound([0,
width]).padding(0.1).domain(data.map(d => d.label));
```

```

    const y = d3.scaleLinear().rangeRound([height, 0]).domain([0,
d3.max(data, d => d.value)]);

    g.append("g")
      .attr("class", "axis axis--x")
      .attr("transform", `translate(0,${height})`)
      .call(d3.axisBottom(x));

    g.append("g")
      .attr("class", "axis axis--y")
      .call(d3.axisLeft(y).ticks(10));

    g.selectAll(".bar")
      .data(data)
      .enter().append("rect")
      .attr("class", "bar")
      .attr("x", d => x(d.label))
      .attr("y", d => y(d.value))
      .attr("width", x.bandwidth())
      .attr("height", d => height - y(d.value));
</script>

</body>
</html>

```

Codice 6: esempio di codice di un grafico a barre utilizzando la libreria D3.js

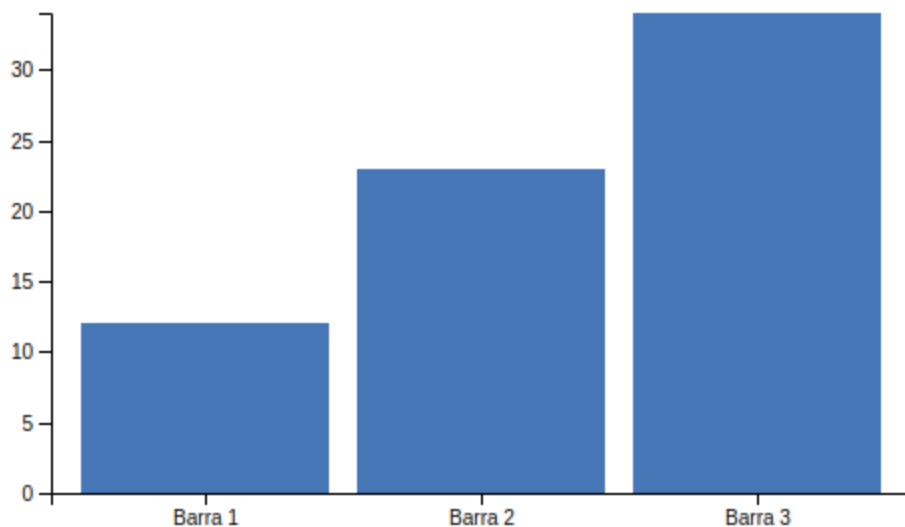


Figura 6: Grafico di esempio costruito utilizzando D3.js

Vi sono delle evidenti differenze fra il Codice 5 e Codice 6, ed è chiaro anche considerando l'output dei grafici, che in questi casi è preferibile utilizzare Nivo per un risultato ottimale. Se si analizza con attenzione il codice in cui viene usato D3 ci si accorge della mole di comandi da scrivere per ottenere lo stesso risultato. In particolare con D3 bisogna

- creare manualmente l'elemento `<svg>` e impostarne altezza e larghezza
- gestire le dimensioni effettive del grafico sottraendo i margini dallo spazio disponibile, (`width` e `height`)
- creare un gruppo `<g>` per contenere gli elementi grafici e posizionarlo correttamente attraverso una trasformazione:
`g=svg.append("g").attr("transform", `translate(${margin.left},${margin.top})`);`
- definire manualmente le scale sia per l'asse orizzontale che per quello verticale, specificando i domini, i range ed eventuali spazi con i padding, (`.scaleBand()`, `.rangeBound()`, `.padding()`, `.domain()`)
- aggiungere gli assi manualmente
- creare gli elementi grafici per le linee e le etichette degli assi, (`d3.axisBottom(x)` e `d3.axisLeft(y)`),
- infine bisogna manualmente creare le barre utilizzando gli elementi svg `<rect>` e associarle ai dati con `.data()` impostando poi altezza e larghezza in base ai valori delle scale, (`.attr("width", x.bandwidth())` e `.attr("height", d => height - y(d.value))`).

Nivo ha gestito tutti questi passaggi autonomamente, permettendo di scrivere un codice semplice, veloce e facilmente leggibile; e benché D3.js sia uno strumento molto più preciso e personalizzabile, Nivo permette in molti casi di avere lo stesso risultato, se non migliore, ma con una drastica riduzione del tempo impiegato per scrivere il codice, e una maggiore facilità nel mantenerlo in futuro.

3.3 Realizzazione della Homepage

Per realizzare l'homepage ho ricevuto delle indicazioni specifiche ed ho seguito dei mockup realizzati da Agnese Pozzobon, una Graphic Designer, e concordati e pianificati man mano con i membri del progetto TIPS.

La pagina realizzata è visivamente molto diversa da quella precedente, nonostante a livello contenutistico non ci sia molta differenza; il sito comprende una sola landing page composta da più sezioni che l'utente visualizza a schermo intero mentre scorre verso il fondo.

Non sono stati divulgati i motivi precisi su cui si basano le scelte stilistiche della nuova homepage, tuttavia è evidente un cambio della palette dei colori, dei font utilizzati, e della gerarchia di alcuni elementi nelle varie sezioni; scegliendo così, un design minimale composto da uno sfondo grigio-chiaro presente in tutta la pagina. Precedentemente, ogni sezione aveva un colore di sfondo diverso, solitamente nelle tonalità del blu e del verde, nonostante ci fosse comunque una sezione con un background di colore grigio. Tuttavia, uno sfondo monocromatico conferisce al sito un aspetto più professionale, uniforme e coeso; rendendo più semplice e intuitiva la navigazione e riducendo il rischio di creare incoerenze visive durante la manutenzione.

I contenuti del sito invece, sono primariamente di colore nero, in modo da farli risaltare sullo sfondo, e per alcuni testi particolarmente importanti, come ad esempio il titolo "TIPS PROJECT" presente nell'header, si è scelto un colore blu acceso. Le singole sezioni con i relativi componenti verranno illustrati singolarmente.

3.3.1 Header

Partendo dall'alto, nella homepage è presente un *header*, (ovvero una barra orizzontale posta sopra tutto il documento), in cui in alto a sinistra si trova il titolo già menzionato, e la dicitura "TECHNOSCIENTIFIC ISSUES IN THE PUBLIC SPHERE" accanto. Dall'altro lato dell'header sono stati aggiunti due bottoni che permettono rispettivamente di cambiare lingua della pagina e di loggarsi all'interno del portale.



Figura 7: nuova versione dell' header della homepage

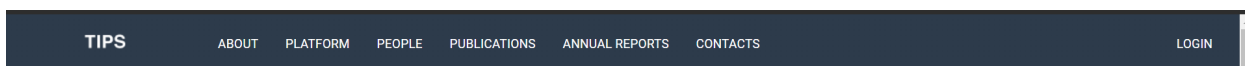


Figura 8: vecchia versione dell'header della homepage

La nuova versione dell'header è diversa dalla precedente non solo a livello estetico, ma anche per quanto riguarda le funzionalità: nella vecchia versione non era possibile cambiare lingua e non vi era la dicitura integrale dell'acronimo TIPS, ma soprattutto,

l'header fungeva anche da barra di navigazione tra le varie sezioni. Nella nuova versione è stato scelto di creare una barra di navigazione laterale visibile mentre si naviga la pagina.

3.3.2 Navigation sidebar

Posizionare la barra di navigazione lateralmente può portare dei vantaggi nell'esperienza utente: può rendere l'organizzazione della pagina più intuitiva e fluida, aiutando gli utenti a organizzarsi meglio, permette di utilizzare più spazio nella sistemazione delle sezioni o di eventuali sottocartelle, e si adatta generalmente meglio ai dispositivi mobili e ai layout responsivi ottimizzando l'esperienza.



Figura 9: Navigation sidebar, visualizzazione desktop

Nella sidebar sono presenti tutte le sezioni del sito, permettendo di passare da una sezione all'altra in maniera fluida grazie alla classe *scroll-smooth* attribuita all'intero documento HTML, in modo da animare l'intera pagina ogni qualvolta si viene rimandati a una specifica sezione.

Anche l'approccio ai dispositivi mobili è molto diverso fra le due versioni del sito.

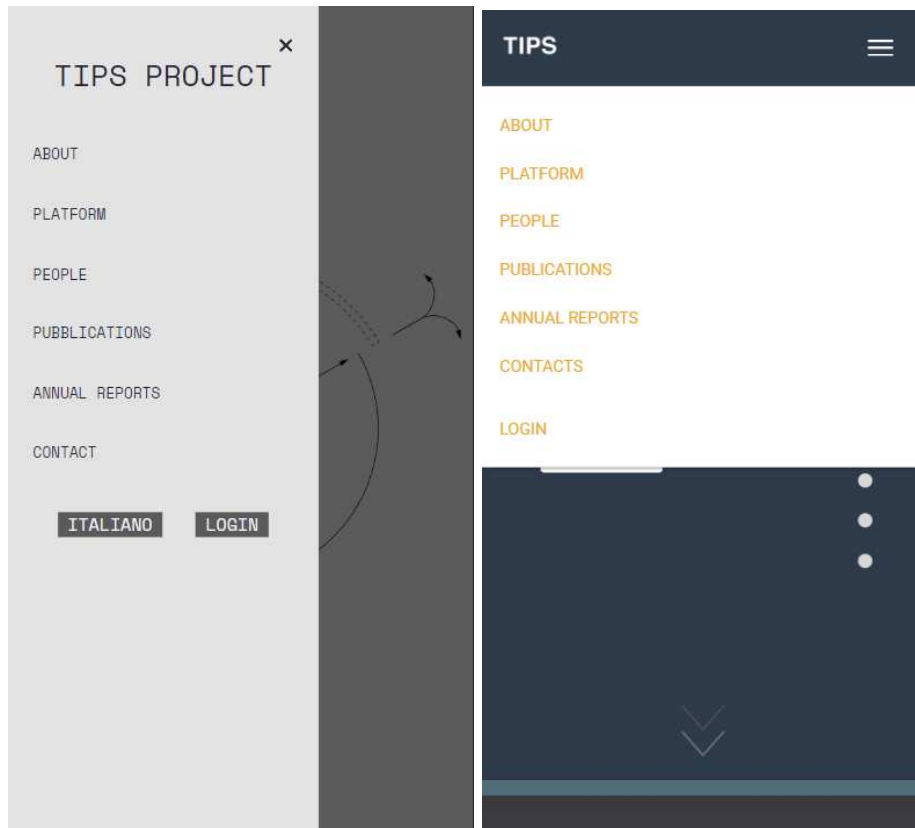


Figura 10: confronto della visualizzazione mobile della barra di navigazione nelle due versioni

In entrambe le versioni si accede al menù di navigazione tramite un'icona di un menù ad hamburger; chiaramente l'animazione avrà direzioni diverse: nel caso della barra laterale partirà da sinistra verso destra e nel caso della barra orizzontale partirà dall'alto verso il basso. Nella nuova versione oltre al titolo e alle sezioni cliccabili, sono presenti i bottoni per cambiare la lingua e per effettuare il login. Si nota come vi sia una migliore gestione dello spazio disponibile che permetterebbe anche di aggiungere contenuti in futuro. Per evitare di confondere l'utente è stato oscurato il corpo del sito visualizzato "sotto" la sidebar ed è stata implementata una funzione che chiude automaticamente il menù ogni volta che viene caricata una sezione, cosa che non avveniva nella vecchia versione, lasciando la barra di navigazione aperta con sotto la sezione a cui si voleva accedere.

3.3.3 Scroll indicator

Una feature presente nell'homepage è la presenza di un indicatore che mostra la percentuale della pagina visualizzata dall'utente, comunemente chiamato *scroll*

indicator. Lo scroll indicator era già presente nella versione precedente ma era visivamente e logicamente diverso.



Figura 11: confronto dello scroll indicator nelle due versioni

L'indicatore scelto per la nuova versione è minimale, in linea con il resto del sito, è composto da una sola linea sottile che si riempie man mano che l'utente scorre verso il basso, all'inizio della pagina è completamente grigio e alla fine è completamente nero. Nella vecchia versione invece, sono presenti dei cerchi che simboleggiano le sezioni del sito, che vengono colorati appena la sezione interessata è attiva.

Lo stile adottato è chiaramente diverso, e il riempirsi in maniera graduale di una barra laterale dona maggiore coesione al design complessivo della pagina. Tuttavia la logica del codice utilizzato per i due indicatori è molto diversa. Nella vecchia versione sono presenti dei cerchi all'interno di un tag `<nav></nav>` che vengono colorati di giallo in base alla sezione presente sullo schermo, automaticamente fornite da i tag `<section></section>` all'interno del codice HTML; ad ogni cerchio viene assegnato l'indirizzo di un `<id></id>`, ovvero un tag HTML, che corrisponde a un id relativo a una sezione specifica.

```

<nav id="cd-vertical-nav">
  <ul>
    <li>
      <a href="#intro" data-number="1" class="">
        <span class="cd-dot"></span>
      </a>
    </li>
    <li>
      <a href="#about" data-number="2" class="is-selected">
        <span class="cd-dot"></span>
      </a>
    </li>
    <li>
      <a href="#platform" data-number="3" class="">
        <span class="cd-dot"></span>
      </a>
    </li>
    <!-- resto delle sezioni... -->
  </ul>
</nav>

```

Codice 7: rappresentazione della logica dello scroll indicator nella vecchia versione

Il codice mostra il tag `<nav></nav>` al cui interno è racchiusa una lista non ordinata, (``), di elementi, (``), in cui sono presenti i cerchi; il tag *anchor*, ovvero `<a>`, presenta l'attributo *href* che rimanda esattamente agli id delle sezioni, che possiedono lo stesso nome senza “#” iniziale, facendo così corrispondere ogni cerchio a una sezione.

La logica implementata nel codice della nuova versione risulta profondamente differente:

```
"use client";

import { useEffect, useState } from "react";

export default function ScrollIndicator() {
  const [scrollProgress, setScrollProgress] = useState(0);

  useEffect(() => {
    const handleScroll = () => {
      const scrollContainer = document.documentElement;
      const winScroll = window.scrollY;
      const height =
        scrollContainer.scrollHeight - scrollContainer.clientHeight;
      const scrolled = (winScroll / height) * 100;
      setScrollProgress(scrolled);
    };

    window.addEventListener("scroll", handleScroll);
    handleScroll(); // Chiamata iniziale

    return () => window.removeEventListener("scroll", handleScroll);
  }, []);

