PEGASO
Università Telematica

TEMPLATE
PROJECT WORK

Corso di Studio	INFORMATICA PER LE AZIENDE DIGITALI (L-31)
Dimensione dell'elaborato	Minimo 6.000 – Massimo 10.000 parole *(pari a circa Minimo 12 – Massimo 20 pagine)*
Formato del file da caricare in piattaforma	PDF
Nome e Cognome	Giuseppe Paolino Caliendo
Numero di matricola	0312200942
Tema n. (Indicare il numero del tema scelto):	3
Titolo del tema (Indicare il titolo del tema scelto):	Tecnologia web per la sostenibilità d'impresa
Traccia del PW n. (Indicare il numero della traccia scelta):	7
Titolo della traccia (Indicare il titolo della traccia scelta):	Sviluppo di una pagina web per il download dei report di sostenibilità di un'impresa del settore secondario
Titolo dell'elaborato (Attribuire un titolo al proprio elaborato progettuale):	"Consultazione sostenibile: design e sviluppo di una piattaforma per i report di sostenibilità Ferrero"

PARTE PRIMA – DESCRIZIONE DEL PROCESSO

Utilizzo delle conoscenze e abilità derivate dal percorso di studio
(Descrivere quali conoscenze e abilità apprese durante il percorso di studio sono state utilizzate per la redazione dell'elaborato, facendo eventualmente riferimento agli insegnamenti che hanno contribuito a maturarle):

Per la realizzazione dell'elaborato sono state applicate trasversalmente diverse conoscenze acquisite durante il percorso di studi, integrate da esperienze lavorative maturate in ambito professionale e da corsi di formazione esterni all'ateneo.

In particolare, le basi teorico-pratiche della programmazione web sono state acquisite nel corso di "**Tecnologie web**" che ha fornito le competenze necessarie per lo sviluppo di pagine e applicazioni web, sia lato client che lato server. Nello specifico per quello che riguarda il "frontend", sono state approfondite le tecnologie fondamentali per la struttura, lo stile e l'interattività delle pagine come HTML, CSS e Javascript, nonché la gestione dello scambio dati attraverso il formato JSON. Sul versante "backend" il corso ha introdotto alla creazione di server web utilizzando NodeJS e il framework ExpressJS, con particolare attenzione alla gestione delle richieste HTTP e alla progettazione di API RESTful. Tali conoscenze sono state integrate da esperienze lavorative pregresse maturate durante un precedente impiego come sviluppatore backend.

Nel corso "**Comunicazione digitale e social media**" sono stati affrontati temi legati all'usabilità e alla scrittura efficace all'interno di un sito web e all'ottimizzazione dei contenuti per i motori di ricerca (SEO). Questi elementi si sono rivelati essenziali nella progettazione dei testi e dell'interfaccia del sito, con l'obiettivo di rendere la consultazione dei report chiara e accessibile.

