

Assistente nutrizionale ecosostenibile

AeSW A.A.19/20

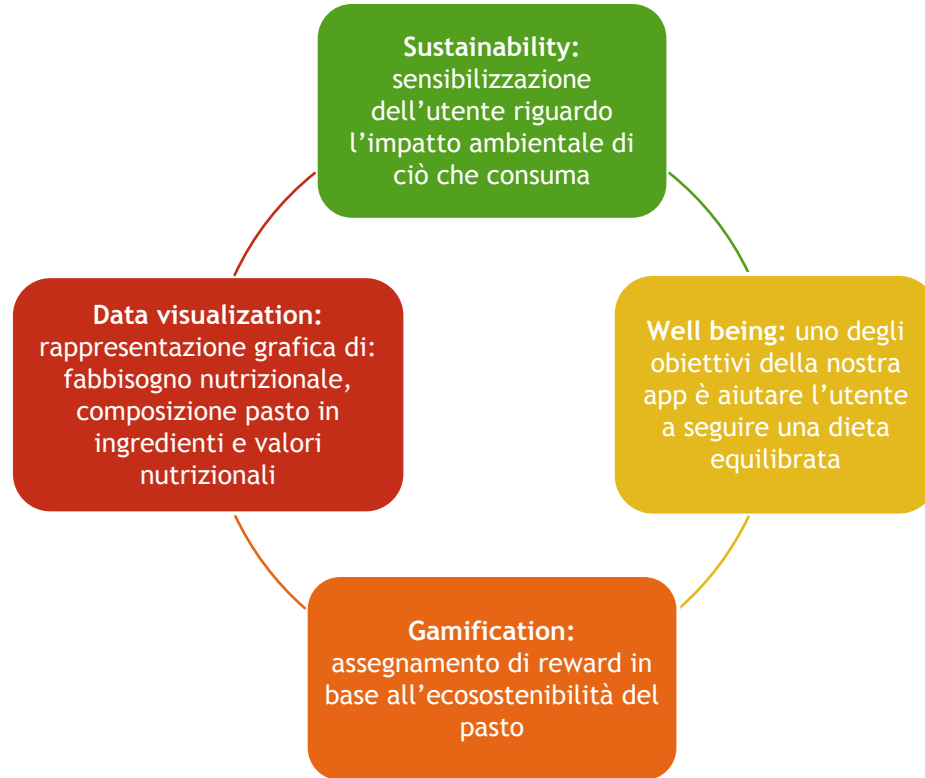
Idea

Assistente nutrizionale ecosostenibile nasce con lo scopo di aiutare chi voglia mantenere una dieta equilibrata nel rispetto dell'ambiente.

Si fornisce all'applicazione l'elenco di prodotti che si intende mangiare in un pasto, scannerizzando il barcode presente sulla confezione e fornendo la quantità che si vuole assumere. Viene calcolato l'apporto del pasto completo e i suoi valori nutrizionali vengono comparati con i fabbisogni giornalieri raccomandati dalla World Health Organization (WHO).

L'impatto ambientale viene rappresentato in termini di CO2 e acqua impiegati per la produzione del pasto in base alla composizione e alla quantità di ogni ingrediente.

Temi trattati e intersezioni



Pasti



Inserimento pasto



Ricerca ingrediente



Grafici composizione pasto e componenti nutrizionali



Grafico fabbisogno giornaliero secondo WHO



Grafico impatto ambientale

Utenti

Per attirare l'interesse degli utenti fin dal primo approccio con l'applicazione ci siamo concentrati sulla presentazione della home page cercando di renderla accattivante ed esplicativa dell'obiettivo della web app.

Per rendere il processo di registrazione spontaneo piuttosto che obbligatorio abbiamo voluto fornire funzionalità anche a **utenti non registrati**. Infatti chiunque può ricercare informazioni riguardanti un prodotto tramite barcode.

L'utente registrato può accedere alle seguenti funzionalità:

- Creazione/Rimozione di **pasti** in una data
- Aggiunta di un **ingrediente** ad un pasto tramite ricerca del **barcode** e rimozione
- Visualizzazione di **grafici** relativi alla composizione del pasto in termini di ingredienti e di macro componenti
- Visualizzazione grafici **impatto ambientale**
- Visualizzazione grafici rispetto **fabbisogno nutrizionale**

Data visualization

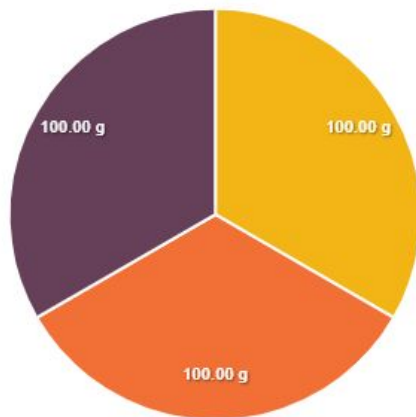
- **Fabbisogno nutrizionale giornaliero:** il seguente grafico riporta i valori nutrizionali dei pasti assunti durante una giornata e li confronta con quelli del proprio fabbisogno nutrizionale giornaliero secondo la WHO.



Data visualization

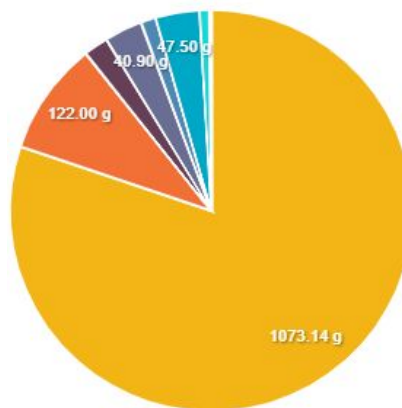
Composizione pasto

► Ingredienti



● torta sbrisolona - 100 g
● Tonno all'olio di oliva - 100 g
● Vollkorn Spaghetti - 100 g