  return (
    <div className="hidden md:block fixed top-20 right-0 h-2/5 w-1
    bg-gray-300/20 z-50 mx-5 my-44">
      <div
        className="bg-gray-800"
        style={{ height: `${scrollProgress}%` }}
      ></div>
    </div>
  );
}
```

Codice 8: componente scrollindicator.js

Il codice 8 è responsabile del componente `<ScrollIndicator />`, che costituisce un indicatore di scorrimento verticale sul lato destro della pagina, mostrando la percentuale di scorrimento dell'utente rispetto all'altezza totale della pagina.

Il componente inizia con la linea di codice `"use client"`, che serve per specificare che il componente deve essere renderizzato dal client (CSR). Il motivo di tale comando è che l'indicatore di scroll deve potersi aggiornare ogni volta che l'utente scorre la pagina; se venisse utilizzato il Server-Side-Rendering il componente non potrebbe cambiare stato nel tempo perché il server lo invierebbe in una versione finita e immutabile e sarebbe, dunque, statico.

Nel codice vengono utilizzati due principali *hook* di React.js: *useState* e *useEffect*. Gli Hook sono funzioni introdotte nella versione 16.8 di React, e permettono agli sviluppatori di utilizzare funzionalità come lo *stato* e gli *effetti collaterali* all'interno dei componenti funzionali senza doverle scrivere autonomamente. In React lo stato è un oggetto che rappresenta i dati di un componente e determina come questo viene visualizzato. Ogni volta che lo stato di un componente cambia, React riesegue il rendering del componente per riflettere le nuove informazioni. Gli effetti collaterali sono invece delle operazioni che interagiscono con gli elementi del "mondo esterno" del componente, come ad esempio le richieste di rete o le manipolazioni del DOM.

Specificatamente, in questo caso vediamo: *useState*, che consente di gestire lo stato all'interno di un componente funzionale restituendo un array con due elementi: il valore attuale dello stato e una funzione per aggiornarlo; e *useEffect*, che permette di gestire effetti collaterali, come il recupero di dati o la manipolazione del DOM.

All'interno del componente viene utilizzato `useState` per inizializzare lo stato `scrollProgress` che terrà traccia della percentuale di scorrimento della pagina e verrà aggiornato dalla funzione `setScrollProgress`.

L'hook *useEffect* definisce una funzione chiamata `handleScroll`. Questa funzione è responsabile del calcolo della percentuale di scorrimento della pagina e dell'aggiornamento dello stato `scrollProgress`. La funzione ottiene l'elemento radice del documento, (`scrollContainer`), che equivale al tag `<html></html>`, e `winScroll`, ovvero la quantità di scorrimento verticale della finestra. Successivamente `height` calcola l'altezza totale della pagina meno l'altezza visibile della finestra e `scrolled` utilizza questi valori per determinare la percentuale di

scorrimento, che verranno utilizzati per aggiornare lo stato tramite `setScrollProgress`.

`useEffect`, quindi, aggiunge un listener all'evento di scorrimento della finestra, chiamando la funzione `handleScroll` ogni volta che l'utente scorre la pagina. Inoltre, la stessa funzione viene chiamata immediatamente quando il componente viene montato per impostare l'indicatore di scorrimento nella posizione corretta nel momento in cui la pagina viene caricata per la prima volta. Quando poi il componente viene smontato, il listener dell'evento di scorrimento viene rimosso per evitare perdite di memoria.

Infine il componente restituisce un elemento `<div></div>` che rappresenta l'indicatore di scorrimento che ha come altezza i valori di `scrollProgress`, aggiornandosi nel tempo.

Si evince, quindi, che gli indicatori di scroll nelle due versioni hanno una logica e un funzionamento molto diverso fra loro, la nuova versione è sicuramente più complessa e articolata ma permette di ottenere un risultato moderno e personalizzato.

3.3.4 Home section

Quando viene visualizzato il sito, la prima sezione che l'utente vede è una sorta di home all'interno della home page, che è stata lievemente rivisitata rispetto alla vecchia versione.

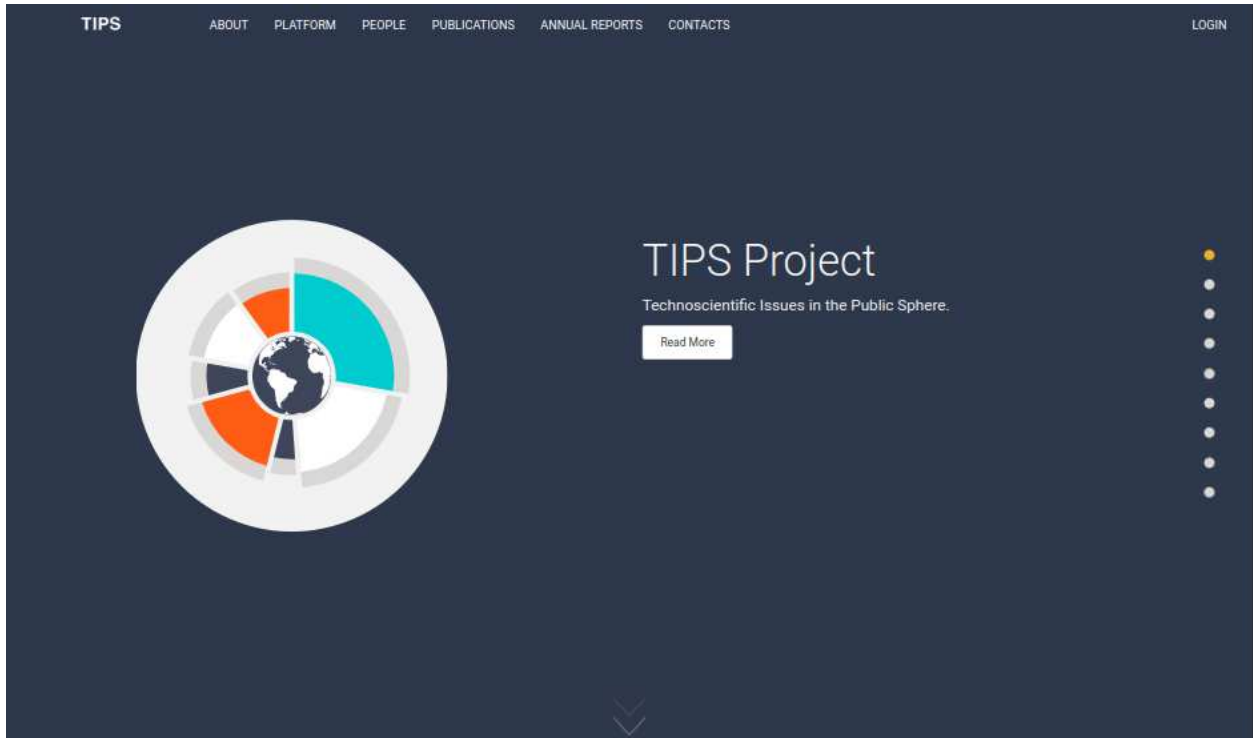


Figura 12: home section nella vecchia versione

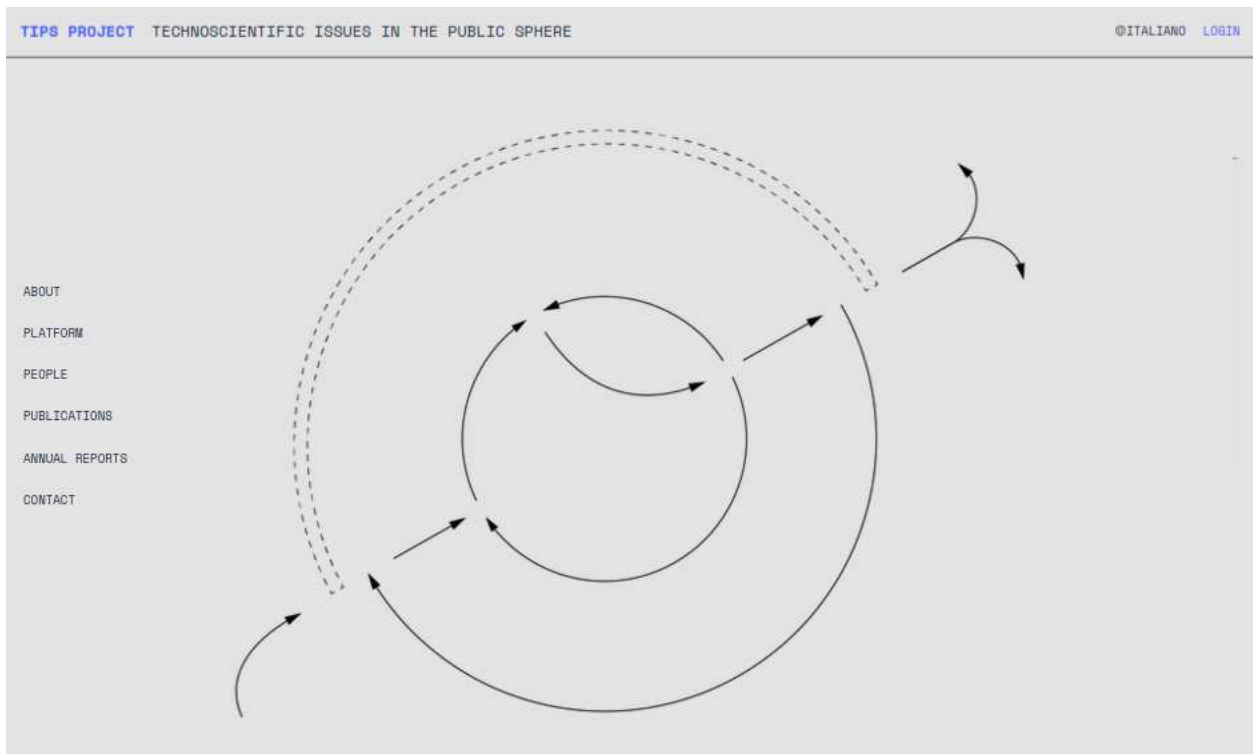


Figura 13: home section nella nuova versione

La nuova versione presenta un'unica immagine che ricopre praticamente tutto lo schermo e senza nessun tipo di testo, scegliendo così uno stile più minimale e immediato. Ciò si riflette soprattutto nella visualizzazione mobile.

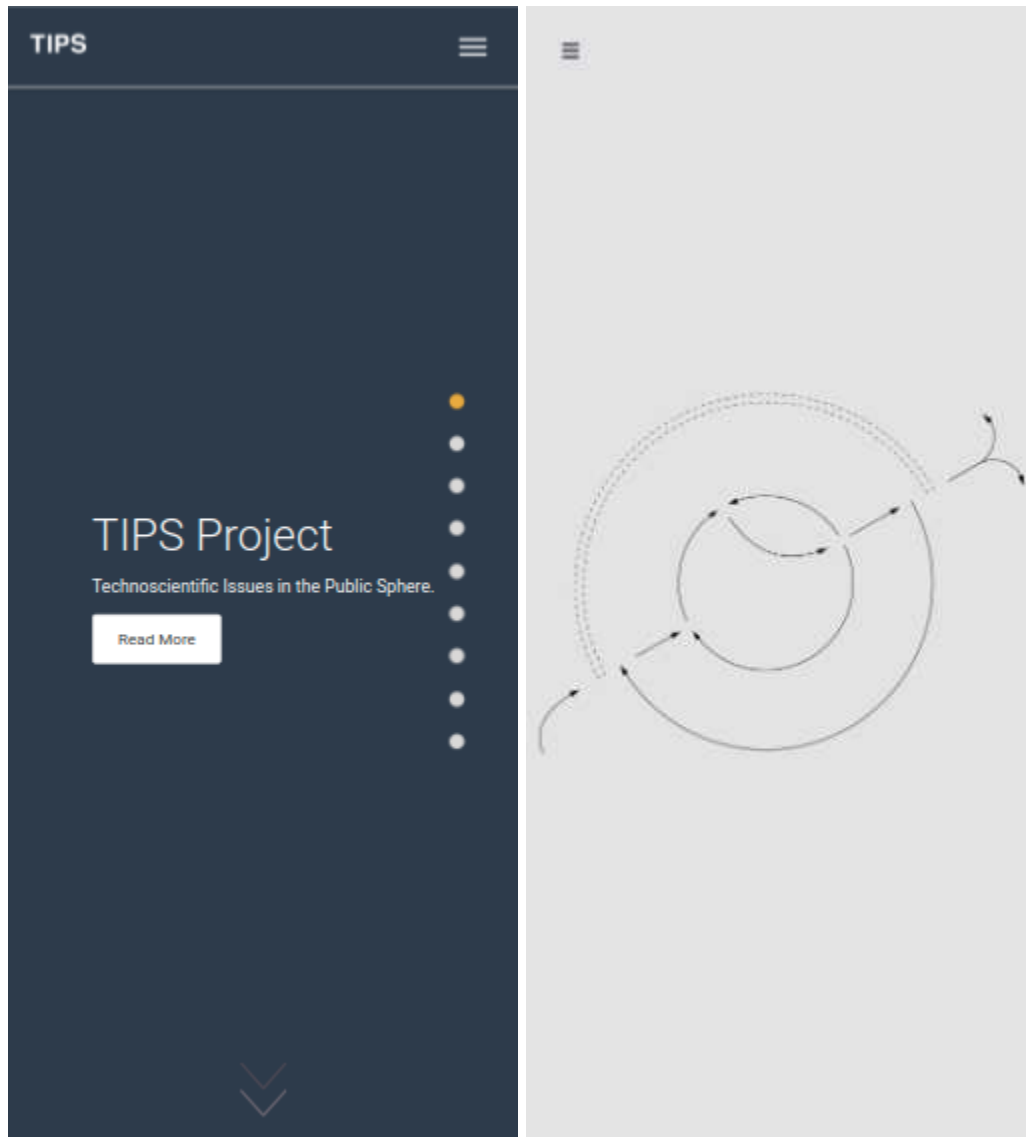


Figura 14: confronto tra le due versioni di home su dispositivo mobile Samsung Galaxy S20 Ultra

Molti meno elementi sono presenti sullo schermo, con un risultato meno distraente e che facilita la navigazione.

3.3.5 About section

In questa sezione viene spiegato brevemente in cosa consiste il progetto TIPS, menzionando l'importanza di monitorare le evoluzioni di temi legati alla scienza e alla tecnologia tramite l'analisi dei contenuti digitali presenti sul web.

Le versioni sono molto simili sia come struttura che a livello contenutistico, viene cambiata l'immagine e rimosso il bottone "request a demo" che rimanderebbe alla sezione contatti.

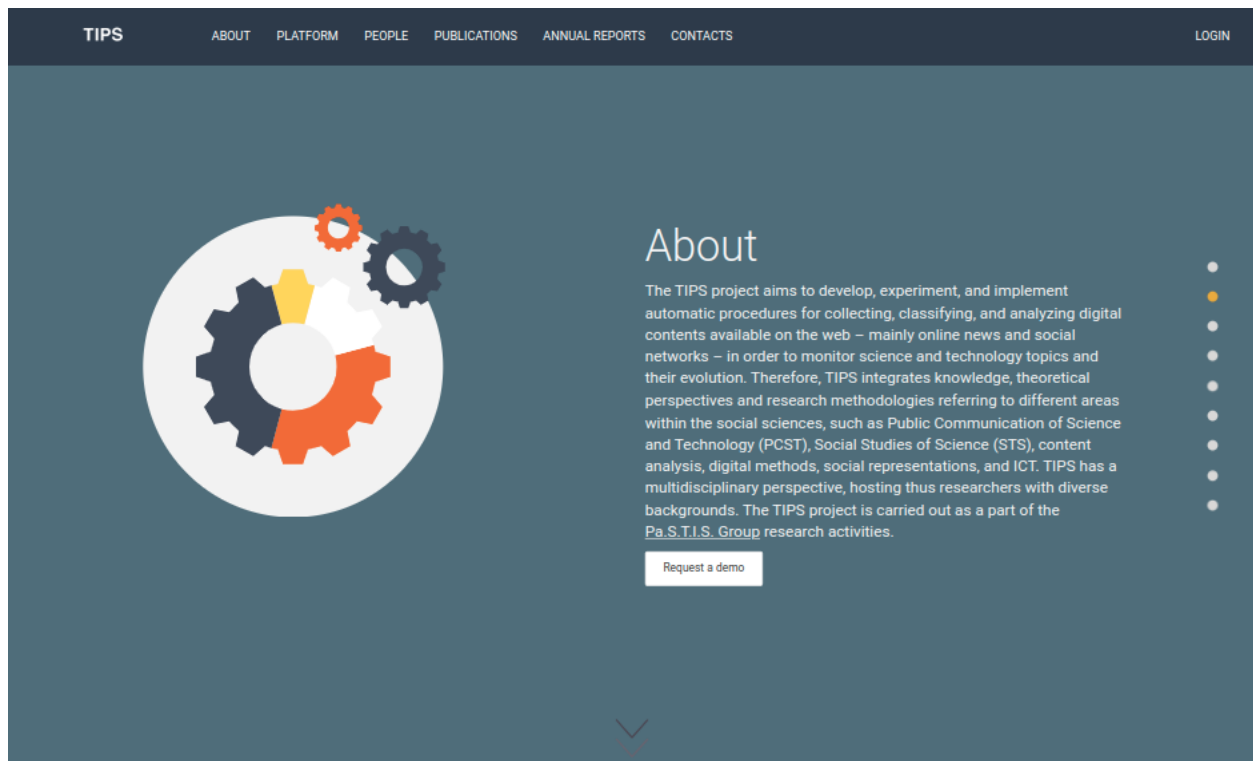


Figura 15: about section nella vecchia versione

About

The TIPS project aims to develop, experiment, and implement automatic procedures for collecting, classifying, and analyzing digital contents available on the web - mainly online news and social networks - in order to monitor science and technology topics and their evolution. Therefore, TIPS integrates knowledge, theoretical perspectives and research methodologies referring to different areas within the social sciences, such as Public Communication of Science and Technology (PCST), Social Studies of Science (STS), content analysis, digital methods, social representations, and ICT. TIPS has a multidisciplinary perspective, hosting thus researchers with diverse backgrounds. The TIPS project is carried out as a part of the Pa.S.T.I.S. Group research activities.

Figura 16: about section nella nuova versione

3.3.6 Platform

La sezione platform è particolarmente importante per il lavoro svolto all'interno di questa tesi, perché coniuga in maniera ottimale l'utilizzo delle varie tecnologie adottate per la realizzazione del sito.

In entrambe le versioni, la struttura è concepita come un carosello scorrevole, all'interno del quale i contenuti visualizzati cambiano in base alla scheda selezionata nel carosello. La vecchia versione riporta una breve descrizione della dashboard descrivendone alcune caratteristiche principali come la possibilità di ricerca di segmenti di testo e la possibilità di generare grafici in maniera automatica utili per le attività di ricerca. Ciò occupa tutte e tre le schede del carosello; nella nuova versione queste informazioni sono state riportate in un'unica scheda.

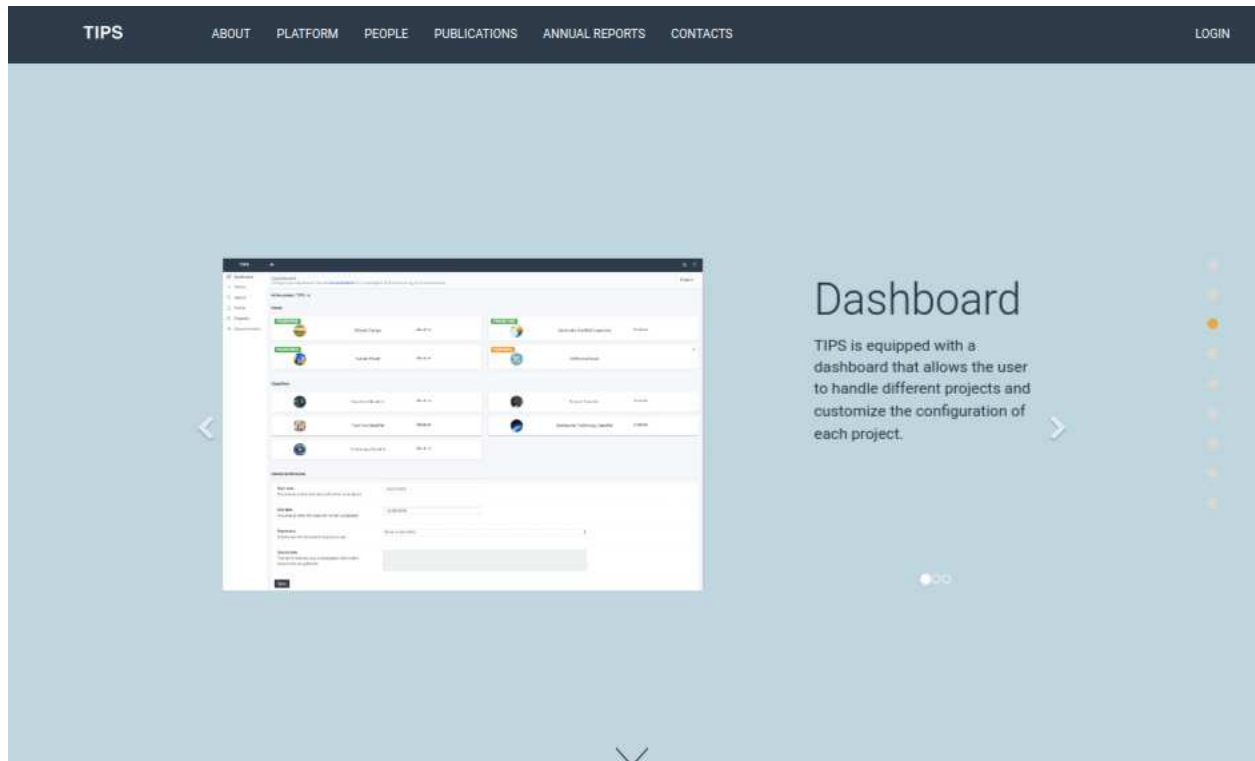


Figura 17: platform section nella vecchia versione

Nella vecchia versione erano presenti delle immagini affiancate da delle schermate esplicative della pagina corrispondente; queste sono state rimosse nella versione aggiornata in modo da scrivere un testo più conciso e veloce da leggere.

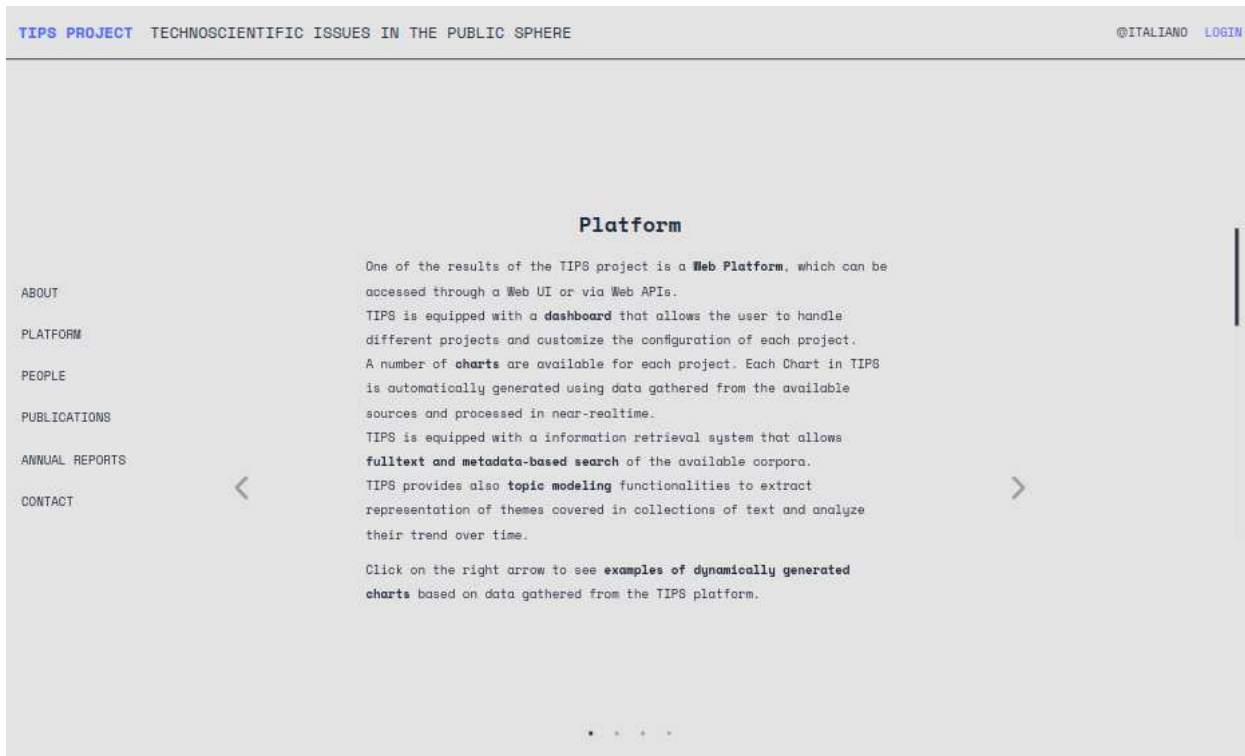


Figura 18: scheda testuale della sezione platform della nuova versione

Nelle successive schede del carosello sono presenti dei grafici di esempio particolarmente significativi. Ma prima è opportuno fare alcune precisazioni: la piattaforma del progetto TIPS raccoglie articoli provenienti da diverse testate giornalistiche presenti sul web; all'interno di ogni articolo sono presenti delle informazioni rilevanti e dei metadati, come la categoria di classificazione dell'articolo stesso, che vengono poi analizzati e organizzati tramite vari algoritmi che si occupano di classificarli in base al contenuto e ad altri indicatori, automatizzando così il processo. Questo processo di raccolta dei dati e classificazione è il fulcro della piattaforma, e i grafici servono proprio a permettere all'utente di poter interagire visivamente con questi dati e manipolarli a seconda delle proprie esigenze professionali e/o di ricerca.

3.3.6.1 Line chart

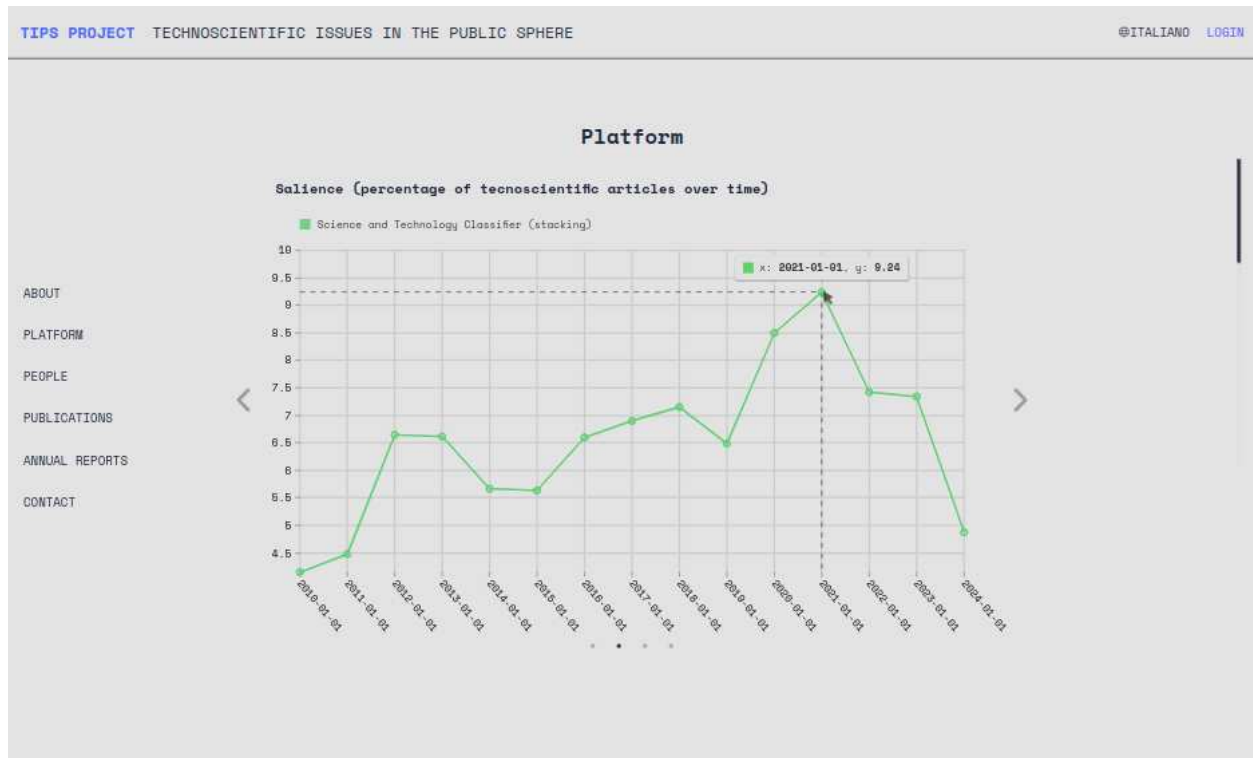


Figura 19: line chart costruito con Nivo

Il grafico è stato realizzato utilizzando la libreria Nivo, e consiste in un line chart che rappresenta la salienza, ossia l'evoluzione della percentuale di articoli rilevanti a scienza e tecnologia nel tempo. Viene analizzato un intervallo temporale che va dal 1° gennaio 2010 al 1° gennaio 2024, utilizzando un anno come unità temporale. Analizzando il grafico emerge un andamento complessivamente altalenante ma tendenzialmente in crescita dal 2010 al 2021, seguito da una significativa diminuzione negli anni più recenti.

Per la realizzazione del grafico è stato creato il componente `<LineChart />`