Il corso di "**Algoritmi e strutture dati**" ha fornito le basi per la gestione e l'organizzazione delle informazioni risultando utile nella logica di elaborazione e presentazione dinamica dei report provenienti dal server all'interno della pagina dei download. In particolare, ha contribuito alla strutturazione del comportamento dei filtri, alla disposizione dei risultati e alla generazione delle card mediante l'uso di Javascript.
Le conoscenze acquisite nel corso di "Reti di calcolatori e cybersecurity" si sono rivelate fondamentali per comprendere ed applicare i principi dei protocolli HTTP e HTTPS, utilizzati per la comunicazione tra il frontend e il backend attraverso chiamate API, anche quando virtualmente ospitati su macchine differenti. La curiosità stimolata da questo insegnamento mi ha spinto ad approfondire in autonomia il tema delle CORS (Cross-Origin Resource Sharing), rivelatosi poi essenziale per la gestione degli errori di comunicazione tra client e server emersi in fase di sviluppo del sito vista l'architettura detached adottata in fase di progettazione.
Il corso di "Programmazione 2" ha approfondito due tematiche che hanno trovato applicazione diretta nel sito realizzato: In primo luogo l'utilizzo delle strutture dati complesse come gli array e gli oggetti, fondamentali nella costruzione e manipolazione dei dati lato frontend e nell'interazione con il database e strutturazione delle risposte HTTP da restituire al client lato backend. In secondo luogo, l'utilizzo del database relazionale, impiegato per registrare all'interno di una tabella dedicata, le informazioni e i metadati dei diversi report di sostenibilità. I dati sono stati interrogati tramite NodeJs senza l'impiego di ORM o modelli intermedi, ma attraverso query SQL dirette. Per quanto riguarda il backend, sono state integrate ulteriori competenze acquisite attraverso un corso di formazione esterno all'ateneo, grazie al quale è stato affrontato in modo pratico l'utilizzo del sistema di gestione dei pacchetti NPM, impiegato per configurare l'ambiente NodeJs e installare librerie fondamentali come 'express' e 'cors'. In tale contesto è stato anche fatto uso di Git Bash e della piattaforma GitHub per il versionamento del codice e la sua condivisione su più dispositivi, facilitando l'organizzazione del lavoro e garantendo la tracciabilità delle modifiche.
Infine, l'insegnamento "Strategia, organizzazione e marketing" ha fornito gli strumenti concettuali per comprendere il significato della sostenibilità in ambito aziendale e il ruolo strategico dei bilanci di sostenibilità, fornendo così la base teorica su cui si è fondato il lavoro di contestualizzazione dell'elaborato.

Fasi di lavoro e relativi tempi di implementazione per la predisposizione dell'elaborato
(Descrivere le attività svolte in corrispondenza di ciascuna fase di redazione dell'elaborato. Indicare il tempo dedicato alla realizzazione di ciascuna fase, le difficoltà incontrate e come sono state superate):

Inserisci qui il testo

Risorse e strumenti impiegati
(Descrivere quali risorse - bibliografia, banche dati, ecc. - e strumenti - software, modelli teorici, ecc. - sono stati individuati ed utilizzati per la redazione dell'elaborato. Descrivere, inoltre, i motivi che hanno orientato la scelta delle risorse e degli strumenti, la modalità di individuazione e reperimento delle risorse e degli strumenti, le eventuali difficoltà affrontate nell'individuazione e nell'utilizzo di risorse e strumenti ed il modo in cui sono state superate):

Inserisci qui il testo

PARTE SECONDA – PREDISPOSIZIONE DELL'ELABORATO

Obiettivi del progetto
(Descrivere gli obiettivi raggiunti dall'elaborato, indicando in che modo esso risponde a quanto richiesto dalla traccia):

Inserisci qui il testo

Contestualizzazione
(Descrivere il contesto teorico e quello applicativo dell'elaborato realizzato):

Inserisci qui il testo

Descrizione dei principali aspetti progettuali
(Sviluppare l'elaborato richiesto dalla traccia prescelta):