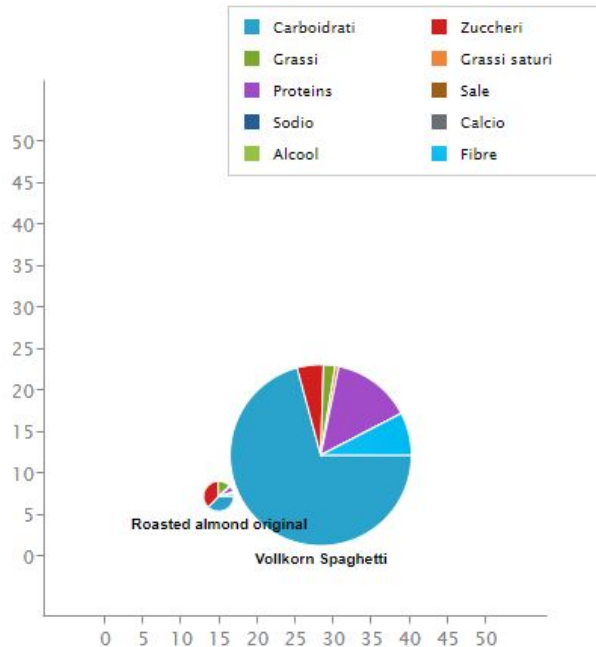
► Percentuale valori nutrizionali



● energy_kcal_tot
● carbohydrates_tot
● sugars_tot
● fat_tot
● saturated_fat_tot
● proteins_tot
● fiber_tot
● salt_tot
● sodium_tot
● alcohol_tot

Data visualization

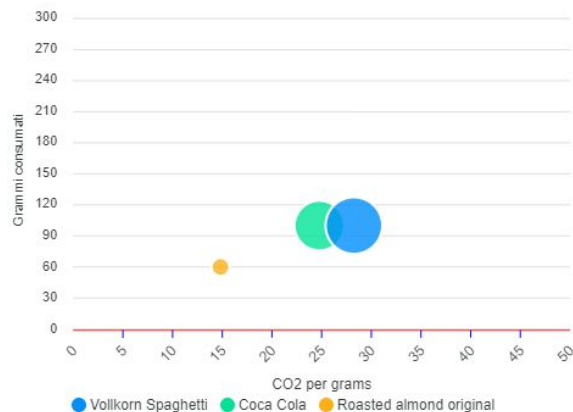
- Ingredienti del pasto e composizione in valori nutrizionali di ciascuno di essi



Data visualization

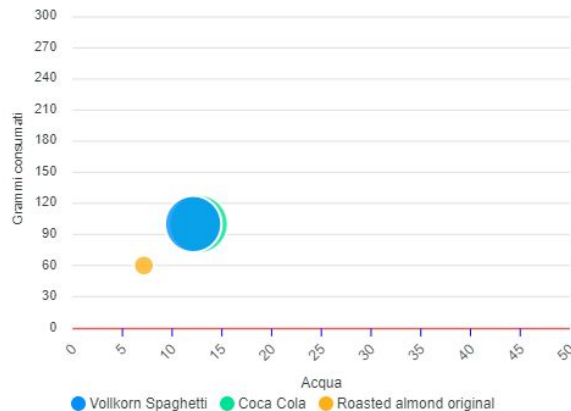
● Emissioni CO2

Consumo CO2



● Consumo acqua

Consumo d'acqua



Gamification

Per coinvolgere maggiormente l'utente abbiamo pensato di introdurre degli **obiettivi**.

Questa funzionalità è mirata solo agli utenti che si registrano, infatti il primo obiettivo è "Prima registrazione".

Attualmente l'applicazione prevede obiettivi focalizzati sui pasti green, cioè per quei pasti che hanno un ridotto impatto ambientale. Un esempio è il "Pasto ecologico" che viene assegnato quando un pasto con un ridotto consumo di co2 viene completato.

In questo modo gli utenti vengono invogliati ad evitare o comunque ridurre il consumo di cibi con alti valori di co2 e di h2o.

Grazie a questi obiettivi si aumenta la sensibilizzazione sul tema ambientale e si aiuta gli utenti a capire che piccoli accorgimenti sulla scelta degli ingredienti e ciò che mangiamo può fare una notevole differenza.

The background features a series of overlapping, semi-transparent green triangles and polygons that create a dynamic, layered effect. The colors range from a light, pale green to a deep, forest green. The shapes are primarily located on the right side of the image, with some extending towards the center. The overall composition is modern and minimalist.

UX design

Personas

Per improntare lo sviluppo della UX sono state definite 2 personas, che incarnano il target di riferimento della nostra applicazione:

- Martina è una ragazza lavoratrice che ha molto a cuore l'ambiente e si sta approcciando a uno stile di vita ecosostenibile. Inoltre, sta iniziando a stare molto attenta alla linea e a rispettare il valori nutrizionali giornalieri da assumere per condurre uno stile di vita sano. Tuttavia a causa degli impegni lavorativi non ha molto tempo a disposizione e cerca una linea guida che l'aiuti in questo percorso.
- Giuseppe è un signore di 75 anni che in seguito a problemi cardiaci si vuole attenere a una dieta sana ed equilibrata e vuole monitorare i grassi che assume durante il giorno.

Scenarios

Per raggiungere i propri obiettivi, le personas dovranno compiere una serie di task

- Martina vuole tenere sotto controllo le emissioni dei suoi pasti.

Tasks:

- Analizzare l'impatto di singoli prodotti
- Visualizzare le emissioni complessive dei suoi pasti

- Giuseppe vuole tenere sotto controllo i grassi assunti durante il giorno.

Tasks:

- Tenere un diario dei pasti
- Controllo fabbisogno nutrizionale giornaliero

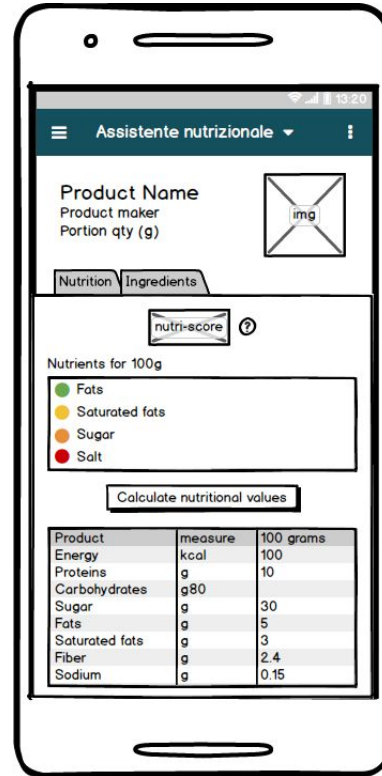
Storyboard

1. **Aggiunta pasto** tramite inserimento nome pasto e click (invio da tastiera) sul bottone “+”.
2. **Ricerca ingrediente:** si parte dal bottone “add component” e si viene rimandati alla schermata di scan barcode. A scannerizzazione completa l’utente viene rimandato alla pagina “Products informations”