```
import { ResponsiveLine } from "@nivo/line";
import lineData from "../.././linechart.json";

export default function LineChart() {
  return(
    <ResponsiveLine
      data={lineData}
      margin={{ top: 60, right: 50, bottom: 70, left: 50 }}
    />
  )
}
```

```

xScale={{ type: "point" }}
yScale={{ /* personalizzazioni grafiche */ }}
yFormat=" >-.2f"
axisTop={null}
axisRight={null}
axisBottom={{ /* personalizzazioni grafiche */ }}
axisLeft={{ /* personalizzazioni grafiche */ }}
pointSize={7}
pointColor={{ theme: "background" }}
pointBorderWidth={2}
pointBorderColor={{ from: "serieColor" }}
pointLabel="data.yFormatted"
pointLabelYOffset={0}
enableTouchCrosshair={true}
useMesh={true}
colors=[["#42cf66"]]
legends=[[ /* personalizzazioni grafiche */ ]]
theme={{ /* personalizzazioni grafiche */ }}
/>
);}

```

Codice 9: componente linechart.js

Il codice 9 consiste in una versione semplificata del codice del grafico. Per prima cosa viene importata la libreria `@nivo/line` per la realizzazione del line chart, le cui dimensioni si adatteranno automaticamente alla larghezza e altezza del contenitore, (in questo caso la scheda del carosello); successivamente viene definita la funzione `LineChart()` che rappresenta il corpo del componente. Tutto il codice all'interno del componente riguarda l'aspetto estetico del grafico, in modo da poterlo visualizzare al meglio all'interno della pagina. Nivo snellisce enormemente il flusso di lavoro permettendo allo sviluppatore di potersi concentrare sul problema da analizzare. Oltretutto, la libreria di default presenta una funzione non presente automaticamente utilizzando D3.js; se l'utente posiziona il cursore del mouse in corrispondenza dei punti del grafico, vedrà apparire delle finestre chiamate *tooltips*, che riportano le informazioni specifiche del punto selezionato. Questa feature risulta particolarmente utile quando vi è una quantità considerevole di dati presenti sullo schermo, e può essere attivata semplicemente cambiando il valore *booleano*, (vero o falso), dell'attributo

useMesh={true}. Si ha l'esempio del tooltip osservando il cursore posizionato sul valore dell'anno 2021.

I dati utilizzati in questo grafico provengono da un file JSON presente localmente, come si può vedere dalla linea `import lineData from "../../././linechart.json";`, tuttavia, si può ottenere lo stesso risultato anche se si dovessero richiedere dei dati presenti in un server, ad esempio tramite una chiamata API, quindi non si è limitati in questo senso

3.3.6.2 General framing

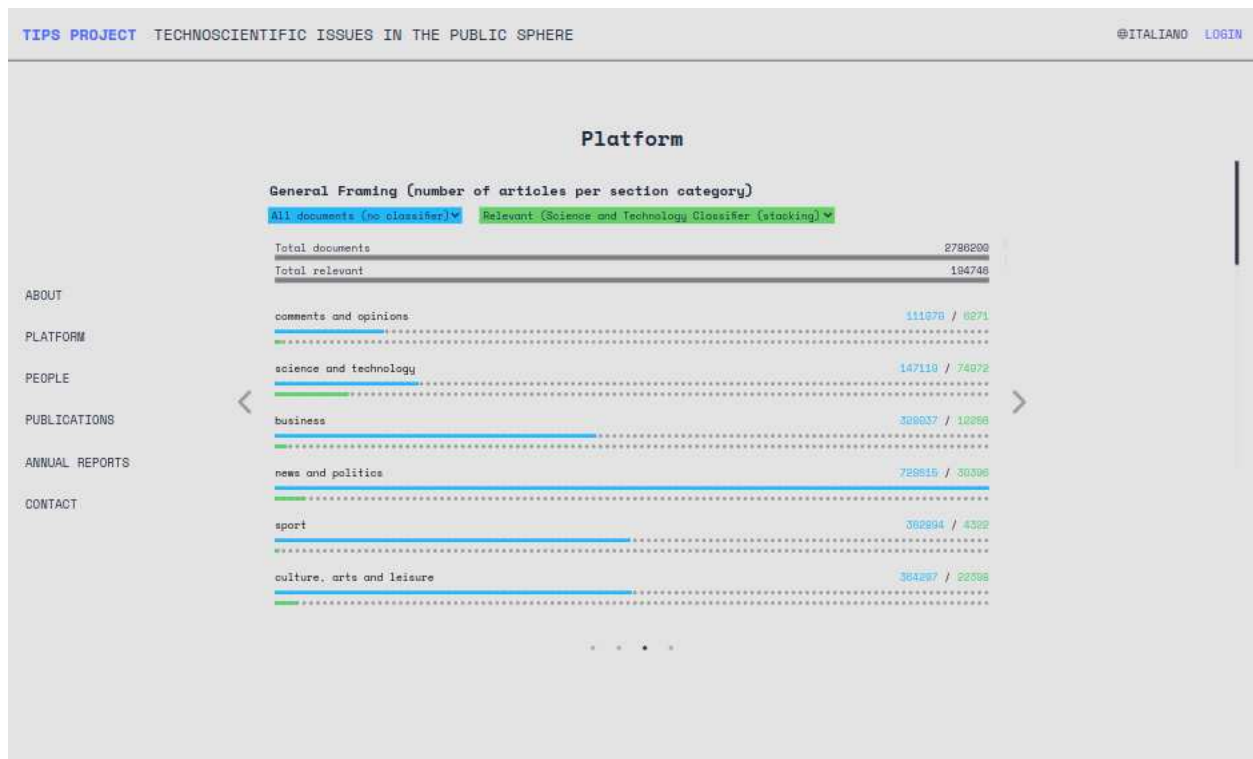


Figura 20: grafico general framing creato con D3.js

Il secondo grafico è stato realizzato utilizzando la libreria D3.js dato che questa tipologia particolare di rappresentazione visiva dei dati non è supportata di default da Nivo. Il grafico indica come prima cosa il numero totale di articoli e il numero totale di articoli rilevanti l'ambito del discorso su scienza e tecnologia. Successivamente mostra la quantità di articoli totali e la quantità di articoli rilevanti divisi per categoria tematica, come sport, business, cultura o politica. Tale suddivisione per categorie tematiche è basata sulle sezioni in cui sono stati pubblicati gli articoli nei quotidiani online. Le sezioni sono state raggruppate manualmente in categorie dai membri del progetto TIPS, mentre l'assegnazione degli articoli nelle varie categorie avviene in modo automatico

sulla base dei metadati; la predizione della rilevanza rispetto al discorso su scienza e tecnologia è effettuata automaticamente mediante un classificatore basato su tecniche di Machine Learning. Il grafico, oltre ad affiancare le barre di diverso colore per dare un'idea all'utente del rapporto fra le due quantità, fornisce anche i valori precisi che possono essere molto utili per svolgere analisi approfondite. La presenza di una barra verde in categorie diverse da quelle relative a scienza e tecnologia, supporta la scelta fatta nel progetto TIPS di non basarsi solo sulla categorizzazione in sezioni degli articoli fatta dai quotidiani.

Anche in questo caso i dati sono stati recuperati da un file JSON locale ma non vi è nessun limite riguardo la possibilità di attingere a dati presenti su server esterni. In questo caso, la logica utilizzata per costruire la dimensione delle barre è stata di calcolare il valore massimo tra i due array contenenti i dati da confrontare in base alla categoria e calcolare poi la lunghezza delle altre barre come percentuali; in questo modo la distribuzione maggiore rappresenta il 100% della lunghezza disponibile nella barra, mentre tutte le altre barre hanno lunghezze proporzionali al rapporto fra il numero di articoli della categoria e il numero più alto di articoli di tutte le categorie.