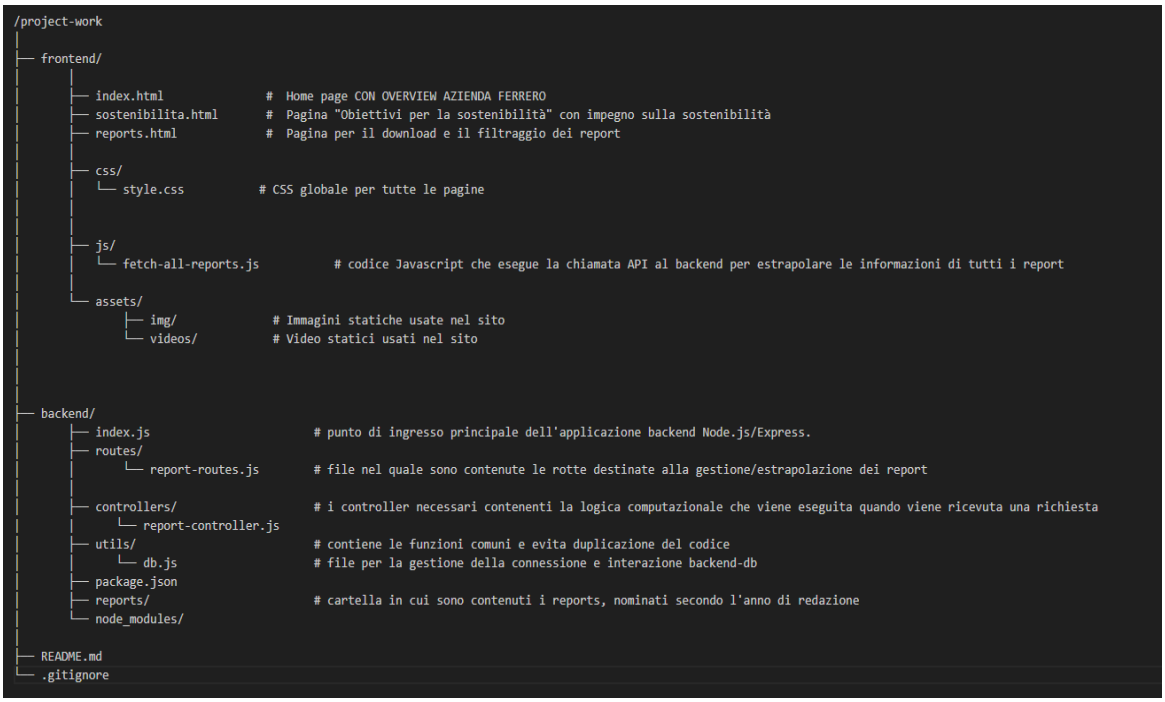
Nota sullo stato dei lavori:

L'elaborato presentato in questa sede è da considerarsi in fase di sviluppo.
Lo scopo di questa versione intermedia è quello di fornire una panoramica dello stato di avanzamento dei lavori con l'obiettivo di ricevere una valutazione preliminare e indicazioni utili per la prosecuzione.

INTRODUZIONE:

L'obiettivo del progetto è lo sviluppo di un sito web per la consultazione e il download dei report di sostenibilità di un'impresa del settore secondario, in questo caso **FERRERO**, con particolare attenzione alla chiarezza dell'interfaccia e all'usabilità complessiva, all'organizzazione dei contenuti e alla fruibilità delle informazioni.
Il sito è strutturato secondo un'**architettura detached**, che prevede la separazione tra frontend e backend, i quali comunicano attraverso chiamate API basate sul protocollo HTTP.
Al momento, i codici sorgente di entrambe le componenti **non sono ancora ospitati su un server accessibile online**, pertanto il progetto è **eseguibile unicamente in ambiente locale**.
In questa fase del lavoro, sono già stati definiti i linguaggi e le tecnologie principali, avviata l'implementazione delle pagine web e sviluppate le funzionalità di base per la gestione dei dati.
Nei paragrafi seguenti vengono illustrate l'architettura generale del sistema, le tecnologie adottate e lo stato di avanzamento di frontend, backend e database, accompagnate da alcuni screenshot e riferimenti al codice sorgente.

- Il repository del progetto è ospitato su GitHub, è pubblico e liberamente consultabile al seguente indirizzo: https://github.com/caliendogiuseppe/project-work
- La struttura delle cartelle del progetto allo stato attuale è la seguente:

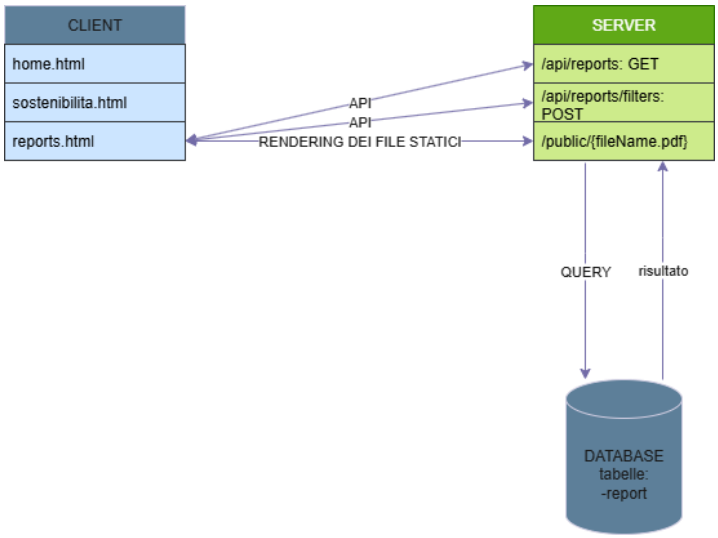


ARCHITETTURA GENERALE DEL SISTEMA

Il progetto è strutturato secondo un'architettura **detached**, in cui il **frontend** e il **backend** operano in ambienti distinti e comunicano tra loro tramite richieste HTTP.

- Il **frontend** è stato sviluppato utilizzando **HTML, CSS e JavaScript vanilla.** Attualmente è stata introdotta soltanto una libreria esterna per il rendering dei loghi dei social network di Ferrero all'interno del footer del sito.
- Il **backend** è realizzato con **Node.js** e **Express.js**, e fornisce API RESTful per l'accesso ai dati. Si fa uso di **npm** per la gestione e l'importazione di pacchetti e dipendenze necessarie. Attualmente i pacchetti npm utilizzati sono:
  - express: impiegato per la configurazione semplificata del server backend responsabile della gestione delle richieste HTTP provenienti dal frontend.
  - mysql2: impiegato per l'interazione tra il backend e il database relazionale, al momento impiegato esclusivamente per l'esecuzione di query di lettura (SELECT);
  - cors: necessario per abilitare le richieste cross-origin tra frontend e backend, in quanto ospitati su domini differenti (N.B. in fase di sviluppo locale il dominio è lo stesso: http://localhost/ ma cambia la porta).
- I dati sono gestiti tramite un **database relazionale SQL**, progettato per contenere i metadati estratti dai report nella tabella predisposta "reports".

Questo tipo di architettura è stato scelto per la sua flessibilità e per la possibilità di agevolare futuri sviluppi (ad esempio, l'integrazione con versioni mobile o API esterne), in quanto consente a più team di lavorare in parallelo sulle diverse componenti del sistema. Di seguito la rappresentazione grafica dell'architettura:



Il sito, allo stato attuale, non prevede funzionalità per l'aggiunta, la modifica o l'eliminazione dei report direttamente tramite interfaccia web.
La gestione dei dati avviene esclusivamente mediante interrogazioni SQL dirette sul database, operazione che richiede la presenza di MySQL installato localmente sulla macchina in uso.

TECNOLOGIE UTILIZZATE (da completare)

Le principali tecnologie adottate finora sono:

- HTML5 / CSS3 / JavaScript per il frontend
- NodeJs con ExpressJs per il backend
- npm come sistema di gestione dei pacchetti
- Git e GitHub per il versionamento del codice
- SQL per la gestione dei dati, attraverso query dirette
- Git Bash come terminale per operazioni locali

Le librerie express e cors sono state installate per creare il server e gestire correttamente le richieste cross-origin. Il repository del codice è ospitato su GitHub (privato/pubblico), e contiene sia il codice del frontend che quello del backend.

FRONTEND (da completare)

Attualmente sono state create tre pagine principali:

- **home**: con una breve introduzione al sito
- **sostenibilità**: una sezione informativa sugli obiettivi ESG Ferrero
- **reports**: la pagina centrale, dove verranno visualizzati i report in formato "card", e dove sarà possibile applicare filtri (per anno, produzione, ecc.)

index.html

L'interfaccia utente del sito è progettata utilizzando esclusivamente tecnologie web standard: HTML5 per la struttura, CSS3 per lo stile visivo e JavaScript per l'interattività (nelle pagine che la prevedono).
Attualmente, è stata completata la pagina Home, che introduce il visitatore al progetto Ferrero e presenta visivamente l'azienda attraverso testi istituzionali, dati di sintesi e un video introduttivo.
STRUTTURA ATTUALE:
La pagina index.html è organizzata in **sezioni semanticamente distinte**, ognuna con una chiara funzione comunicativa:

- **Navbar**: posizionata in alto, permette la navigazione tra le tre pagine del sito (Home, Sostenibilità, Report). È costruita con Flexbox e include il logo SVG di Ferrero, importato inline.
- **Sezione introduttiva con video**: un'area full-screen che mostra un video in autoplay e loop, oscurato da un overlay semi-trasparente, con sovrapposto un titolo animato (h1, h2 e paragrafo). Questa sezione comunica i valori fondanti dell'azienda.
- Sezione "Il gruppo Ferrero": descrive la storia dell'impresa, evidenziando le sue radici italiane e la crescita a livello globale. È strutturata con titoli e paragrafi centrati, per enfatizzare il contenuto testuale.
- **Sezione "I numeri del nostro gruppo"**: include tre card con immagini e testi che evidenziano i dati chiave dell'azienda (fatturato, dipendenti, presenza globale). Ogni card è costruita con classi dedicate (.card, .card--title, .card--p, ecc.) ed è accompagnata da un'immagine descrittiva.
- **Footer**: suddiviso in tre blocchi principali:
    - una lista di link testuali,
    - il logo Ferrero in SVG (bianco su sfondo scuro),
    - un set di icone sociali (Instagram, LinkedIn, Facebook), importate tramite Font Awesome.

sostenibilita.html <non ancora realizzata>

È prevista la realizzazione di una seconda pagina denominata "Sostenibilità", il cui obiettivo sarà quello di fornire un inquadramento generale sull'impegno ambientale e sociale di Ferrero, accompagnato da una panoramica sull'evoluzione degli obiettivi sostenibili dell'azienda nel tempo.

In particolare, la pagina conterrà sezioni dedicate a:

- la descrizione dei pilastri della strategia ESG di Ferrero (ambiente, persone, ingredienti, approvvigionamento);
- l'elenco sintetico dei principali traguardi raggiunti e dei miglioramenti attuati negli anni, anche in risposta alle nuove direttive internazionali;
- un possibile confronto temporale per evidenziare il progresso continuo nella riduzione dell'impatto ambientale e nella valorizzazione delle risorse umane e naturali;
- eventuali elementi visivi (icone, grafici, timeline) a supporto della chiarezza e immediatezza del contenuto.

La pagina sarà progettata nel rispetto della coerenza stilistica con il resto del sito, adottando la stessa palette cromatica, lo stesso font e una struttura a sezioni scrollabili. Potrà essere arricchita da immagini o citazioni tratte dai report ufficiali Ferrero per rafforzare il messaggio comunicativo.

reports.html

La pagina reports.html rappresenta il cuore funzionale del sito e consente all'utente di consultare e scaricare i report di sostenibilità Ferrero filtrandoli in base a specifici criteri.

È stata progettata con una logica modulare e orientata all'esperienza utente, combinando chiarezza visiva, semplicità d'uso e architettura dati connessa a un backend.

STRUTTURA ATTUALE:

La pagina è suddivisa in due aree principali:

- **Sidebar dei filtri** (a sinistra): consente all'utente di selezionare i report da visualizzare attraverso due modalità:
    1. **Filtro per anno**, tramite menu a discesa.
    2. **Filtri numerici avanzati**, mediante slider che permettono di specificare soglie massime per:
        - Produzione totale (in tonnellate)
        - Fatturato netto (in milioni di euro)
        - Numero totale di dipendenti
        - Emissioni di CO_2 (in tonnellate)
        - Consumo di acqua totale (in m^3)

        Ogni slider aggiorna dinamicamente il valore selezionato tramite elementi <output> collegati via JavaScript.
- **Area centrale dei risultati**: inizialmente vuota, viene popolata dinamicamente con **schede (card)** una volta ricevuti i dati dal backend. Ogni card riporta:
    - l'**anno del report**;
    - un **titolo sintetico**;
    - un elenco di valori chiave (produzione, fatturato, emissioni, ecc.);
    - due pulsanti per **visualizzare** o **scaricare** il report.

L'interazione con il backend avviene tramite uno script JavaScript (fetch-all-reports.js) che esegue una fetch API al caricamento della pagina, recuperando i dati da un endpoint esposto in

NodeJs.
Sono inoltre presenti due **bottoni funzionali** per applicare i filtri selezionati o azzerarli con un reset.