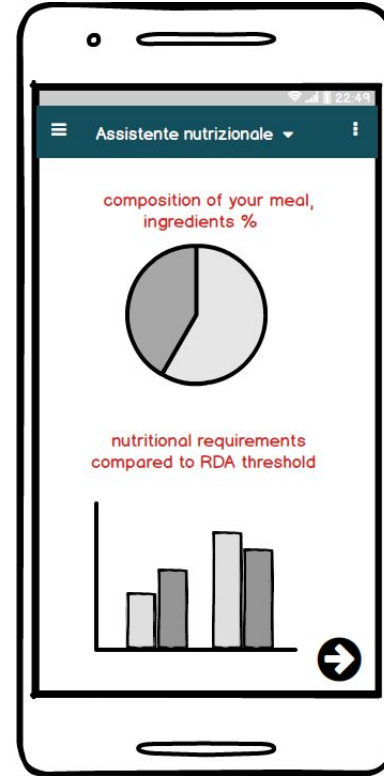
Storyboard

3. **Verifica ingrediente:** Una volta selezionato un prodotto, l'utente si trova nella schermata "Product informations" dove può vedere tutte le informazioni relative all'ingrediente che vuole aggiungere al pasto. L'aggiunta al pasto avviene in seguito alla pressione di "calculate nutritional values", il quale computa i valori per la quantità indicata.



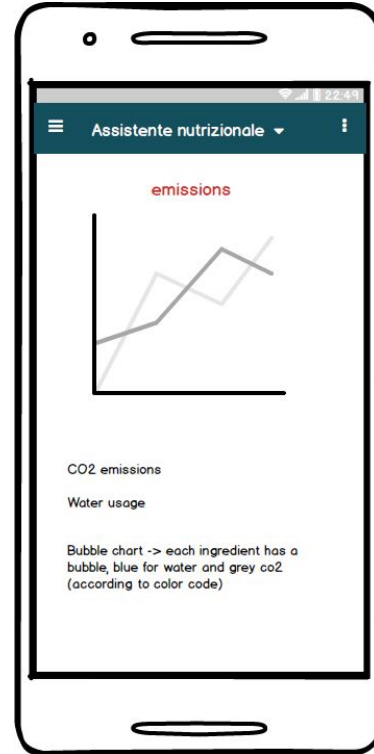
Storyboard

4. **Visualizzazione grafico fabbisogno nutrizionale:** vi si accede dall'elenco pasti tramite click sul bottone “go to charts” con icona pie chart.



Scenario: Storyboard

4. **Visualizzazione grafico emissioni** La pagina dei grafici comprende diverse schermate, il grafico delle emissioni si trova nell'ultima pagina, raggiungibile tramite click sul bottone freccia destra.



Participatory scenario

Durante tutto il processo di sviluppo, dal design al testing, di sono stati coinvolti alcuni utenti finali dell'applicazione in modo da venire incontro alle loro esigenze, rendendoli partecipi e con un ruolo attivo.

L'interazione con gli utenti è stata necessaria a scopo di migliorare il design, le feature e rendere l'applicazione user-friendly.

Ad esempio alcuni utenti, posti di fronte agli scenari precedenti, ci hanno suggerito le seguenti modifiche:

- la modifica della quantità del prodotto che si vuole assumere in fase di inserimento potendo vedere i valori nutrizionali relativi alla quantità data
- poter rivedere le tabelle nutrizionali degli ingredienti del pasto senza doverli ricercare nuovamente tramite barcode
- modificare la sezione grafici aggiungendo delle tab

Nella sezione dedicata al frontend vedremo nel dettaglio il flusso di modifiche dai mockup iniziali all'applicazione finale.

The background features a series of overlapping, semi-transparent green triangles and polygons that create a dynamic, geometric pattern. The colors range from a light, pale green to a deep, forest green. The shapes are layered, with some appearing to be in front of others, creating a sense of depth and movement. The overall effect is modern and tech-oriented.

Tecnologie

Stack MEVN

Come solution stack per lo sviluppo dell'applicazione abbiamo scelto MEVN, framework di tipo full-stack Javascript che permette la realizzazione di Single Page Applications (SPA).

I principali vantaggi dell'utilizzo sono quelli che presentano gli stack più recenti:

- ▶ Utilizzo dello stesso linguaggio di programmazione per la programmazione frontend e backend
- ▶ JSON come unico formato per la rappresentazione e lo scambio dei dati
- ▶ Scalabilità di Node.js e MongoDB



MongoDB

- Database non relazionale orientato ai documenti con schema dinamici. Classificato come database NoSQL.
- **Mongoose** è una libreria di tipo Object Data Modeling (ODM) per MongoDB e Node.js. Gestisce le relazioni tra i dati, fornisce la convalida dello schema e viene utilizzato per tradurre oggetti nel codice nella loro rappresentazione in MongoDB.



mongoose

Express.js

È il web framework per Node.js più popolare, nella nostra applicazione web è stato utilizzato per:

- ▶ Gestione delle chiamate HTTP su diverse route
- ▶ Configurazione di impostazioni comuni per l'applicazione web da utilizzare per la connessione
- ▶ Definizione di Model e Controller per la gestione della logica di backend

express

Node.js

È un ambiente runtime Javascript che ci ha permesso di sviluppare le seguenti funzionalità lato server:

- ▶ Gestione richieste HTTP tramite il framework Express
- ▶ Salvataggio informazioni relative a: pasti, utenti e prodotti.
- ▶ Logica di calcolo valori nutrizionali e fabbisogno nutrizionale
- ▶ Logica di gamification
- ▶ Gestione sicurezza della password utente (hashing + salt)



Abbiamo utilizzato **Nodemon** per agevolare lo sviluppo del backend in quanto permette il riavvio automatico dell'applicazione ad ogni modifica su file rilevata.

Vue.js

Vue.js framework recente, leggero e allo stesso tempo molto versatile. La sua semplicità porta ad una buona curva di apprendimento. Vue.js è ottimo per sviluppo rapido di prototipi, compiuto tramite l'utilizzo di Vue CLI, che ha facilitato il set up del frontend dell'applicazione e lo sviluppo. Per la gestione dello stato centralizzato è stata utilizzata la libreria **Vuex**.