```
// Calcola il valore massimo tra i due array di dati  
const maxVal = Math.max(...d3Data.data[0], ...d3Data.data[1]);  
  
// Calcola le lunghezze delle barre come percentuali  
const barLengthsData0 = d3Data.data[0].map((value) => (value /  
maxVal) * 100);  
const barLengthsData1 = d3Data.data[1].map((value) => (value /  
maxVal) * 100);
```

Codice 10: snippet di codice per calcolare le lunghezze delle barre in general framing

Tuttavia, è corretto fare una precisazione riguardo la realizzazione di questo grafico, che rappresenta, tra l'altro, un esempio del naturale processo di evoluzione che caratterizza lo sviluppo di un progetto software.

Il grafico general framing nasce inizialmente come progetto *standalone*, ovvero strutturato in un'unica pagina HTML che aveva come unico scopo quello di fare da "contenitore" per il grafico. Per la sua realizzazione è stata scelta la libreria D3.js che ha permesso di creare un grafico interattivo e dinamico, manipolando direttamente il DOM per gestire gli elementi grafici e le loro proprietà in maniera altamente funzionale.

Il progetto è però progredito e si è trasformato nella homepage attuale. E' quindi emersa la necessità di integrare il grafico nel sito sviluppato con Next.js e Tailwind CSS, rendendo il grafico parte integrante della piattaforma. A questo punto, dopo attente riflessioni, è stato deciso di non utilizzare D3.js all'interno della web-app, in favore di strumenti nativi di Next.js e Tailwind.

Il motivo principale dietro questa scelta è che sia D3.js che React, (che sta alla base di Next.js), manipolano il DOM.

Nello specifico, D3.js lo fa in maniera *imperativa*, permettendo al programmatore di specificare esattamente il comportamento di ogni elemento all'interno del DOM, decidendo quindi, quando e se l'elemento in questione va creato, modificato o eliminato; ciò comporta tutti i benefici analizzati in precedenza, consentendo un controllo completo sugli elementi che vengono creati.

React invece, adotta una logica *dichiarativa*, non modificando effettivamente il DOM. Lo sviluppatore decide cosa vuole creare e React utilizza invece il *virtual DOM* per determinare le operazioni necessarie. Il virtual DOM è una rappresentazione virtuale della struttura della pagina immagazzinata in memoria e del tutto simile al DOM originale, del quale può essere vista come una astrazione. Quando si verifica un evento che necessita una risposta da parte della pagina, React modifica il DOM virtuale e confronta in maniera estremamente veloce il virtual DOM con quello "reale", determinando i cambiamenti effettivi da apportare.

Considerando le modifiche delle due tecnologie al DOM, emerge quindi la possibilità che D3.js e Next.js entrino in conflitto portando a comportamenti del software inaspettati e possibili bug in fase di produzione. Inoltre, in questo caso specifico, Tailwind permette di definire il design del grafico in maniera più chiara e semplice.

```
import d3Data from "../.././d3graph.json";

export default function D3Graph() {
  // Calcola il valore massimo tra i due array di dati
  const maxVal = Math.max(...d3Data.data[0], ...d3Data.data[1]);

  // Calcola le lunghezze delle barre come percentuali
  const barLengthsData0 = d3Data.data[0].map((value) => (value /
```

```

maxVal) * 100);
  const barLengthsData1 = d3Data.data[1].map((value) => (value /
maxVal) * 100);

  return (
    <>
      <div className="dropSide flex flex-col 2xl:flex-row text-xs
justify-between">
        {/* Dropdown menu */}
        <div className=" mx-3 2xl:m-3">
          <select id="drop1" className=" w-full sm:w-56 lg:mr-5">
            <option value="all">{d3Data.series[0]}</option>
          </select>
          <select id="drop2" className=" md:w-[410px] w-full mt-2">
            <option value="classifier">{d3Data.series[1]}</option>
          </select>
        </div>
      </div>
      {/* total1 */}
      <div id="all">
        <div className="totalPieces mt-5">
          <p id="value"></p>
          <div className="flex justify-between mx-5 text-xs">
            <p>Total documents</p>
            <p>2786200</p>
          </div>
          <div id="total"></div>
        </div>
        <div className="totalPieces">
          <p id="value2"></p>
          <div className="flex justify-between mx-5 mt-1 text-xs">
            <p>Total relevant</p>
            <p>194746</p>
          </div>
          <div id="total"></div>
        </div>
      </div>
      {/* Categorie e valori */}
      <div>

```

```

    {d3Data.labels.map((labels, index) => (
      <div key={index} className=" m-5">
        <div className="flex justify-between">
          <p className="text-black text-xs mb-2">{labels}</p>
          <p className="text-[#03b8ff] text-xs mb-2 text-right">
            {d3Data.data[0][index]} <span
className="text-black">/</span>{" "}
            <span className="text-[#42cf66] text-xs">
              {d3Data.data[1][index]}
            </span>
          </p>
        </div>
        {/* Barre di confronto */}
        <div className="relative">
          <div className="h-full w-full border-t-4 border-dotted
border-[#32323270]"></div>
          <div
            className="absolute top-0 left-0 h-1 bg-[#03b8ff]"
            style={{
              width: `${barLengthsData0[index]}%`,
            }}
          ></div>
          <div className="h-full w-full border-t-4 border-dotted
border-[#32323270] mt-2"></div>
          <div
            className="absolute top-3 left-0 h-1 bg-[#42cf66]"
            style={{
              width: `${barLengthsData1[index]}%`,
            }}
          ></div>
        </div>
      </div>
    )))
  </div>
</>
);
}

```

Codice 11: componente D3Graph.js

Il codice finale, realizzato grazie al solo impiego di Javascript e Tailwind CSS, risulta facilmente comprensibile e facilmente personalizzabile; nel caso si volessero apportare delle modifiche agli stili del grafico basterebbe soltanto modificare le classi CSS, e Tailwind rende questo processo particolarmente rapido, in quanto delle modifiche in vanilla CSS richiederebbero molte più linee di codice.

3.3.6.3 Radar chart

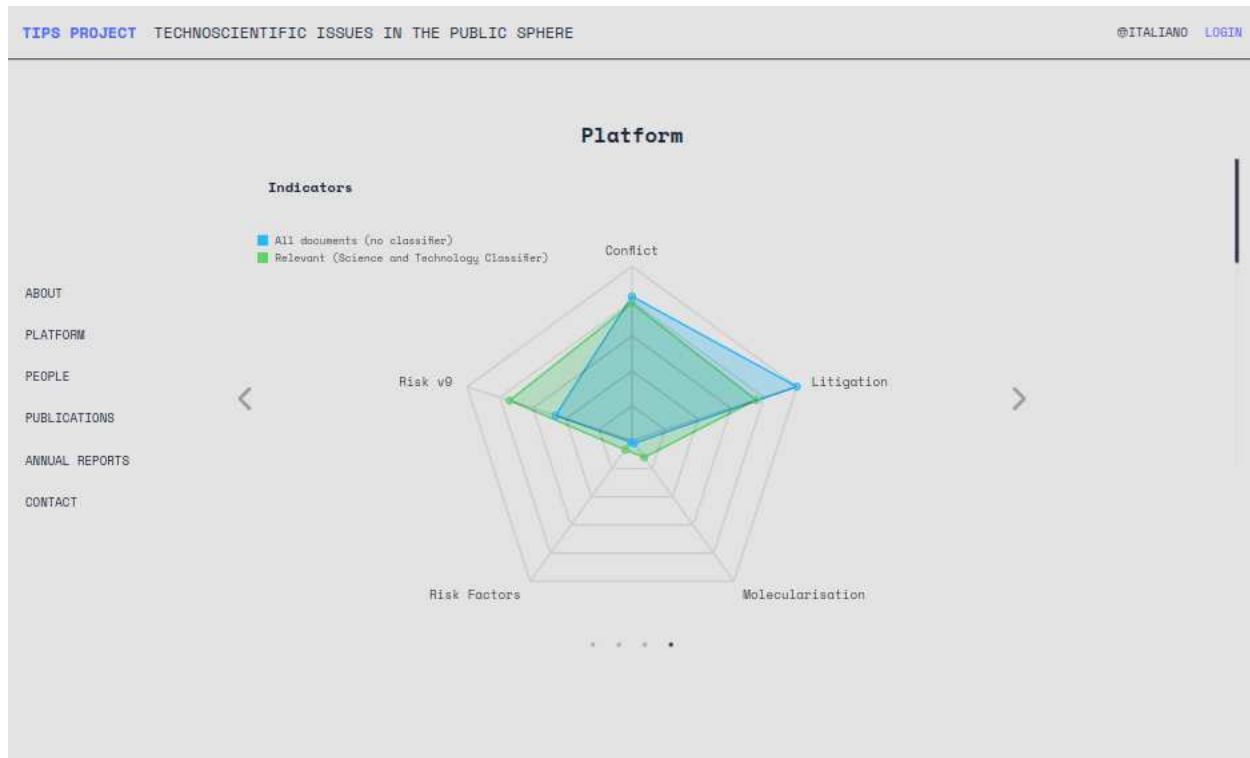


Figura 21: radar chart realizzato con la libreria Nivo

L'ultima scheda del carosello è dedicata a un radar chart costruito utilizzando la libreria Nivo. Questo grafico mostra una serie di indicatori calcolati in base al contenuto degli articoli, in particolare utilizzando informazioni relative alla frequenza con cui compaiono alcuni termini legati allo specifico indicatore nel documento. Ad esempio, per l'indicatore *rischio* il vocabolario contiene i termini come *risch**, *pericol**, *danno*, *danni*, *dannos**, *dannegg**; l'asterisco indica che si conterà la presenza di quel termine se una parola nel testo dell'articolo inizia con la sequenza di caratteri prima dell'asterisco. Il calcolo degli indicatori riportati nel radar chart non si basa unicamente sulla frequenza ma prende in considerazione altre proprietà degli articoli e del corpus, come ad esempio la lunghezza dell'articolo o la lunghezza media degli articoli nel corpus. Una descrizione dettagliata è riportata in (Di Buccio et al., 2016) e (Neresini et al., 2019).

La realizzazione di questo grafico, dato l'utilizzo della stessa libreria, appare molto simile a quella del line chart mostrato in precedenza.

```
import { ResponsiveRadar } from "@nivo/radar";
import radarData from "../../../../../radarchart.json";

export default function RadarChart() {

return(
<>
<ResponsiveRadar
  data={radarData}
  keys={[
    "All documents (no classifier)",
    "Relevant (Science and Technology Classifier)",
  ]}
  indexBy="labels"
  valueFormat=" >-.2f"
  margin={{ top: 80, right: 50, bottom: 20, left: 50 }}
  gridShape="linear"
  gridLabelOffset={18}
  dotSize={7}
  dotColor={{ theme: "background" }}
  dotBorderWidth={2}
  colors=[["#03b8ff", "#42cf66"]]
  borderWidth={2}
  blendMode="multiply"
  motionConfig="wobbly"
  legends=[ /* personalizzazioni grafiche */ ]
  theme={{ /* personalizzazioni grafiche */ }}
  />
</div>
</>
);}
```

Codice 12: componente radarchart.js

In questo caso la libreria importata è stata `@nivo/radar`, infatti Nivo permette di importare solo la libreria legata alla tipologia di grafico che si vuole realizzare; ciò

consente di non appesantire il progetto con codici che non vengono utilizzati dallo sviluppatore.

Le impostazioni grafiche sono lievemente differenti rispetto al line chart, perché in base alla tipologia di grafico che si deve rappresentare bisogna modificare aspetti differenti, come per esempio l'attributo `blendMode="multiply"` che regola come i colori delle aree devono comportarsi se vengono sovrapposti con altre aree. In un grafico in cui non vi è sovrapposizione di aree questa opzione risulta inutile e appesantisce il codice.

Entrambi i grafici realizzati tramite Nivo comprendono, oltre che i codici mostrati, anche una funzione per cambiare la dimensione del testo in base alla dimensione della finestra su cui veniva visualizzato il grafico. Dato che questi grafici erano posizionati all'interno delle schede del carosello, capitava che parole particolarmente lunghe venissero tagliate dalla visualizzazione della scheda. Questa funzione vale, tra l'altro, anche se l'utente dovesse ridimensionare la pagina mantenendo lo stesso dispositivo.

```
import { useState, useEffect } from "react";
//altre importazioni in base alla tipologia di grafico da utilizzare

// Stato per gestire la larghezza della finestra
const [windowWidth, setWindowWidth] = useState(0);

// Funzione per aggiornare la dimensione della finestra
const handleResize = () => {
  setWindowWidth(window.innerWidth);
};

// Aggiorna la dimensione della finestra al montaggio del
componente e quando la finestra cambia dimensioni
useEffect(() => {
  setWindowWidth(window.innerWidth);
  window.addEventListener("resize", handleResize);

  return () => {
    window.removeEventListener("resize", handleResize);
  };
});
```

```

    };
  }, []);

  // Funzione per scalare la dimensione del font in base alla
  // larghezza della finestra
  const scaleFontSize = (baseSize) => {
    if (windowWidth > 1200) {
      return baseSize * 1.2; // Schermi grandi
    } else if (windowWidth > 768) {
      return baseSize; // Tablet e desktop
    } else {
      return baseSize * 0.8; // Schermi piccoli
    }
  };
};

```

Codice 13: funzione per ridimensionare il testo nei grafici Nivo

In entrambi i codici sono stati importati gli hook *useState* e *useEffect* visti in precedenza; e proprio tramite *useState* viene impostato `const [windowWidth, setWindowWidth] = useState(0)`; inizialmente a zero, questo stato tiene traccia della larghezza della finestra corrente. Successivamente viene definita la funzione `handleResize()` che aggiorna il valore di `windowWidth` ogni volta che la finestra cambia dimensione.

useEffect aggiorna la dimensione della finestra al momento del montaggio tramite `setWindowWidth(window.innerWidth)` e quando quest'ultima cambia dimensione, aggiungendo `addEventListener` che "ascolta" l'evento `resize` e richiama `handleResize()` ogni volta che la finestra deve essere ridimensionata. Quando il componente viene smontato l'event listener viene rimosso per evitare problemi di memoria.

Infine la funzione `scaleFontSize()` utilizza il valore di `windowWidth` per calcolare la dimensione del font proporzionata alla larghezza della finestra, moltiplicando la dimensione del font per 1.2 nel caso di schermi particolarmente grandi, e per 0.8 se si tratta di schermi piccoli, in modo da ridurre del 20% la dimensione del font. Per implementare questa funzione basta passare il valore di grandezza desiderata alla funzione come in `fontSize: ScaleFontSize(10)`, permettendo di visualizzare lo stesso testo con una grandezza di 8 su schermi piccoli e 12 su schermi grandi, (mantenendola invariata su schermi medi).

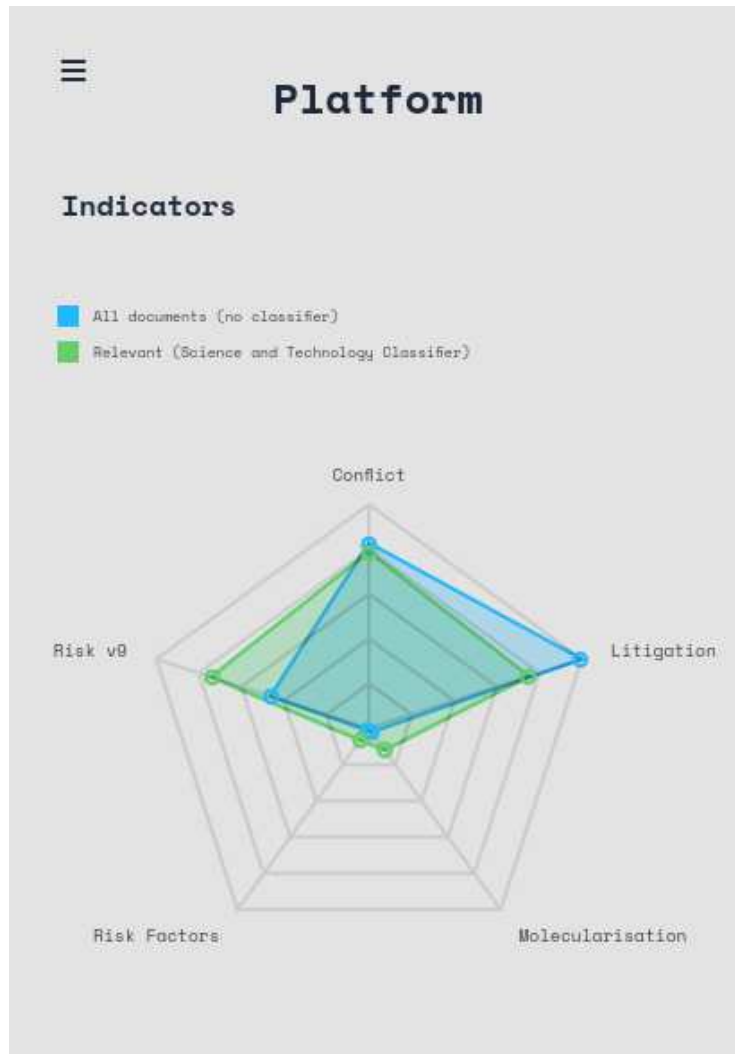


Figura 22: radar chart visualizzato su Pixel 7

Tutti i grafici del carosello sono poi stati suddivisi in componenti e ricongiunti nel componente <Platform />.