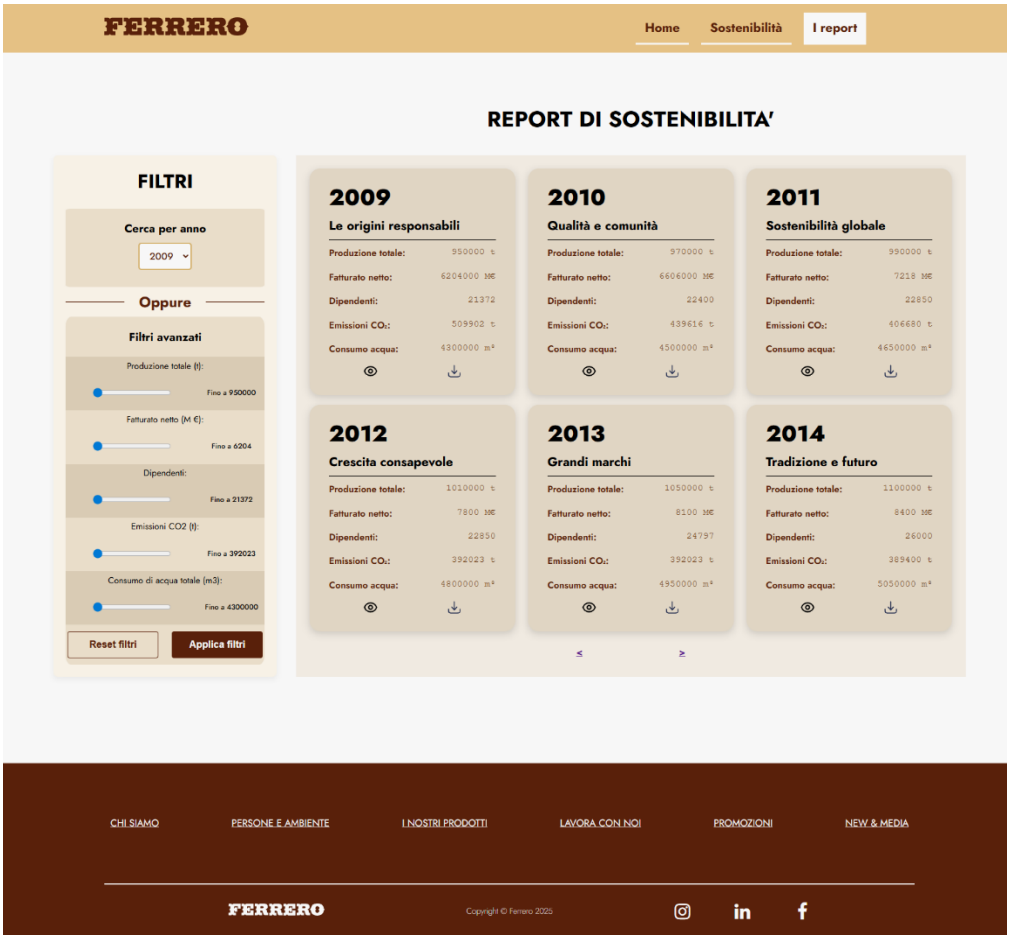
Aspetti grafici e stilistici

- L'interfaccia riprende la palette cromatica coerente con il brand Ferrero, con sfondi beige (#e0d5c3), bordi marroni e accenti scuri per titoli e footer.
- I **font** sono gestiti tramite Google Fonts (famiglia "Jost") e le icone social sono integrate via Font Awesome.
- La pagina è responsive e costruita tramite Flexbox, con particolare attenzione alla leggibilità dei valori nella sezione delle card.
- È prevista una sezione di paginazione (attualmente commentata) che sarà attivata quando il numero di report supererà una certa soglia.

Stato attuale

La struttura HTML è completa e già integrata con i principali script di interazione. I dati dei report sono recuperati dinamicamente tramite chiamate API, anche se attualmente la parte di rendering è ancora in fase di avanzamento. Ulteriori migliorie previste includono l'aggiunta della paginazione dinamica, un sistema di messaggi per la gestione di errori lato client (es. nessun risultato trovato), e un perfezionamento della responsività grafica per la fruizione da dispositivi mobili.