Per la composizione del frontend si è deciso di utilizzare la libreria **Bootstrap-vue**.



Vue I18n

Plugin di internazionalizzazione per Vue, utilizzato per fornire l'applicazione in più lingue.



Per la personalizzazione dello stile si è optato per Sass. Il suo utilizzo, ci ha permesso l'impiego di variabili, che hanno semplificato il processo di sviluppo di **temi** per la nostra applicazione. Per una impostazione corretta del file .sass è stato utilizzato **sass-lint**.

Librerie e APIs

Open Food Facts è un database di prodotti alimentari realizzato da tutti, per tutti.

È un progetto senza scopo di lucro sviluppato da migliaia di volontari provenienti da tutto il mondo.

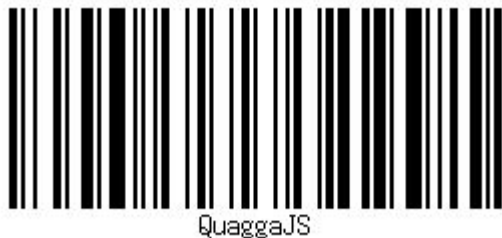
Contiene informazioni sui valori nutrizionali di oltre 1 milione di prodotti, ricercabili tramite codice a barre.

Offre anche delle **RESTful APIs** per essere interrogato.



QuaggaJS è uno scanner di codici a barre interamente scritto in JavaScript che supporta la localizzazione e la decodifica in tempo reale di vari tipi di codici a barre.

La libreria è anche in grado di utilizzare **getUserMedia** per ottenere l'accesso diretto allo stream della fotocamera dell'utente.



Charts libraries

- **Apexcharts**

libreria utilizzata per la creazione dei seguenti grafici:

- Bar chart con valori negativi e positivi per lo sforamento del fabbisogno giornaliero
- Pie chart ingredienti pasto
- Pie chart componenti pasto



- **Zingchart**

questa libreria è stata utilizzata nella creazione del grafico Bubble pie chart, riguardante la composizione del pasto in ingredienti (bubble) ciascuno di essi contenente un pie chart con i componenti.



Development



Gitkraken: client git con interfaccia grafica grazie alla quale siamo riusciti a lavorare simultaneamente su più branch ed essere aggiornati sui progressi dei vari componenti del gruppo e della branch di development.



Trello: tool utilizzato per organizzare il lavoro settimanale dei vari componenti del gruppo.



Eslint: tool che permette di rendere il codice più consistente e pulito, riconoscendo in anticipo potenziali comportamenti anomali.



Postman: è stato utilizzato per creare test automatizzati per le nostre API condivisi tramite un gruppo di lavoro.

Deployment



Heroku è un servizio di platform as a service (PaaS) che consente agli sviluppatori di creare, eseguire e gestire applicazioni interamente nel cloud. Lo abbiamo utilizzato per l'hosting dell'applicazione tramite un processo di deploy automatico



Mlab hosting di database MongoDB



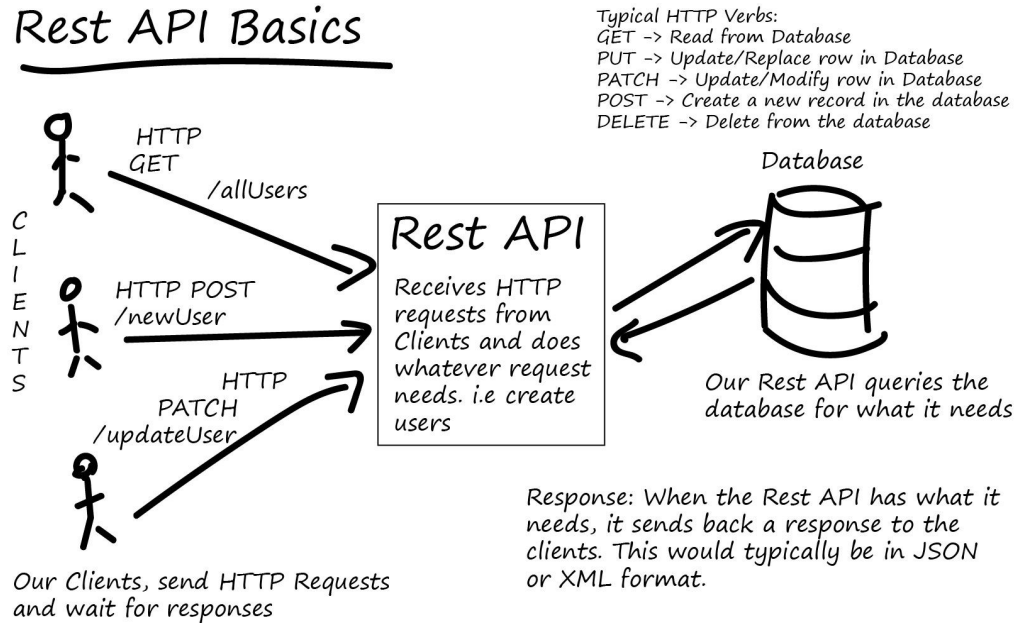
Github utilizzato come repository online sia per la gestione del lavoro in fase di sviluppo sia per il deploy, dato che heroku è stato collegato ad un branch del repository

The background features a series of overlapping, semi-transparent green triangles and polygons that create a dynamic, geometric pattern. The colors range from a light lime green to a darker forest green. The pattern is concentrated on the right side of the image, with the left side being mostly white.