```
"use client";

import React from "react";
import { IoIosArrowForward } from "react-icons/io";
import { IoIosArrowBack } from "react-icons/io";
import D3Graph from "./d3graph";
import Slider from "react-slick";
import "slick-carousel/slick/slick.css";
import "slick-carousel/slick/slick-theme.css";
import RadarChart from "./radarchart";
```

```

import LineChart from "./linechart";

function NextArrow(props) {
  /* impostazioni grafiche della freccia per andare avanti */
}
function PrevArrow(props) {
  /* impostazioni grafiche della freccia per andare indietro */
}

export default function Platform() {
  const settings = {
    dots: true,
    infinite: true,
    speed: 500,
    slidesToShow: 1,
    slidesToScroll: 1,
    nextArrow: <NextArrow />,
    prevArrow: <PrevArrow />,
  };

  return (
    <>
      <Slider {...settings}>
        <div className="flex flex-col lg:flex-row">
          <!-- Scheda testuale -->
        </div>
        <div>
          <h2 className="text-base font-semibold justify-between ml-3 mt-3 pl-2.5">
            Salience (percentage of tecnoscintific articles over
            time)
          </h2>
          <LineChart />
        </div>
        <div>
          <h2 className="text-base font-semibold justify-between ml-3 mt-3 pl-0.5">
            General Framing (number of articles per section category)
          </h2>
        </div>
      </Slider>
    </>
  );
}

```

```

        <D3Graph />
      </div>
      <div>
        <h2 className="text-base font-semibold justify-between ml-3
mt-3">
          Indicators
        </h2>
        <RadarChart />
      </div>
    </Slider>
  </>
);
}

```

Codice 14: componente platform.js

Nel componente viene utilizzato `"use client"` perché viene sempre aggiornato ciò che si vede sullo schermo quando si scorrono le schede del carosello. Vengono importati tutti i componenti da visualizzare oltre che due componenti della libreria `react-icons/io` per aggiungere le icone che permettono di scorrere il carosello a destra e a sinistra. Tali icone sono in realtà già presenti di default nella libreria utilizzata, tuttavia per una questione di coesione dal punto di vista del design sono state cambiate. Le icone verranno poi personalizzate tramite le funzioni `NextArrow` e `PrevArrow`.

Per l'implementazione del carosello è stato scelto di utilizzare la libreria esterna `react-slick`, dato che snellisce particolarmente il corpo del codice; si è rivelata una scelta particolarmente efficace, in quanto per costruire il carosello viene usato il tag `<Slider></Slider>` al cui interno verranno renderizzate un numero di schede uguale al numero di elementi `<div></div>` presenti. La forza di questa libreria è che, a differenza di molti caroselli presenti online, non pone limite alla quantità e alla tipologia di elementi che possono essere renderizzati all'interno di una singola scheda. Ciò ha permesso una struttura più ordinata del codice, in cui all'interno di singoli `<div></div>` sono presenti i componenti creati con Next.js, cosicché da garantire un migliore aggiornamento e manutenibilità del codice.

3.3.7 People

La sezione mostra i ricercatori coinvolti nel progetto TIPS, con dei placeholder in cui potrebbero essere inserite delle immagini, e sottostante, il nome della persona con il relativo ruolo all'interno del progetto. La struttura è infatti rimasta invariata rispetto alla vecchia versione, aggiornando solo il design dei placeholder delle foto.

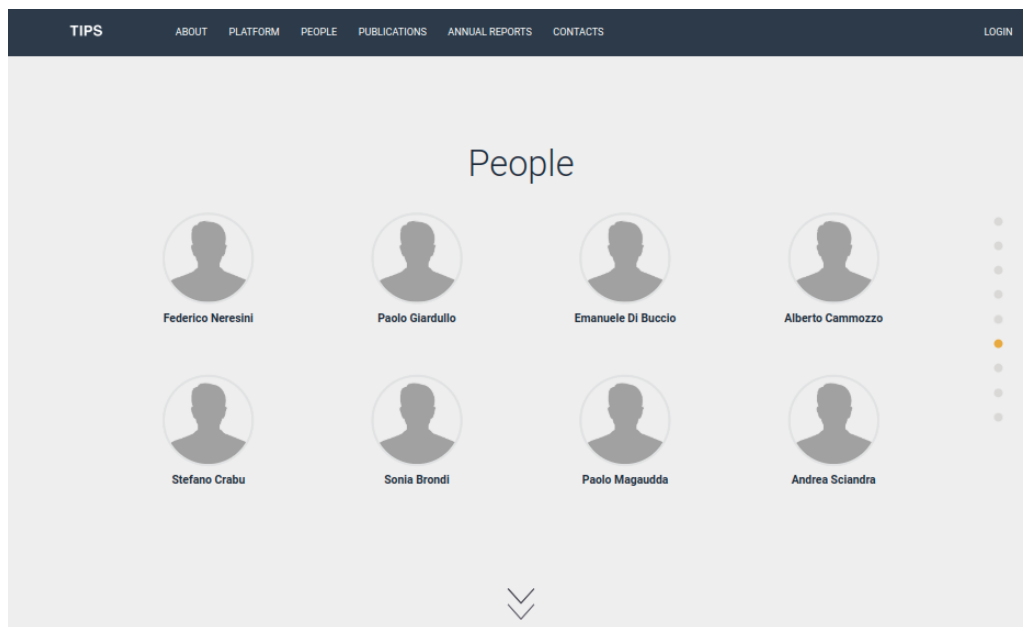


Figura 23: sezione people nella vecchia versione

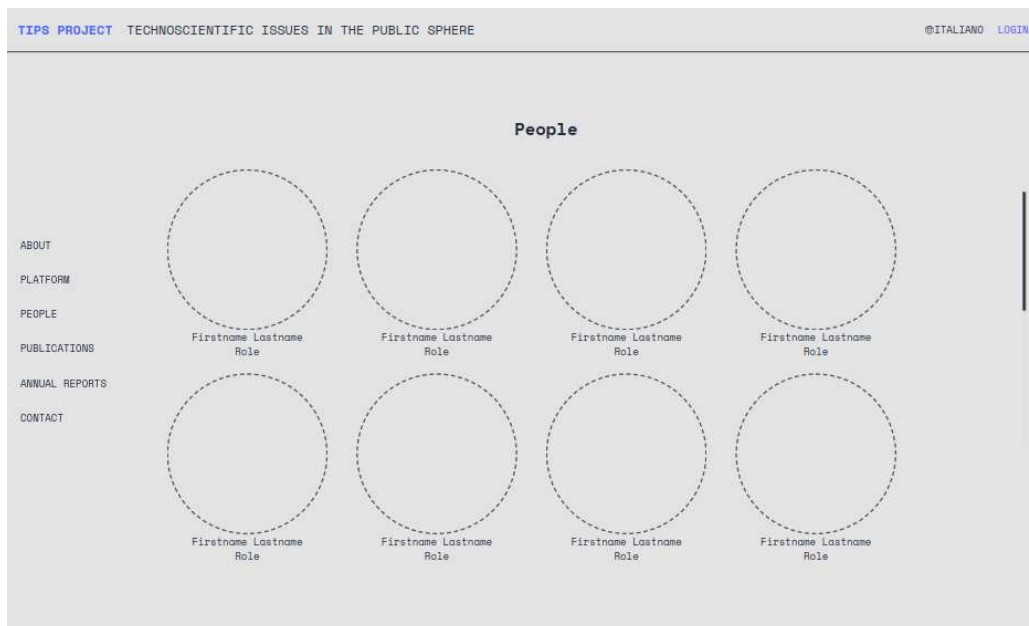


Figura 24: sezione people nella nuova versione

Le indicazioni ricevute per la nuova versione erano di rendere i cerchi più grandi e di usare un bordo tratteggiato per indicarne la presenza.

Nonostante non sia cambiato quasi nulla fra le due versioni a livello desktop, se si osserva la sezione su mobile si nota una certa differenza, data da due comportamenti molto diversi nella disposizione degli elementi.

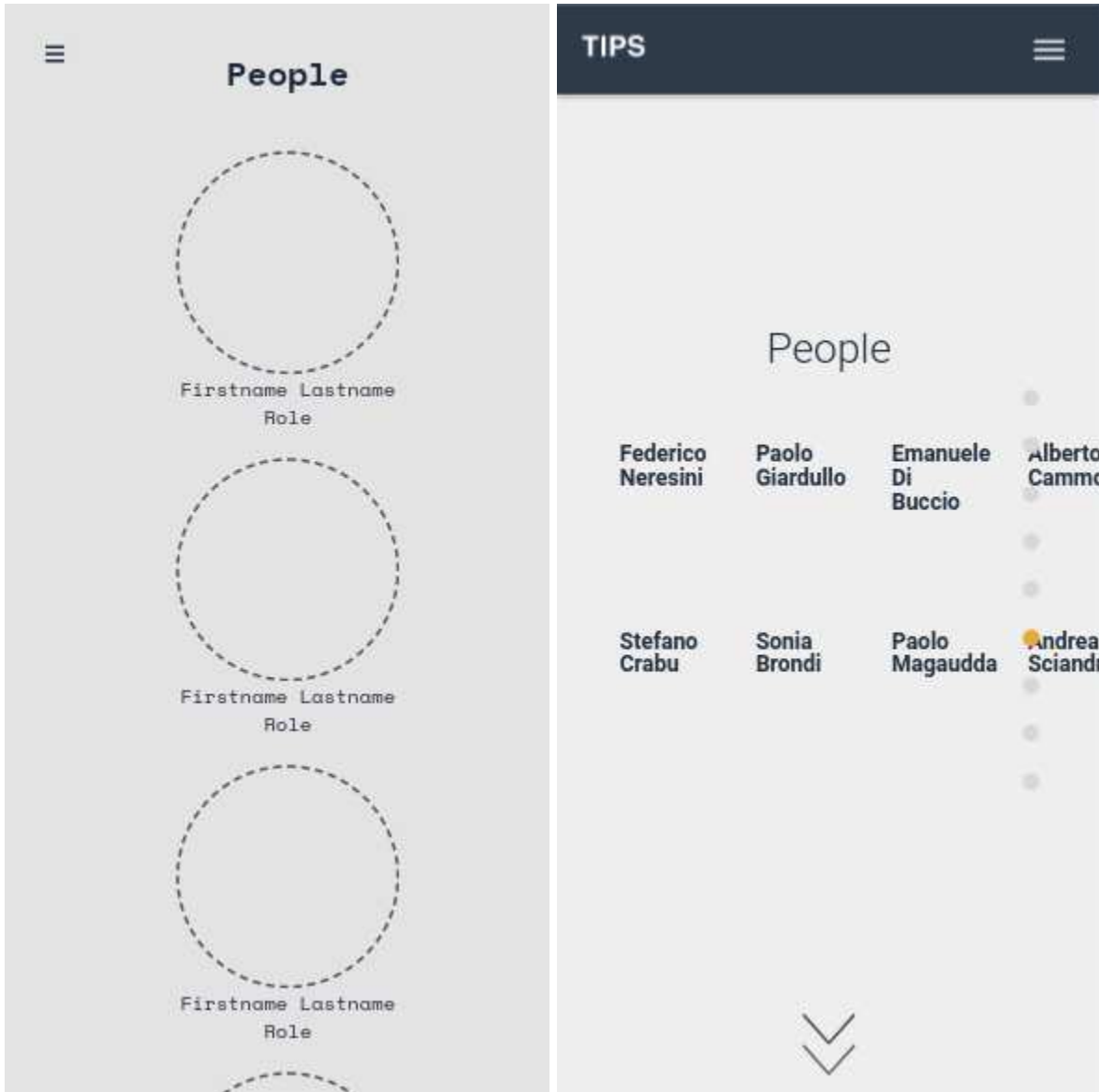


Figura 25: confronto della visualizzazione mobile della sezione people

In precedenza, il codice della sezione non comprendeva un comportamento diverso per la visualizzazione sui dispositivi mobili. Questo comporta che la sezione, sviluppata orizzontalmente, non disponga di sufficiente spazio per una corretta visualizzazione. Comportando: un *overflow* di elementi nel lato destro che vengono tagliati

impedendone la corretta visualizzazione, la sovrapposizione dei nomi dei ricercatori con lo scroll indicator, (che dell'ultima sezione è andato avanti graficamente di due punti), e la scomparsa delle immagini dei placeholder, che in questo caso può essere sopperita data l'assenza di foto presenti nel sito, ma questa informazione verrebbe comunque a mancare nel caso il nome di un ricercatore fosse accompagnato dalla sua foto. Questa sezione rappresenta un esempio concreto di design non responsive. Il problema è stato risolto nella nuova versione grazie all'utilizzo delle *flexbox*.