Tutte le funzionalità principali sono testabili in ambiente locale.



BACKEND (da completare)

Il backend del progetto è stato sviluppato utilizzando NodeJs in combinazione con il framework ExpressJs, secondo un'architettura modulare orientata alla separazione delle responsabilità. L'applicazione espone un set di API RESTful che restituiscono i dati in formato JSON al frontend, facilitando la comunicazione asincrona tra le due componenti.

Organizzazione delle cartelle

Il progetto è organizzato in cartelle distinte, ciascuna delle quali risponde a una specifica funzione logica:

- **/routes/**
  Contiene il file report-routes.js, all'interno del quale è definita la rotta principale GET /api/reports, attualmente incaricata di restituire tutti i report. La struttura è stata pensata per essere **facilmente estendibile**, in modo da ospitare in futuro ulteriori rotte relative a funzionalità avanzate, come ad esempio filtri dinamici o altre azioni correlate ai report.
- **/controllers/**
  Include report-controller.js, in cui è presente la funzione getAllInfo(), responsabile della logica applicativa che interroga il database e restituisce la risposta al client. Anche questa sezione è predisposta per accogliere nuove funzioni man mano che l'applicazione si arricchirà di ulteriori endpoint.
- **/utils/**
  Contiene db.js, che centralizza la logica di connessione al database MySQL. In particolare, include:
  - createConnection() per stabilire la connessione al database tramite il pacchetto mysql2/promise;
  - executeQuery() per eseguire query SQL in modo riutilizzabile e parametrico.
- **/reports/**
  È una cartella predisposta per ospitare file PDF o eventuali risorse collegate alla gestione diretta dei report.

Questa struttura riflette un approccio scalabile, nel quale ogni nuova funzionalità può essere facilmente integrata senza compromettere l'organizzazione esistente. Ad esempio, l'aggiunta di nuove entità logiche (es. utenti, categorie, statistiche) potrà essere gestita creando nuovi file separati per rotte, controller e servizi dedicati, mantenendo così chiarezza e modularità.

Endpoint attualmente implementati

Attualmente è presente un solo endpoint operativo:

```
GET /api/reports
```

Tale endpoint restituisce, in formato JSON, l'elenco completo dei report salvati nel database relazionale MySQL. L'API è pensata per essere consumata direttamente dal frontend tramite fetch asincrona e rappresenta il primo punto di contatto tra frontend e backend: questa API verrà chiamata al caricamento della pagina "reports.html".
L'output della risposta segue questa struttura:

```
{
  "data": [ ...array di oggetti report... ],
  "status": 200
}
```

L'architettura adottata, unita all'organizzazione modulare dei file, rende semplice l'aggiunta di futuri endpoint, sia in GET che in POST/PUT/DELETE, permettendo di espandere le funzionalità dell'applicazione in maniera ordinata e mantenibile.

Stato attuale e prospettive

Il backend è correttamente funzionante in ambiente locale previa configurazione del database MySQL con la tabella reports. Al momento l'accesso è limitato alla sola lettura (query SELECT), ma la struttura modulare adottata consente facilmente di estendere l'applicazione con metodi POST, PUT e DELETE in una fase successiva.

DATABASE E STRUTTURA DATI

Il database utilizzato all'interno del progetto è di tipo relazionale ed è gestito tramite MySQL.
È stato creato un database denominato "ferrero", all'interno del quale è presente attualmente un'unica tabella, chiamata "reports", progettata per raccogliere e organizzare i dati principali estratti dai report di sostenibilità pubblicati annualmente dall'azienda.
La tabella reports è composta da otto campi principali, di seguito lo screenshot della struttura:

```
mysql> describe reports;
+----------------------+--------------+------+-----+---------+----------------+
| Field                | Type         | Null | Key | Default | Extra          |
+----------------------+--------------+------+-----+---------+----------------+
| ID                   | int          | NO   | PRI | NULL    | auto_increment |
| anno                 | year         | YES  |     | NULL    |                |
| filename             | varchar(255) | NO   |     | NULL    |                |
| produzione_totale    | int          | YES  |     | NULL    |                |
| fatturato_netto      | int          | YES  |     | NULL    |                |
| dipendenti           | int          | YES  |     | NULL    |                |
| emissioni_co2        | int          | YES  |     | NULL    |                |
| consumo_acqua_totale | int          | YES  |     | NULL    |                |
| titolo               | varchar(255) | YES  |     | NULL    |                |
+----------------------+--------------+------+-----+---------+----------------+
```