Sviluppo

Backend: Restful API

Rest API Basics



Backend: APIs

▼  Meals 6 requests		▼  Products 3 requests		▼  Users 3 requests	
GET	LoadMealsList	POST	InsertProduct	GET	LoadUser
GET	LoadMeal	GET	LoadProduct	POST	InsertUser
POST	NewMeal	GET	LoadProductQuantity	PUT	UpdateUser
PUT	NewComponent				
DEL	DeleteMeal				
DEL	DeleteComponent				

Backend: Mongoose schema

- WHO

```
const WhoSchema = new Schema({
  sex: String,
  age_min: Number,
  age_max: Number,
  calories: Number,
  protein_g: Number,
  carbohydrate_g: Number,
  fiber_g: Number,
  sugars_perc: Number,
  total_fat_perc: Number,
  saturated_fat_perc: Number,
  calcium_g: Number,
  sodium: Number,
});
```

- User

```
const UserSchema = new Schema({
  username: { type: String, required: true },
  password_hash_salt: { type: String, required: true },
  salt: { type: String, required: true },
  name: { type: String, required: true },
  surname: { type: String, required: true },
  birth_date: { type: Date, required: true },
  email: { type: String, required: true },
  sex: String,
  user_img_url: String,
  weight: Number,
  height: Number,
  allergens: Array,
  daily_requirement: WhoSchema,
  achievements: [],
});
```


Backend: Mongoose schema

- Products

```
const ProductSchema = new Schema({  
  code: Number,  
  product_name: String,  
  image_url: String,  
  quantity: Number,  
  brands: String,  
  ingredients_text: String,  
  traces: String,  
  origin: String, // new  
  packaging: String, // new  
  serving_size: Number,  
  allergens: String,  
  energy_kj_100g: Number,  
  energy_kcal_100g: Number, // new  
  carbohydrates_100g: Number,  
  sugars_100g: Number,  
  fat_100g: Number,  
  saturated_fat_100g: Number,  
  proteins_100g: Number,  
  fiber_100g: Number,  
  salt_100g: Number,  
  sodium_100g: Number,  
  alcohol_100g: Number,  
  calcium_100g: Number,  
  nutrition_score_uk_100g: String,  
  nova_group: String, // new  
  carbon_footprint_100g: Number,  
  water_footprint_100g: Number,  
  measure_unit: String, // new  
});
```

Backend: Mongoose schema

- UserMeal

```
const UserMealSchema = new Schema({  
  username: String,  
  meals: [MealSchema],  
});
```

```
const MealSchema = new Schema({  
  meal_name: String,  
  components: [MealComponentSchema],  
  energy_kj_tot: Number,  
  energy_kcal_tot: Number,  
  carbohydrates_tot: Number,  
  sugars_tot: Number,  
  fat_tot: Number,  
  saturated_fat_tot: Number,  
  proteins_tot: Number,  
  fiber_tot: Number,  
  salt_tot: Number,  
  sodium_tot: Number,  
  alcohol_tot: Number,  
  calcium_tot: Number,  
  carbon_footprint_tot: Number,  
  water_footprint_tot: Number,  
  timestamp: Date,  
  is_closed: Boolean,  
}, { _id: false });
```

```
const MealComponentSchema = new Schema({  
  barcode: Number,  
  product_name: String,  
  image_url: String,  
  energy_kj: Number,  
  energy_kcal: Number,  
  quantity: Number,  
  nutrition_score: String,  
  carbon_footprint: Number,  
  water_footprint: Number,  
  measure_unit: String,  
}, { _id: false });
```

Backend: Sicurezza

Un aspetto al quale abbiamo dato molta importanza è la sicurezza.

L'applicazione usa **HTTPS** per garantire sicurezza nella comunicazione client-server, così che le credenziali non siano soggette a eavesdropping.

Inoltre la password non viene salvata sul database; al suo posto viene creato un hash della password e un valore di sale.

Per calcolare l'hash è stato usato **crypto**, un modulo di node specifico per la sicurezza.

Vengono eseguite 420 iterazioni e la lunghezza dell'hash è di 512 byte.

Questo garantisce che in caso di furto di dati le password degli utenti non siano leggibili.

Inoltre, l'adozione di HTTPS è indispensabile per l'utilizzo dell'API del browser **getUserMedia** poichè l'accesso allo stream della fotocamera viene concesso solo ad origini "trusted".

Frontend

- **Mobile first:** l'applicazione è stata pensata per un utilizzo mobile in quanto, tipicamente, durante i pasti gli utenti non hanno il computer a portata di mano.
- **Esperienza nativa:** Offerta dall'integrazione con la fotocamera tramite media capabilities API e web workers, impiegati per la scansione e riconoscimento dei barcode
- **Responsive design:** adatto a qualsiasi risoluzione mobile e ridimensionabile in versione Desktop

Frontend: Accessibilità

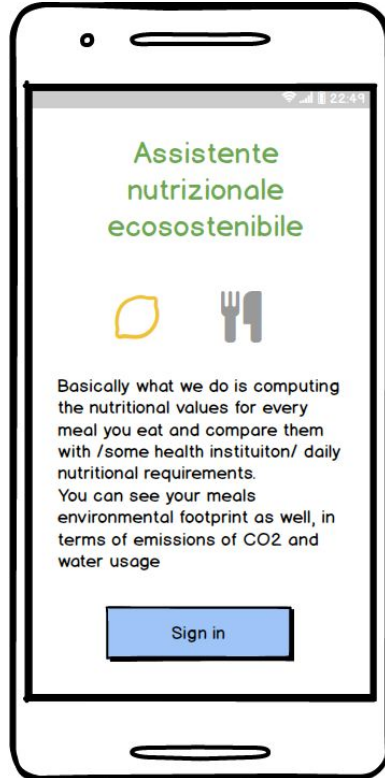
- **Localizzazione:** per rendere la nostra app accessibile anche a utenti non italiani abbiamo deciso di svilupparla in due lingue: italiano e inglese. Grazie all'adozione del plugin i18n è comunque estremamente facile produrre altre traduzioni ed adottarle.
- **Tema:**
 - **chiaro:** abbiamo scelto il verde come colore principale in quanto è associato a natura, ambiente ed ecologia.
 - **scuro:** è stato introdotto per agevolare la vista degli utenti nelle ore notturne.