```
export default function People() {
  return (
    <>
      <div className="flex flex-col mx-10 md:ml-[10%] lg:ml-0
2xl:ml-[0]">
        <div className="rounded-full h-40 w-40 xl:h-56 xl:w-56
2xl:h-72 2xl:w-72 border-dashed border-2 border-gray-600 "></div>
        <p className="text-sm text-center">Firstname Lastname</p>
        <p className="text-sm text-center mb-5">Role</p>

        <div className="rounded-full h-40 w-40 xl:h-56 xl:w-56
2xl:h-72 2xl:w-72 border-dashed border-2 border-gray-600"></div>
        <p className="text-sm text-center">Firstname Lastname</p>
        <p className="text-sm text-center mb-5">Role</p>
      </div>
    </>
  );
}
```

Codice 15: componente people.js

Flexbox (Flexible Box Layout), è una tecnologia di CSS che permette di creare layout di vario genere in maniera intuitiva. Risulta particolarmente utile per gestire la disposizione spaziale degli elementi, che possono a loro volta contenere altri elementi, diventando così contenitori; permettendo la creazione di design particolarmente complessi.

Una delle proprietà delle flexbox è *flexwrap*, attivo di default nelle flexbox, che consente di non far comprimere gli elementi all'interno del contenitore e andare a capo una volta esaurito lo spazio utile per visualizzarli. Un'altra proprietà utile è *flex-direction*, che permette di definire la direzione con la quale ordinare gli elementi.

Il Codice 15 mostra che i placeholder sono renderizzati due per volta e sono all'interno di un `<div></div>` che possiede il layout flex e una flex-direction *column*, che indica che gli elementi saranno disposti in colonna dall'alto verso il basso. I placeholder sono dati da `rounded-full` e `border-dashed border-2`, tratteggiando il bordo dei cerchi che potrebbero essere riempiti in futuro dalle foto dei ricercatori.

Il componente `<People />` viene ripetuto quattro volte all'interno di una flexbox con flex-direction *row*, ovvero disponendo gli elementi in una riga ordinata da sinistra a destra, generando otto cerchi identici ed equidistanti. Dato che i cerchi sono renderizzati due per volta uno sotto l'altro e che flexwrap è attivo di default, negli schermi più piccoli vengono visualizzati tutti e otto in un'unica colonna, se le dimensioni dello schermo crescono si arriva a quattro file da due cerchi per riga, fino alla visualizzazione desktop completa.

All'interno del codice si notano delle classi come `xl:h-56` oppure `lg:m1-0`. Queste regole CSS scritte con Tailwind permettono di gestire comportamenti diversi degli elementi in base alla dimensione della finestra. Il primo esempio è utilizzato per gestire la grandezza dei placeholder, (quindi l'altezza dei cerchi), in base allo spazio disponibile. Il secondo compie gli stessi compiti ma per quanto riguarda i margini di sinistra della flexbox. I prefissi `sm`, `md`, `lg`, `xl` e `2xl`, indicano la larghezza minima che deve possedere una finestra, per effettuare determinate modifiche; questo approccio consente di adottare un'ottica *mobile first*, e poi usando i prefissi per cambiare il design man mano che la dimensione dei dispositivi cresce, questa logica induce lo sviluppatore a pensare la struttura dei design prima per i dispositivi più piccoli, risultando in dei componenti completamente responsive.

3.3.8 Publications

La sezione publications contiene le più recenti pubblicazioni dei ricercatori che collaborano con il progetto TIPS. In questo caso, dal punto di vista dei contenuti, le sezioni sono uguali ma strutturalmente presentano sostanziali differenze.

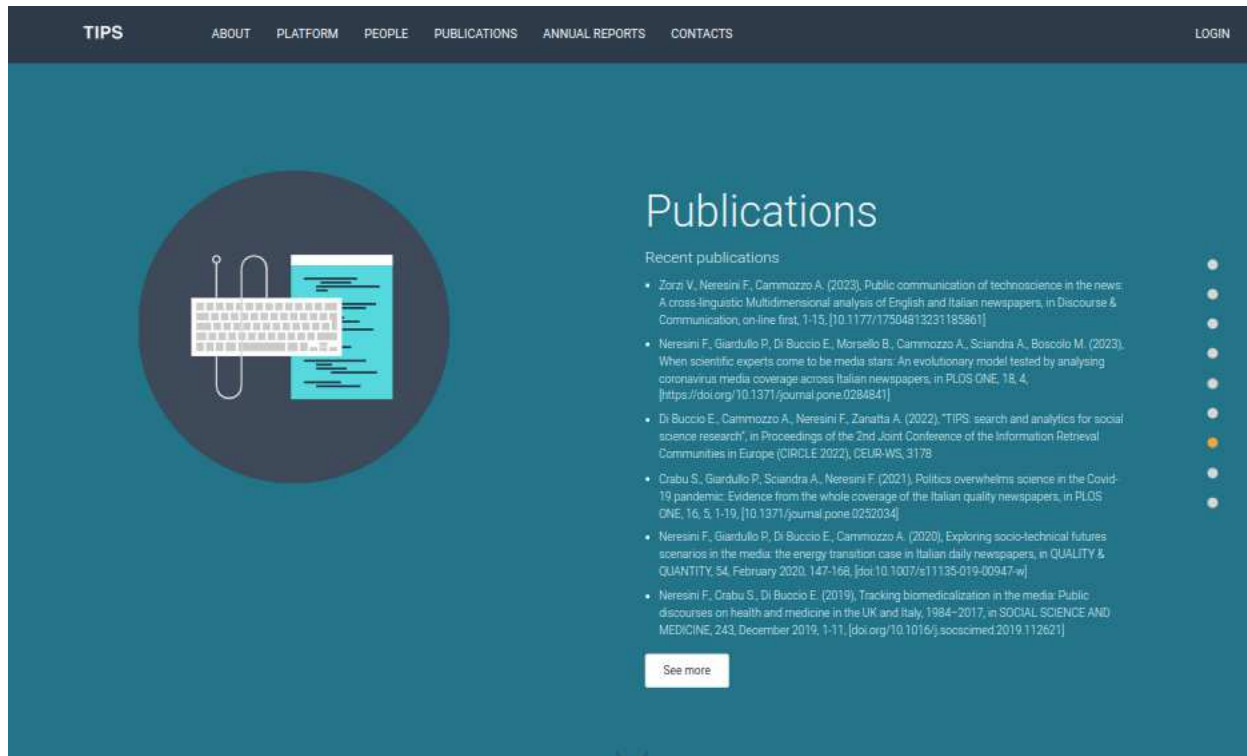


Figura 26: sezione publications nella vecchia versione

TIPS PROJECT TECHNOSCIENTIFIC ISSUES IN THE PUBLIC SPHERE		Publications			©ITALIANO LOGIN
	AUTHORS	YEAR	TITLE	PUBLISHED IN	
ABOUT	Neresini F. Giardullo P. Di Buccio E. Morsello B. Cammozzo A. Sciandra A. Boscolo M.	2023	When scientific experts come to be media stars: An evolutionary model tested by analysing coronavirus media coverage across Italian newspapers	PLOS ONE	
PLATFORM					
PEOPLE	Zorzi V. Neresini F. Cammozzo A.	2023	Public communication of technoscience in the news: A cross-linguistic Multidimensional analysis of English and Italian newspapers	Discourse & Communication	
PUBLICATIONS					
ANNUAL REPORTS					
CONTACT	Di Buccio E. Cammozzo A. Neresini F. Zanatta A.	2022	"TIPS: search and analytics for social science research"	Proceedings of the 2nd Joint Conference of the Information Retrieval Communities in Europe (CIRCLE 2022)	
	Crabu S. Giardullo P. Sciandra A. Neresini F.	2021	Politics overwhelms science in the Covid-19 pandemic: Evidence from the whole coverage of the Italian quality newspapers	PLOS ONE	

Figura 27: sezione publications nella nuova versione

Precedentemente la sezione presentava un'immagine e una lista di pubblicazione scritte nello stile di una bibliografia. Nella nuova versione è stata abbandonata l'immagine ed è stato scelto di seguire lo stile della pagina e di visualizzare le pubblicazioni in una tabella in maniera decisamente più pulita e ordinata; il nuovo design può sicuramente giovare agli utenti, che riusciranno con più facilità a navigare la sezione per cercare informazioni specifiche.

```
"use client";

import { useState, useEffect } from "react";
import publicationsData from "../../../publications.json";
import { FontAwesomeIcon } from "@fortawesome/react-fontawesome";
import { faArrowRight } from "@fortawesome/free-solid-svg-icons";
import Link from 'next/link'

export default function Publications() {
  const [publications, setPublications] = useState([]);

  useEffect(() => {
    const filteredPublications = publicationsData
      .filter((pub) => pub.inHomepage)
      .sort((a, b) => b.year - a.year)
      .slice(0, 5)
      .map(({ authors, year, title, venue }) => ({
        authors,
        year,
        title,
        venue,
      }));

    setPublications(filteredPublications);
  }, []);

  return (
    <>
      <div className="flex flex-col w-full 2xl:px-44 xl:px-24
lg:px-10 px-5 my-16">
```

```

    <h1 className="text-2xl text-center font-bold mb-10
">Publications</h1>

    <div className="hidden md:flex md:justify-between
border-solid border-b-[1px] border-black mx-64 font-semibold text-sm
pb-4">
    <!-- titoli colonne della tabella -->
    {publications.map((pub, index) => (
    <div key={index} className="hidden md:flex md:flex-col">
    <div className="hidden md:flex md:justify-between mx-64
text-sm">
        {/* nuova riga */}
        <div className="my-5 w-2/12 mr-10">
            {pub.authors.map((author, i) => (
                <p key={i}>`${author.lastName}
${author.firstName.substring(0, 1)}.` </p>
            ))}
        </div>
        <h3 className="my-5 w-2/12">{pub.year}</h3>
        <h3 className="my-5 w-4/12 mr-5">{pub.title}</h3>
        <h3 className="my-5 w-4/12">{pub.venue}</h3>
    </div>
    <div className="mx-64 border-solid border-b-[1px]
border-black my-3"></div>
    </div>
    ))}

    {/* mobile */}
    {publications.map((pub, index) => (
    <div key={index} className="justify-center md:hidden mx-16
my-3">
    <!-- visualizzazione mobile -->
    </div>
    ))}

    <div className="flex justify-center mb-5">
    <Link> <!-- see more button --> </Link>
    </div>
</div>

```

```
    </>
  );
}
```

Codice 16: componente publications.js

Per ottenere la logica della nuova versione sono state effettuate delle operazioni con le stringhe di testo, utilizzando i metodi. I metodi sono funzioni di Javascript che consentono di manipolare gli oggetti. Ad esempio, nel caso di oggetti di tipo string, consentono di manipolare e analizzare le stringhe di testo.

L'hook `useEffect` filtra i dati delle pubblicazioni, mostrando solo quelli che presentano l'attributo `inHomepage`. Vengono poi ordinati per anno in ordine decrescente con il metodo `.sort((a, b))` e infine vengono limitati ai primi cinque risultati utilizzando il metodo `.slice(0, 5)`.

I dati vengono *mappati* utilizzando il metodo `.map`, che consente di applicare determinate funzioni a un array di elementi, creandone un altro con delle regole precise. In questo vengono estratti i campi `authors, year, title, venue` da ogni pubblicazione, per evitare di avere informazioni superflue e per preparare i dati da visualizzare. I dati vengono poi salvati nello stato `publications` tramite `setPublications`.

Durante il rendering degli autori i nomi vengono formattati in `<p key={i}>`${author.lastName} ${author.firstName.substring(0, 1)}.`` `</p>` e vengono visualizzati uno sotto l'altro. Il codice combina il cognome completo con l'iniziale del nome seguita da un punto. Ciò comporta una leggibilità maggiore, soprattutto nel caso di svariati autori.

Nella versione mobile è stata rimossa la tabella ed è stata scelta una visualizzazione più compatta, risolvendo anche il problema delle sovrapposizioni degli elementi adottando un design responsive.

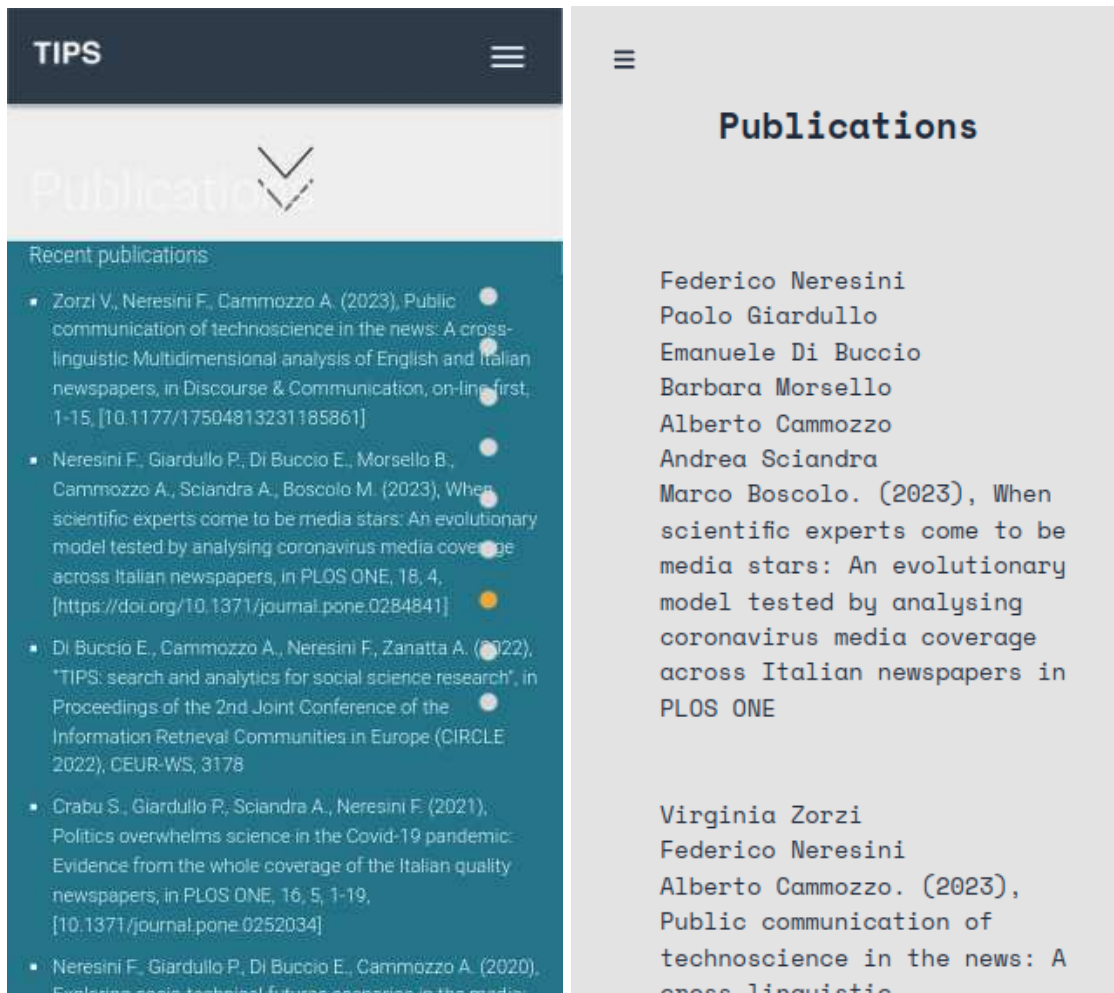


Figura 28: confronto delle versioni della sezione publications visualizzata su mobile

3.3.9 Annual reports

Lo scopo della sezione è quello di informare l'utente dell'esistenza e della struttura dei *TIPS Report*, report annuali pubblicati dal progetto TIPS in cui vengono presentati alcuni dei risultati delle analisi della copertura mediatica della tecnoscienza da parte dei principali quotidiani italiani, utilizzando grafici, tabelle e contenuti aggiornati ogni anno.



Figura 29: sezione annual reports nella vecchia versione



Figura 30: sezione annual reports nella nuova versione

Anche in questo caso è stata rimossa l'immagine in favore di una tabella simile a quella presente nella sezione publications, che mostra i titoli e gli anni degli ultimi report stilati. Il corpus del testo è rimasto invariato, è stata tuttavia invertita la disposizione, posizionando il testo a sinistra e l'elemento della tabella a destra.

```
"use client";

import { FontAwesomeIcon } from "@fortawesome/react-fontawesome";
import { faArrowRight } from "@fortawesome/free-solid-svg-icons";
import Link from 'next/link';

export default function Reports() {
  return (
    <>

      <div className="flex flex-col items-center justify-between px-5 md:px-44 xl:px-64 ">
        <div className="">
          <div className="flex flex-col xl:flex-row justify-center items-start md:space-x-10 mx-auto">

            <div className="w-full justify-center">

              <h1 className="text-2xl text-center lg:text-left font-bold mt-16 lg:mt-0 mb-8 pl-4 pr-6 w-7/8">
                Annual Reports
              </h1>
              <p className="text-center lg:text-left text-sm mb-12 leading-loose pl-4 pr-6 w-7/8"> <!-- testo della sezione --> </p>
            </div>

            <div className=" m1-12 w-2/3 ">

              <div className="hidden md:flex text-xs font-semibold">
                <h2 className="w-1/6">YEAR</h2>
                <h2 className="w-4/6">TITLE</h2>
              </div>
              <div className="w-full border-solid border-b-[1px]>
```

```

border-black"></div>

    <div className="flex flex-col md:flex-row my-5
items-start">
    <h3 className="hidden md:flex text-sm
w-1/6">2024</h3>
    <h3 className="text-sm md:w-4/6">
    NUOVE SFIDE E VECCHI RICORDI
    </h3>
    </div>
    <div className="w-full border-solid border-b-[1px]
border-black"></div>

    <div className="flex flex-col md:flex-row my-5
items-start">
    <h3 className="hidden md:flex text-sm
w-1/6">2023</h3>
    <h3 className="text-sm md:w-4/6">OLTRE LA
PANDEMIA</h3>
    </div>
    <div className="w-full border-solid border-b-[1px]
border-black"></div>

    <div className="flex flex-col md:flex-row my-5
items-start">
    <h3 className="hidden md:flex text-sm
w-1/6">2022</h3>
    <h3 className="text-sm md:w-4/6">2022: E&apos; ANCORA
PANDEMIA</h3>
    </div>
    <div className="w-full border-solid border-b-[1px]
border-black"></div>

    <div className="flex flex-col md:flex-row my-5
items-start">
    <h3 className="hidden md:flex text-sm
w-1/6">2021</h3>
    <h3 className="text-sm md:w-4/6">L&apos;ANNO DELLA
PANDEMIA</h3>

```

```

        </div>
        <div className="w-full border-solid border-b-[1px]
border-black"></div>