I campi "anno" e "filename" sono obbligatori (NOT NULL), poiché rappresentano gli identificatori fondamentali per associare i dati a ciascun file PDF effettivo. Gli altri campi possono essere lasciati vuoti (NULL), in quanto non sempre tutte le informazioni sono reperibili con precisione nei documenti originali. Il campo "filename" è utilizzato dal frontend per ricostruire dinamicamente il link completo al file PDF associato al report, combinando il valore del campo con il percorso /public/ esposto dal backend.
Questo è reso possibile grazie all'utilizzo del middleware express.static() in ExpressJs, che rende disponibile la cartella reports/ come risorsa pubblica all'interno dell'URL /public:

```
app.use('/public', express.static('./reports/'));
```

Funzione della tabella
La tabella reports funge da ponte strutturato tra i dati statici (PDF dei report) e la loro rappresentazione dinamica nella pagina “reports.html”. Ogni riga della tabella rappresenta un report distinto, e i valori numerici vengono utilizzati sia per la visualizzazione nelle card frontend, sia per l’applicazione dei filtri dinamici (es. per produzione, CO_2, acqua, ecc.).
L’interazione con la tabella avviene attualmente solo in lettura (query SELECT) tramite l’endpoint GET /api/reports implementato in NodeJs ma la struttura è già predisposta per eventuali operazioni di scrittura o aggiornamento.

CONCLUSIONE <parziale>

L’implementazione del progetto è attualmente in fase avanzata: le tecnologie principali sono state definite e integrate, e le componenti base del sito sono già state predisposte. Le prossime fasi riguarderanno l’affinamento dell’interfaccia utente, l’ottimizzazione dei filtri nella pagina “reports.html” e l’inserimento definitivo dei dati provenienti dai report Ferrero nel database. Ulteriori dettagli progettuali saranno descritti nella versione finale del project work.

DOMANDE PER IL DOCENTE:

1. È consigliabile creare un nuovo file partendo da zero, mantenendo comunque la struttura prevista dal template, al fine di ottenere una maggiore libertà nella personalizzazione grafica dell’impaginato?
2. L’architettura progettuale adottata (frontend-backend separati, architettura detached, comunicazione via API, gestione locale dei dati) e l’impostazione generale del progetto risultano adeguate rispetto agli obiettivi previsti dalla traccia? Oppure consiglia di apportare modifiche strutturali o metodologiche?
3. La sintassi, i tempi verbali e il livello di formalità utilizzati nelle sezioni già redatte dell’elaborato risultano coerenti e adeguati a un elaborato accademico? In caso contrario, suggerisce correzioni specifiche o un diverso orientamento stilistico?
4. Il livello di dettaglio attuale nella “Descrizione dei principali aspetti progettuali” è adeguato o dovrebbe essere ulteriormente ampliato o ridotto? Ci sono sezioni su cui suggerisce di aggiungere chiarimenti, esempi di codice o approfondimenti tecnici?

Rimango a disposizione per eventuali chiarimenti o integrazioni. Grazie in anticipo per il tempo e l’attenzione dedicati alla valutazione del lavoro svolto fino a questo punto.

Campi di applicazione
(Descrivere gli ambiti di applicazione dell’elaborato progettuale e i vantaggi derivanti della sua applicazione):

Inserisci qui il testo

Valutazione dei risultati
(Descrivere le potenzialità e i limiti ai quali i risultati dell’elaborato sono potenzialmente esposti)

Inserisci qui il testo