Frontend: Accessibilità

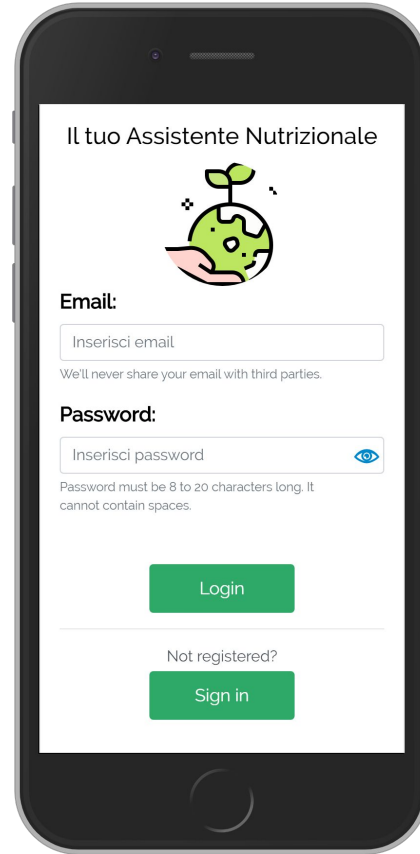
- **Google lighthouse:** è stato utilizzato per verificare il livello di accessibilità, le performance e le prestazioni della web app



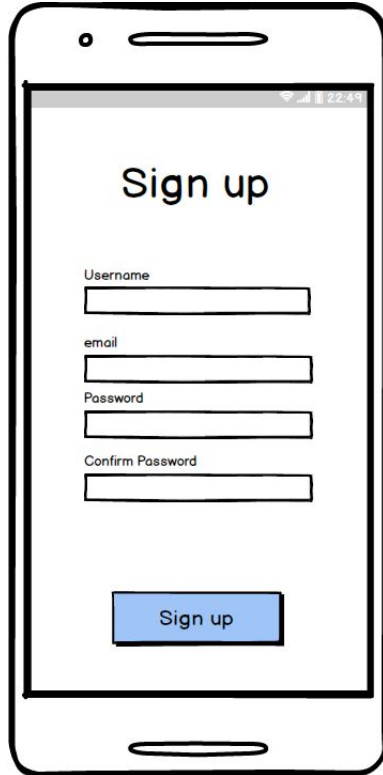
Frontend: Home Page



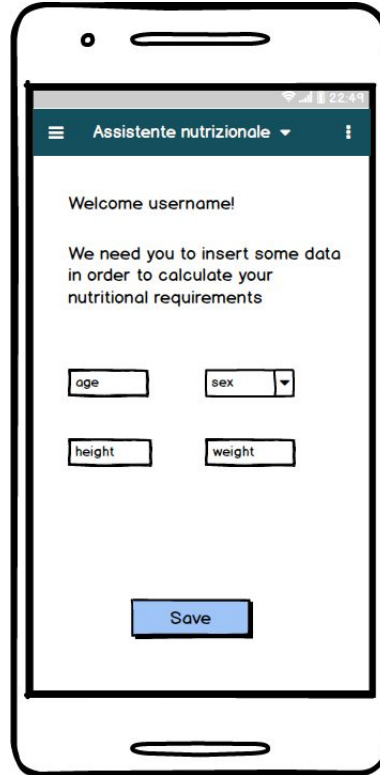
Frontend: Login



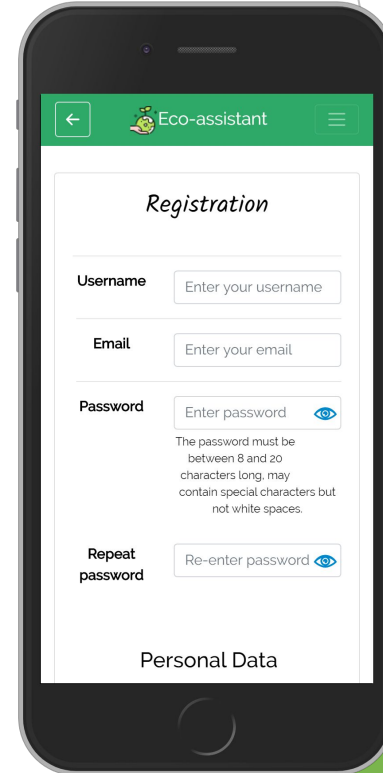
Frontend: Registrazione



A wireframe of a mobile app sign-up screen. It features a title "Sign up" at the top. Below the title are four input fields: "Username", "email", "Password", and "Confirm Password". At the bottom is a blue button labeled "Sign up".

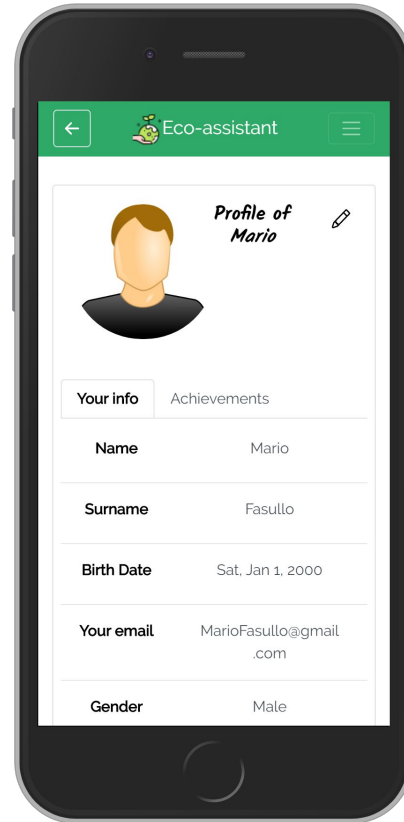
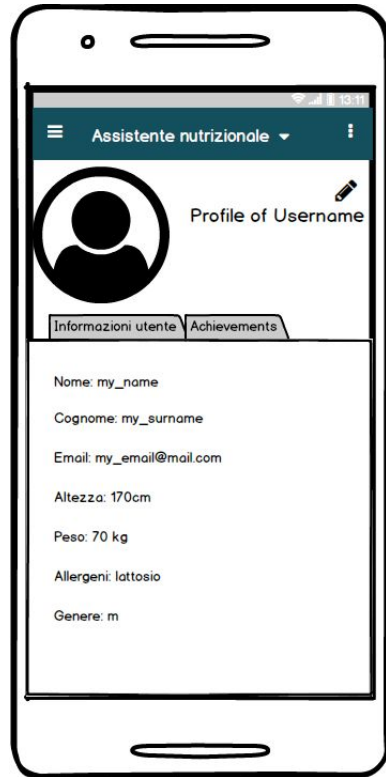


A wireframe of a mobile app registration screen. It features a title "Assistente nutrizionale" at the top. Below the title is a welcome message: "Welcome username!". Below this is a text prompt: "We need you to insert some data in order to calculate your nutritional requirements". Below the prompt are four input fields: "age", "sex" (with a dropdown arrow), "height", and "weight". At the bottom is a blue button labeled "Save".