    </div>

</div>
</div>
<div className="flex justify-center mb-5">
  <Link
    className="mt-10 bg-black/70 text-slate-50 text-xs w-fit
px-2 p-1"
    href="/reports"
  >
    SEE MORE
    <FontAwesomeIcon icon={faArrowRight} className="ml-2" />
  </Link>
</div>
</div>
</>
);
}

```

Codice 17: componente reports.js

Il codice del componente `<Report />` è strutturato in maniera molto semplice. Non è presente nessun tipo di codice Javascript o funzione particolare; tutto il lavoro viene svolto dall'HTML e Tailwind CSS, che ha permesso di rendere questa sezione responsive ed eliminare i problemi di visualizzazione relativi alla vecchia versione.

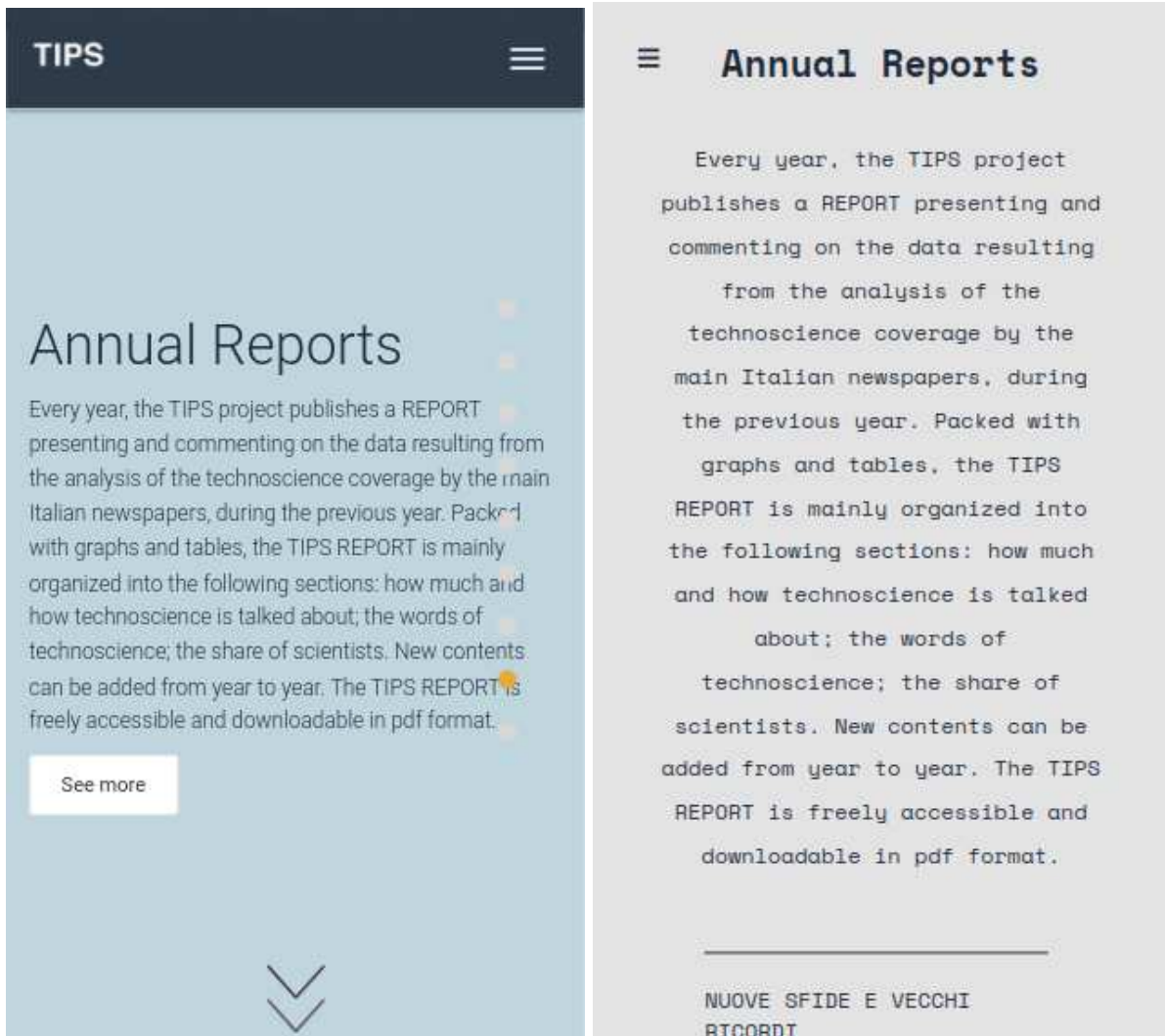


Figura 31: confronto delle versioni della sezione annual reports visualizzata su mobile

Anche in questo caso il corpo del testo si sovrapponeva con lo scroll indicator. Inoltre nella vecchia versione è stata rimossa l'immagine, ma nella nuova è stata mantenuta la possibilità di visualizzare la tabella; posizionandola sotto il testo e mostrando solo i titoli, senza l'anno di riferimento.

Osservando con più attenzione i Codici 16 e 17, ci si accorge che in entrambe le parti finali è presente il componente Next.js `<Link></Link>` e che entrambi i codici iniziano con `"use client"`; tuttavia, prima di spiegare il perché di queste scelte è opportuno introdurre il concetto di *layout* in Next.js.

Online è possibile trovare una moltitudine di siti al cui loro interno sono presenti pagine molto simili tra loro a livello strutturale; se si naviga, ad esempio, sul sito

Amazon.com e si effettuano delle ricerche, ci si accorge che l'unica parte del sito che muta è quella relativa ai contenuti delle ricerche. L'header, la barra di ricerca, i filtri e le impostazioni dell'utente rimangono invariati a meno che non vengano cambiati dall'utente. Quando si progetta un sito web con le tecnologie tradizionali, si tende spesso a ripetere interi snippet di codice proprio per questo motivo. Generalmente le applicazioni web hanno quindi un design di base su cui poi vengono visualizzate le singole pagine.

Il layout in Next.js è un componente fondamentale che racchiude le pagine della propria web-app, consentendo di definire una struttura e un aspetto coerente per tutte le pagine del sito in maniera immediata evitando le ripetizioni di codice.

```
import './globals.css';
import Header from './components/header';
import Sidebar from './components/sidebar';
import ScrollIndicator from './components/scrollindicator';

export const metadata = {
  title: "TIPS Homepage",
  description: "Homepage",
};

export default function RootLayout({ children }) {
  return (
    <html lang="it" className="no-scrollbar scroll-smooth">
      <body className="flex flex-col min-h-screen ">
        <Header />
        <div className="flex flex-1">
          <Sidebar />
          <main className="flex-1 overflow-y-auto">{children}</main>
        </div>
        <ScrollIndicator />
      </body>
    </html>
  );
}
```

Codice 18: layout completo della pagina

Il codice del componente `layout.js` risulta essere molto semplice. Vengono importati i componenti necessari e il file `./globals.css` per gestire gli stili condivisi in tutta l'applicazione. Vengono inseriti dei metadati di base, come il titolo e la descrizione della pagina; che possono essere successivamente modificati per migliorare il SEO per i motori di ricerca. Il cuore del codice è rappresentato dal componente `RootLayout` che definisce la struttura principale della pagina. Al suo interno si trova l'elemento `<html></html>` che presenta le classi `scroll-smooth` e `no-scrollbar` per una navigazione fluida tra le sezioni del sito e per l'eliminazione della convenzionale scrollbar, in favore del più elegante scroll Indicator.

Nel `<body></body>` del codice sono presenti gli elementi che verranno sempre mostrati anche nelle altre pagine del sito, (a meno che non si voglia fare diversamente), ovvero `<Header />`, `<Sidebar />`, e `<ScrollIndicator />`. Il contenuto dinamico della homepage è passato tramite la proprietà `{children}`. In questo modo i tre componenti principali rimarranno visibili e invariati senza bisogno di riscrittura da parte dello sviluppatore.

L'applicazione del componente `layout` si può notare proprio quando viene utilizzato il componente `<Link></Link>`. Normalmente per la navigazione tra le pagine viene utilizzato il tag `<a>`, ma il componente `<Link></Link>` permette di pre-caricare automaticamente le pagine collegate senza dover ricaricare la pagina, rendendo la navigazione tra le diverse sezioni dell'applicazione molto più veloce, migliorando l'esperienza utente e le prestazioni dell'applicazione.

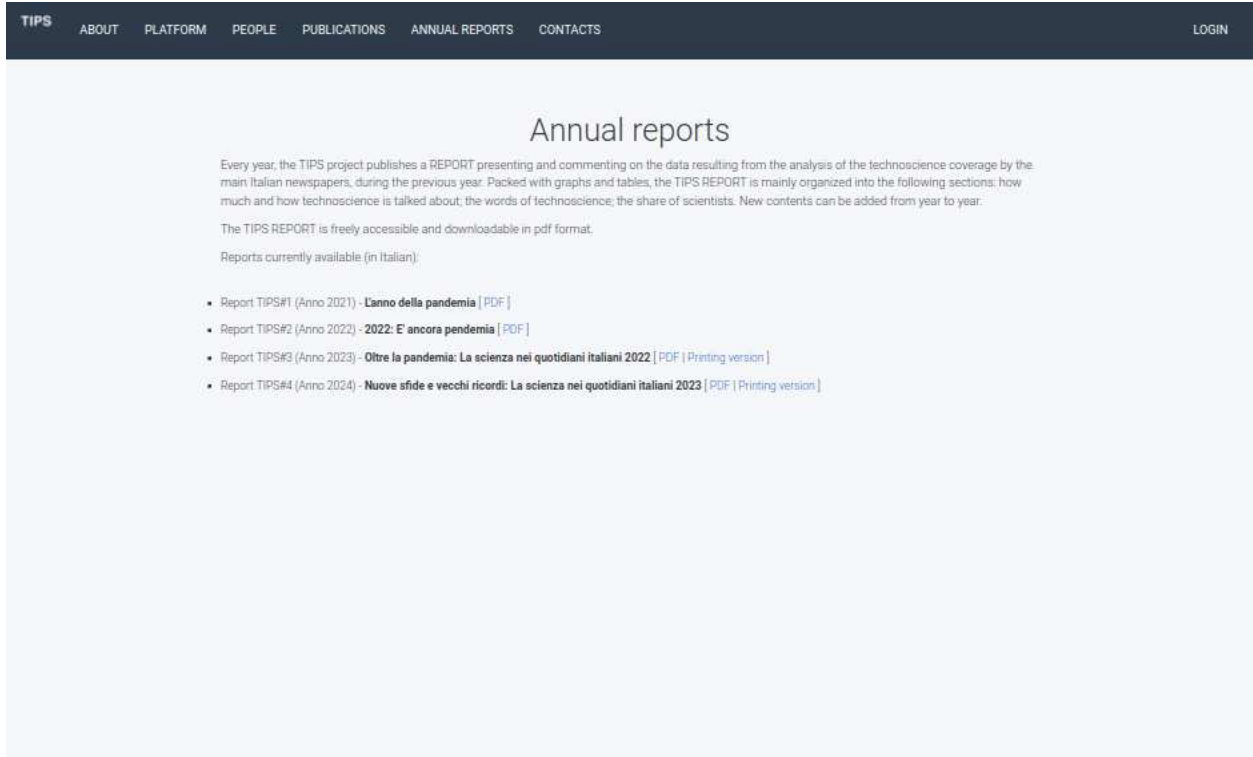


Figura 32: pagina annual reports nella vecchia versione

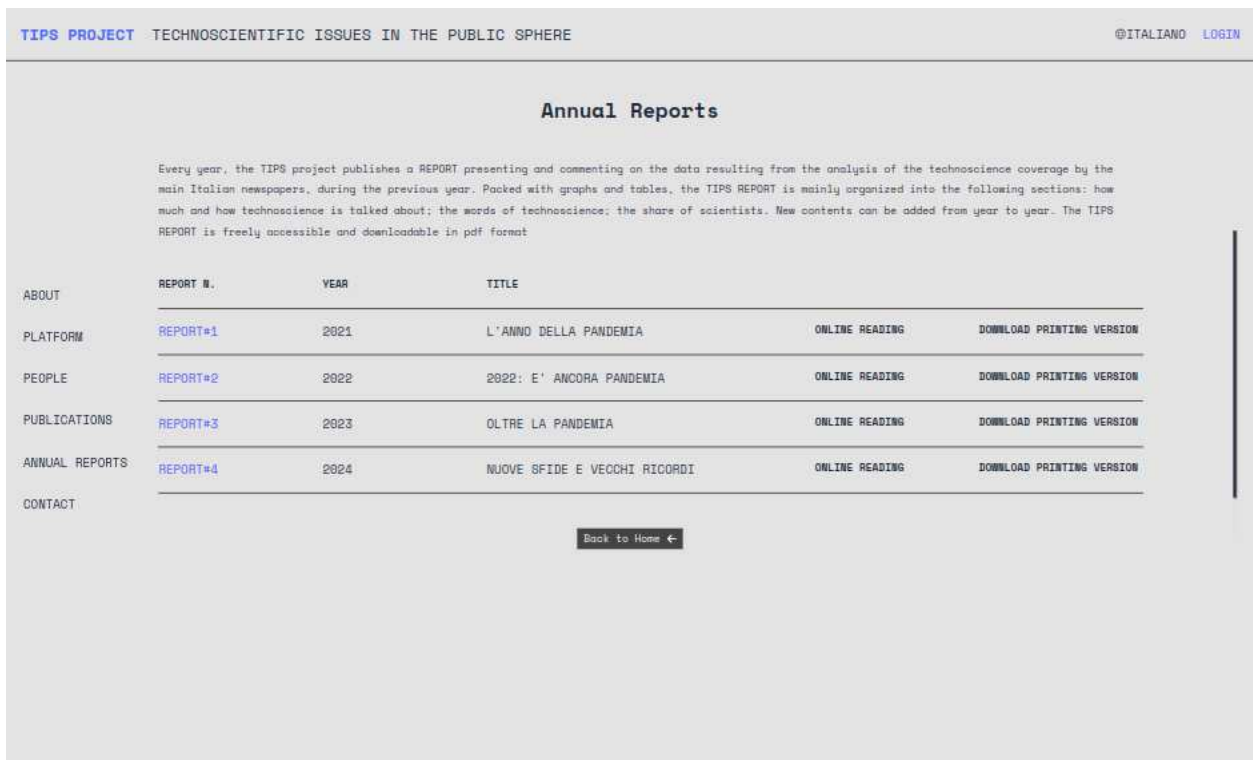


Figura 33: pagina annual reports nella nuova versione

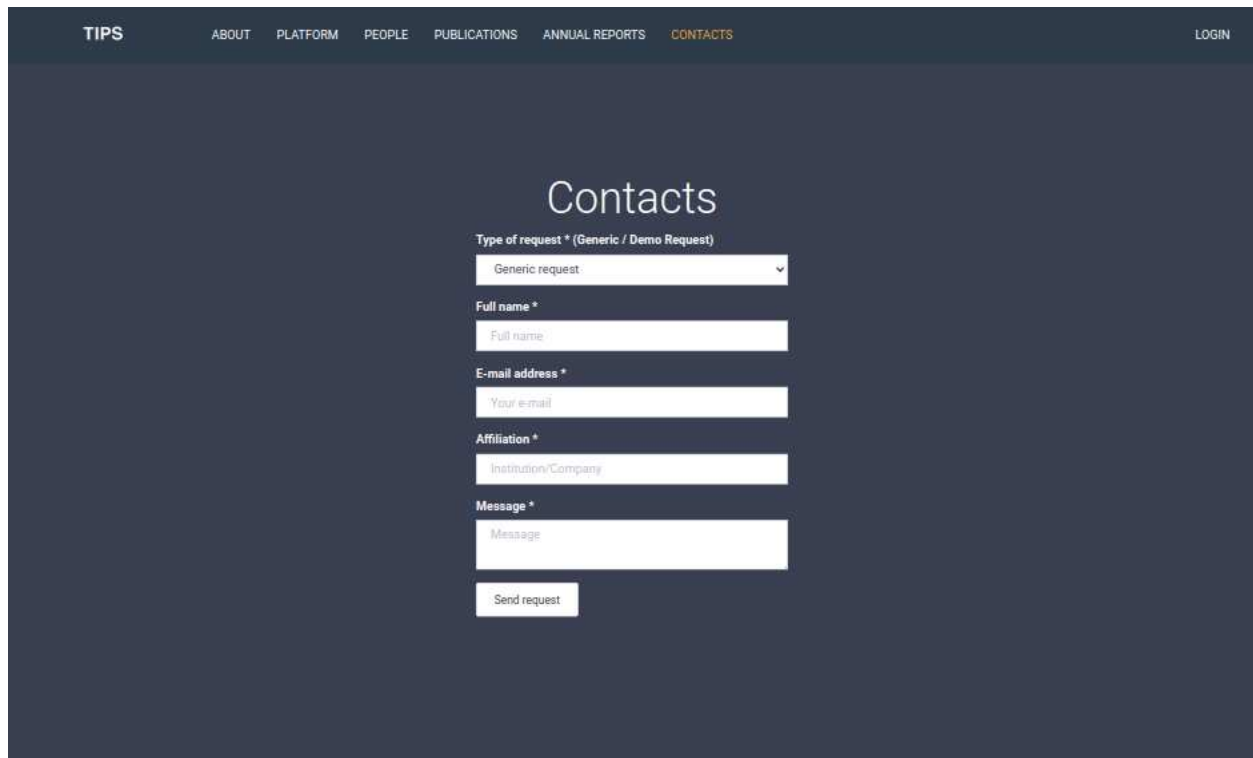
Precedentemente, la pagina dedicata ai report annuali aveva uno stile ben diverso dalla sezione corrispondente, risultando in pagina anonima senza un design pensato; inoltre, nonostante non sia presente l'header, non vi è effettivamente un tasto per tornare alla pagina precedente.

La nuova versione presente uno stile perfettamente coerente con la homepage, riportando header e scroll indicator immutati nella nuova pagina; l'indicatore di scroll presenta lo stesso design e la funzione Javascript con cui è stato realizzato si adatta perfettamente alla nuova pagina, aggiornando il riempimento dell'indicatore di conseguenza.

E' stato poi mantenuto il corpo del testo per intero e la tabella è stata ampliata dando la possibilità agli utenti di visionare e scaricare i report, fornendo un tasto per tornare all'homepage alla fine del contenuto della pagina

3.3.10 Contacts

Infine la sezione contacts contiene un form da compilare per mettersi in contatto con i ricercatori del progetto TIPS.



The screenshot shows a dark-themed website with a navigation bar at the top containing the following links: TIPS, ABOUT, PLATFORM, PEOPLE, PUBLICATIONS, ANNUAL REPORTS, CONTACTS, and LOGIN. The 'CONTACTS' link is highlighted in orange. The main content area features a 'Contacts' form with the following fields and a button:

- Type of request * (Generic / Demo Request)**: A dropdown menu with 'Generic request' selected.
- Full name ***: A text input field with the placeholder 'Full name'.
- E-mail address ***: A text input field with the placeholder 'Your e-mail'.
- Affiliation ***: A text input field with the placeholder 'Institution/Company'.
- Message ***: A text input field with the placeholder 'Message'.
- Send request**: A button located below the message field.

Figura 34: sezione contacts nella vecchia versione

TIPS PROJECT TECHNOSCIENTIFIC ISSUES IN THE PUBLIC SPHERE @ITALIANO LOGIN

Contact Us

ABOUT

PLATFORM

PEOPLE

PUBLICATIONS

ANNUAL REPORTS

CONTACT

TYPE OF REQUEST * (GENERIC / DEMO REQUESTS)

Generic request

FULL NAME *

Full Name

E-MAIL ADDRESS *

Your e-mail

AFFILIATION *

Institution/Company

MESSAGE *

Your message

SEND

Figura 35: sezione contacts nella nuova versione

Le due versioni sono molto simili: si è solo adeguato il form alle scelte di design fatte nell'intera pagina.

Alla fine, per la completa realizzazione della homepage sono stati creati diversi componenti, che grazie a Next.js possono essere combinati per formare un codice principale molto più chiaro di come sarebbe normalmente.

```
import Image from "next/image";
import About from "./components/about";
import People from "./components/people";
import Publications from "./components/publications";
import Contacts from "./components/contacts";
import Reports from "./components/reports";
import Platform from "./components/platform";

export default function Home() {
  return (
    <>
    <div className="scroll-smooth">
```

```

    { /* HomeImage section */ }
    <section
      className="min-h-screen flex items-center justify-center"
      id="image"
    >
      <div className="flex justify-center items-center md:w-4/6
">
        <Image
          src="/homeimage.png"
          alt="Home image"
          width={500}
          height={500}
          className="w-full h-auto"
        />
      </div>
    </section>
    { /* About section */ }
    <section
      className="min-h-screen flex items-center justify-center"
      id="about"
    >
      <About />
    </section>
    { /* Platform section */ }
    <section className="flex justify-center pt-48" id="platform">
      <div className=" w-3/5 ">
        <h1 className="text-2xl text-center font-bold mb-5
">Platform</h1>
        <Platform />
      </div>
    </section>
    { /* People section */ }
    <section
      className="min-h-screen flex items-center justify-center
pt-8 ml-16"
      id="people"
    >
      <div className="flex flex-col w-4/5">
        <h1 className="text-2xl text-center font-bold mb-10

```

```

mt-10">
    People
  </h1>
  <div className="flex flex-wrap justify-center ">
    <People />
    <People />
    <People />
    <People />
  </div>
</div>
</section>
{ /* Publication section */}
<section
  className="min-h-screen flex items-center justify-center
pt-32"
  id="publications"
>
  <Publications />
</section>
{ /* Annual reports section */}
<section className="pt-48 ml-1" id="reports">
  <Reports />
</section>
{ /* Contact form */}
<section
  className="min-h-screen flex items-center justify-center"
  id="contact"
>
  <Contacts />
</section>
</div>
</>
);
}

```

Codice 19: codice completo di page.js

All'inizio del Codice 19 vengono importati tutti i componenti relativi alle sezioni specifiche del sito, alcuni di essi sono formati da più componenti a loro volta, ma è sufficiente importare il componente genitore per poterlo renderizzare correttamente.

Poi all'interno del componente `Home()` sono presenti i tag `<section></section>` che permettono di default di navigare tra le sezioni di un sito HTML e contengono, in ordine di visualizzazione, tutte e sei le sezioni presenti nella homepage. Ogni sezione è preceduta da eventuali titoli o regole di stile valide per l'intera sezione, susseguite dal nome del componente. Così Next.js permette di creare web-app anche particolarmente complesse, in maniera ordinata e intuitiva.

Si nota anche che il codice della pagina è renderizzato lato server: infatti importando dei componenti con cui l'utente deve interagire e che quindi necessitano una renderizzazione lato client (CSR), l'homepage non cessa di poter essere renderizzata lato server. Ciò significa che con Next.js è possibile ottenere la velocità iniziale e la SEO del Server-Side-Rendering, ed allo stesso tempo la fluidità di navigazione e l'interattività del Client-Side-Rendering. Questo rende il framework uno strumento particolarmente potente, versatile e semplice.

4 Conclusioni e sviluppi futuri

Il lavoro di tesi ha permesso di sviluppare la nuova homepage del progetto TIPS. Per la realizzazione di quest'ultima sono stati adottati due framework moderni, ovvero Next.js e Tailwind CSS. Next.js ha permesso di strutturare il sito in componenti React riutilizzabili, e di migliorare le prestazioni del sito tramite l'utilizzo congiunto del Server-Side-Rendering e del Client-Side-Rendering. Tailwind invece, ha consentito di implementare con estrema facilità lo stile di design scelto direttamente nel codice della pagina. Inoltre, durante lo sviluppo, sono state rispettate le linee guida fornite dal relatore e dai collaboratori del progetto, garantendo però un design responsivo per ogni tipo di schermo.

Dato il focus del progetto TIPS nell'analizzare il discorso su scienza e tecnologia, all'interno della landing page sono stati implementati dei grafici utilizzando le librerie D3.js e Nivo, per ottenere delle visualizzazioni moderne e personalizzate. L'obiettivo era quello di creare delle rappresentazioni visive che fossero facilmente comprensibili agli utenti.

Difatti, una parte rilevante della tesi è stata, oltre allo sviluppo web, l'importanza dell'applicazione della data visualization nel contesto del monitoraggio del discorso su scienza e tecnologia dei quotidiani online. Poter trasformare grandi quantità di dati in rappresentazioni visive significative, risulta essere particolarmente importante nel panorama contemporaneo. Inoltre, la piattaforma, non solo facilita il monitoraggio mediatico, ma propone diverse prospettive di analisi dei contenuti, evidenziando pattern e tendenze difficilmente identificabili. La natura multidisciplinare del progetto, consente quindi di unire metodi di ricerca sociale a competenze informatiche specifiche, è ciò rende il progetto TIPS un punto di convergenza unico.

In futuro, lo sviluppo della homepage potrebbe progredire con l'obiettivo di implementare nuove funzionalità all'interno del sito, oppure si potrebbe aggiornare l'intero frontend della piattaforma TIPS, che non si limita alla sola homepage, con le tecnologie utilizzate. Sono già state fatte delle proposte per migliorare alcuni aspetti della User Interface della pagina. Successivamente si potrebbe procedere con l'adozione delle librerie Nivo e D3 per migliorare e personalizzare in maniera ottimale i grafici presenti nella piattaforma.

Il progetto svolto nella tesi, oltre a essere di mio interesse e una probabile strada per il proseguimento degli studi, mi è stato personalmente utile per migliorare delle competenze già possedute e per apprenderne altre, che altrimenti non avrei appreso da solo. E' stato quindi molto stimolante poter fare parte di questo progetto, e mi piacerebbe particolarmente avere altre occasioni, in futuro, per poter continuare a contribuire e a migliorare questo aspetto del progetto TIPS.

Ringraziamenti

Questo lavoro rappresenta la conclusione di un percorso importante, che probabilmente non sarebbe stato possibile senza il supporto e la guida di molte persone. Desidero quindi ringraziare in ordine sparso:

Il mio relatore, per la sua pazienza e per l'immenso supporto mostrato, oltre che per avermi fatto avvicinare a un ambito che ha suscitato in me un certo interesse.

La mia famiglia, che mi ha supportato nonostante le distanze da casa.

La mia ragazza, che mi è stata vicino e mi ha regalato momenti di grande gioia in questi anni. Ti amo.

Tutti coloro che ho incontrato a Padova e con cui ho condiviso parte della mia vita, in particolare, Francesco Caruso, Pietro Lauriola, Andrea Zago, Kevin, Amir, Maira Mattiazzi, Jelena Tufonich e Gaia.

Il mio coinquilino, con cui ho gustato i piatti prelibati della mensa migliore di Padova per due anni, mentre vivevamo in una magnifica casa con tanto di cantina.

La mia psicologa, perché se no sarei impazzito.

Infine, un ringraziamento speciale va ai "Boys ®", amici di vecchia data, compagni di mille avventure, che mi hanno accompagnato durante il mio percorso accademico e non, in particolare:

Salvatore Musumeci, per le esperienze di quasi morte e i traumi subiti.

Samuela Maria Costanzo, per essere la ragazza più bella che conosco.

Simone Sinatra, per aver tenuto al sicuro il nostro paese combattendo valorosamente.

Flavio Vasil Anzalone, per portare avanti ideali politici condivisi da tutti noi.

Domenico Fiumazza, per avermi accompagnato in numerosi viaggi.

Alex Price, per avermi fatto scoprire una città in cui non tornerò mai più.

Christian Tracà, per la tua paura di uscire la sera con me.

Davide D'amico, per essere criticato per le tue scelte di vita.

Tyler, per essere stato presente mentre Logan non lo era.

Vi voglio bene.

Bibliografia

- ❖ Di Buccio, E., Cammozzo, A., Neresini, F., & Zanatta, A. (2022). TIPS: Search and Analytics for Social Science Research, CEUR Workshop Proceedings.
- ❖ Neresini, F. (2017, June 21). Old media and new opportunities for a computational social science on PCST. JCOM - Journal of Science Communication. https://jcom.sissa.it/article/pubid/JCOM_1602_2017_C03/
- ❖ De Maria, G. (2024). Analisi e Refactoring di un Sistema di Monitoraggio Media per l'Estrazione di Tematiche da Articoli di Quotidiani, Tesi di Laurea, Dipartimento di Ingegneria dell'Informazione, Università degli Studi di Padova, A.A. 2023/2024. <https://thesis.unipd.it/handle/20.500.12608/72164>
- ❖ Sinar, E. F. (2015). Data visualization. In *Big Data at Work* (pp. 115-157). Routledge.
- ❖ Healy, K. (2018). *Data visualization: a practical introduction*. Princeton University Press.
- ❖ Srivastava, D. (2023). An Introduction to Data Visualization Tools and Techniques in Various Domains. *International Journal of Computer Trends and Technology*, 71(4), 125-30.
- ❖ Felt, U., Fouché, R., Miller, C. A., & Smith-Doerr, L. (Eds.). (2016). *The handbook of science and technology studies*. Mit Press.
- ❖ Rogers, R. (2015). Digital methods for web research. *Emerging trends in the social and behavioral sciences*, 1322.
- ❖ Dakowicz, J. (2022, November 17). Server-Side Rendering vs Client-Side Rendering vs Static Site Generator. Pagepro, <https://pagepro.co/blog/ssr-csr-ssg/>
- ❖ Chęć, D., & Nowak, Z. (2019). The performance analysis of web applications based on virtual DOM and reactive user interfaces. In *Engineering Software Systems: Research and Praxis* (pp. 119-134). Springer International Publishing.
- ❖ Di Buccio, E., Lorenzet, A., Melucci, M., & Neresini, F. (2016). Unveiling Latent States Behind Social Indicators. In *SoGood@ ECML-PKDD*.
- ❖ Neresini, F., Crabu, S., & Di Buccio, E. (2019). Tracking biomedicalization in the media: Public discourses on health and medicine in the UK and Italy, 1984-2017. *Social Science & Medicine*, Dec. 2019, 243:112621. doi: 10.1016/j.socscimed.2019.112621. Epub 2019 Oct 22. PMID: 31677575.

Sitografia

- ❖ *“Web Analytics, Social Media Monitoring e Data Visualization: Trasforma i Dati in Decisioni”*.(2024), presente.consulting,
<https://www.presente.consulting/web-analytics-social-media-monitoring-e-data-visualization-trasforma-i-dati-in-decisioni/>
- ❖ *“What is media monitoring?”*.(2024), lexisnexis.com,
<https://www.lexisnexis.com/en-us/professional/media-intelligence/glossary/media-monitoring-analytics.page>
- ❖ *“What Is Data Visualization? Definition, Examples, And Learning Resources.”*. (2024), tableau.com, <https://www.tableau.com/visualization/what-is-data-visualization>
- ❖ *“TIPS - Technoscientific issues in the public sphere.”*. (n.d.), tipsproject.eu,
<https://www.tipsproject.eu/tips/#/public/home#about>
- ❖ *“L’importanza della UX per un sito web”*. Serravalle, C. (2024, October 25).
Advantè.it,
<https://www.advante.it/marketing/user-experience-importanza-sito-web/>
- ❖ *“Internet Traffic from Mobile Devices Stats (2024)”*. GilPress. (2024, February 4).
whatsthebigdata.com, <https://whatsthebigdata.com/mobile-internet-traffic/>
- ❖ *“Introduction.”* (n.d.). Next.js, <https://nextjs.org/docs>
- ❖ *“The difference between a framework and a library”*. Wozniewicz, B. (2024, September 4). freeCodeCamp.org,
<https://www.freecodecamp.org/news/the-difference-between-a-framework-and-a-library-bd133054023f/>
- ❖ *“Understanding Rendering Methods for Web Apps”*.(n.d.), (2023, January 27). Let Me Fail.com,
<https://letmefail.com/technology/understanding-rendering-methods-for-web-apps/>
- ❖ *“Tailwind CSS: What It Is, Why Use It & Examples”*. Fitzgerald, A. (2022, May 31). HubSpot.blog, <https://blog.hubspot.com/website/what-is-tailwind-css>
- ❖ *“D3js: Rappresentazione di dati e infografiche dinamiche con JavaScript”*. Chiarelli, A. (2012, November 27). HTML.it,
<https://www.html.it/articoli/d3js-rappresentazione-di-dati-e-infografiche-dinamiche-con-javascript/>

- ❖ *“DOM (Document Object Model) - MDN Web Docs Glossary: Definitions of Web-related terms”*. MDN. (2023, August 9). Developer.mozilla, <https://developer.mozilla.org/en-US/docs/Glossary/DOM?retiredLocale=it>
- ❖ *“Building charts in React with Nivo”*. Maldonado, L. (2021, August 11). LogRocket.blog, <https://blog.logrocket.com/building-charts-in-react-with-nivo/>
- ❖ *“Explaining all React Hooks with examples”*. Junior, S. (2021, January 20). DEV Community, <https://dev.to/sergioamjr/explaining-all-react-hooks-with-examples-4jl1>
- ❖ *“International Network on Public Communication of Science and Technology”*. (n.d.). (2013, January 22). PreventionWeb.net, <https://www.preventionweb.net/organization/international-network-public-communication-science-and-technology>
- ❖ *“Science communication network since 1989”*. (n.d.). PCST Network, <https://www.pcst.network/about/history/>
- ❖ *“Javascript HTML DOM”*. (n.d.). (2021). W3schools.com, https://www.w3schools.com/js/js_htmlDOM.asp
- ❖ *“CSS Flexbox: tutorial per utilizzarli”*. Lamanna, C. (2014, January 13). HTML.it, <https://www.html.it/articoli/css-flexbox/>
- ❖ *“A guide to Next.js layouts and nested layouts”*. Mojeed, I. (2024, February 19). LogRocket Blog, <https://blog.logrocket.com/guide-next-js-layouts-nested-layouts/>
- ❖ *“Exponential Growth of Data”*. Phiri, M. (2022, November 19). Medium.com, <https://medium.com/@mwaliph/exponential-growth-of-data-2f53df89124>