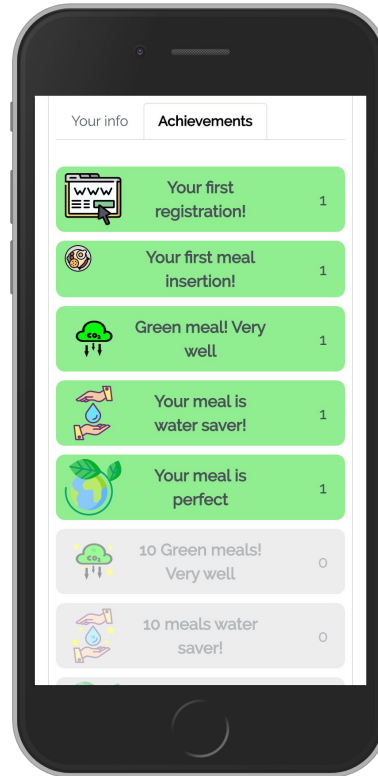


The final design of a mobile app registration screen. It features a title "Registration" at the top. Below the title are four input fields: "Username", "Email", "Password", and "Repeat password". The "Password" field has a hint: "The password must be between 8 and 20 characters long, may contain special characters but not white spaces." The "Repeat password" field has a hint: "Re-enter password". At the bottom is a section labeled "Personal Data".

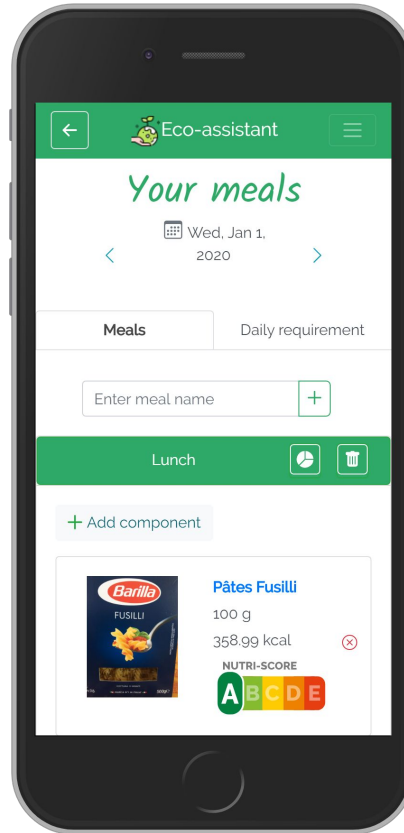
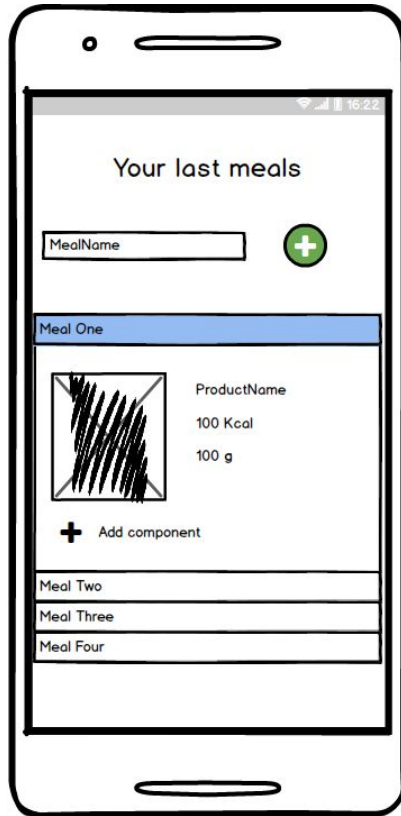
Frontend: Profilo 1



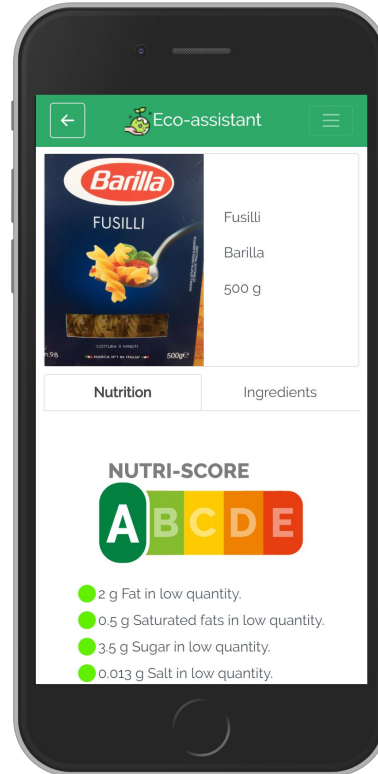
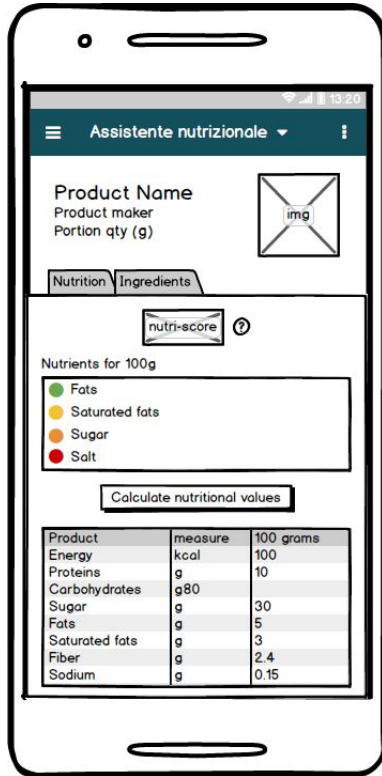
Frontend: Profilo 2



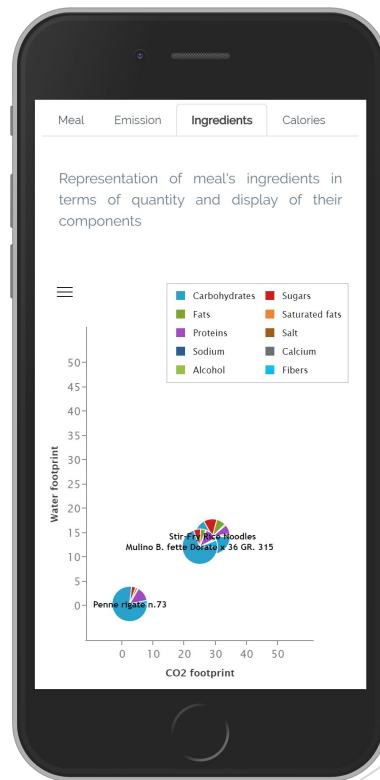
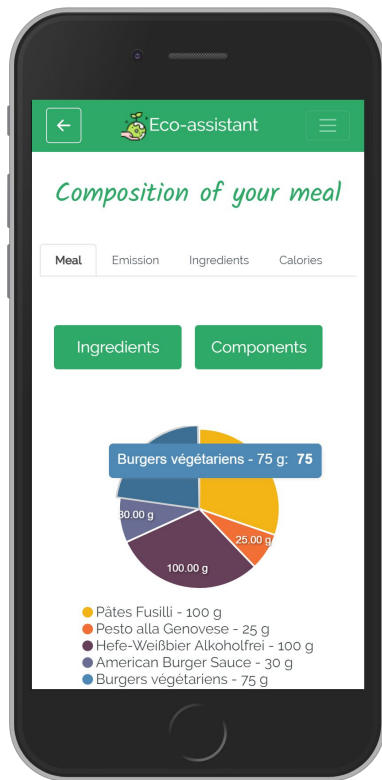
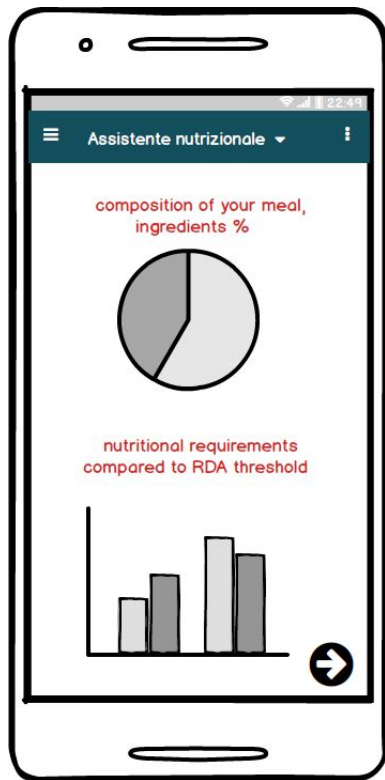
Frontend: I tuoi Pasti



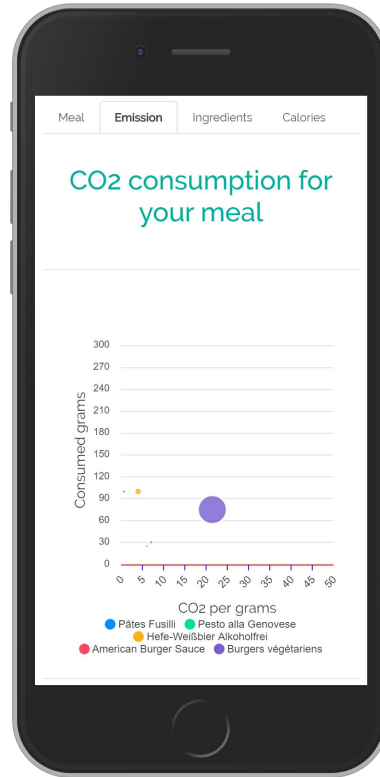
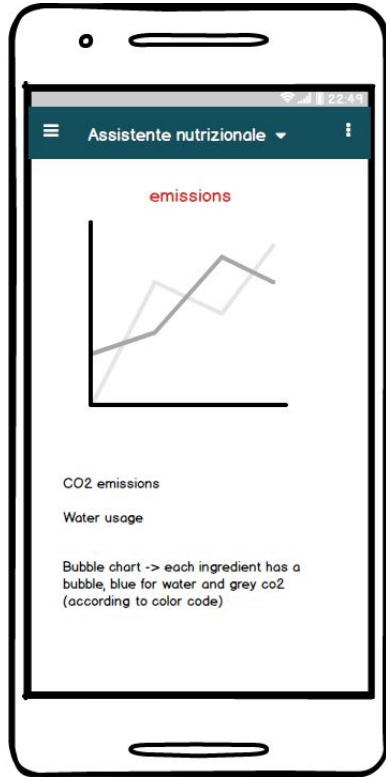
Frontend: Product informations



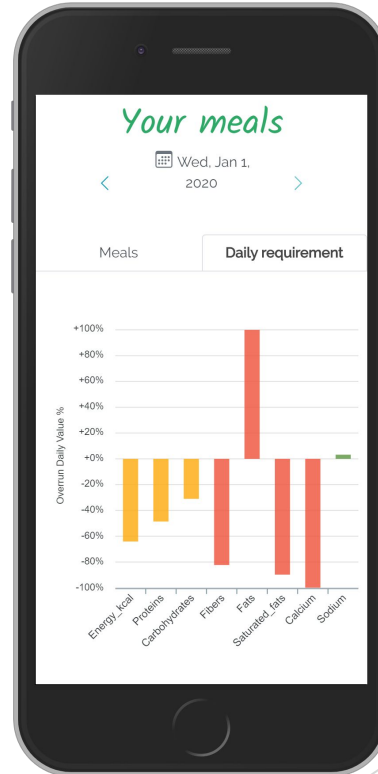
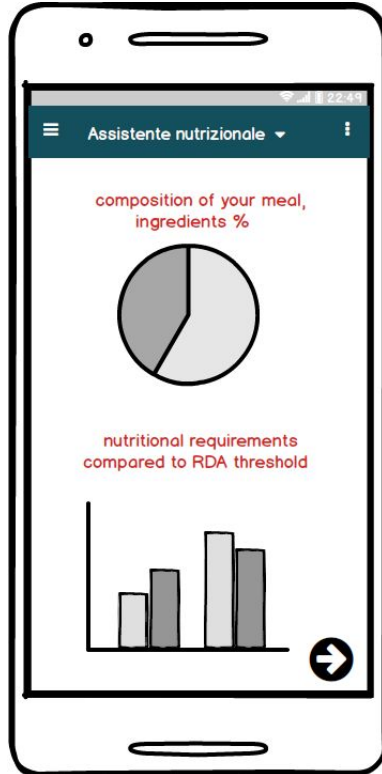
Frontend: Composizione pasti



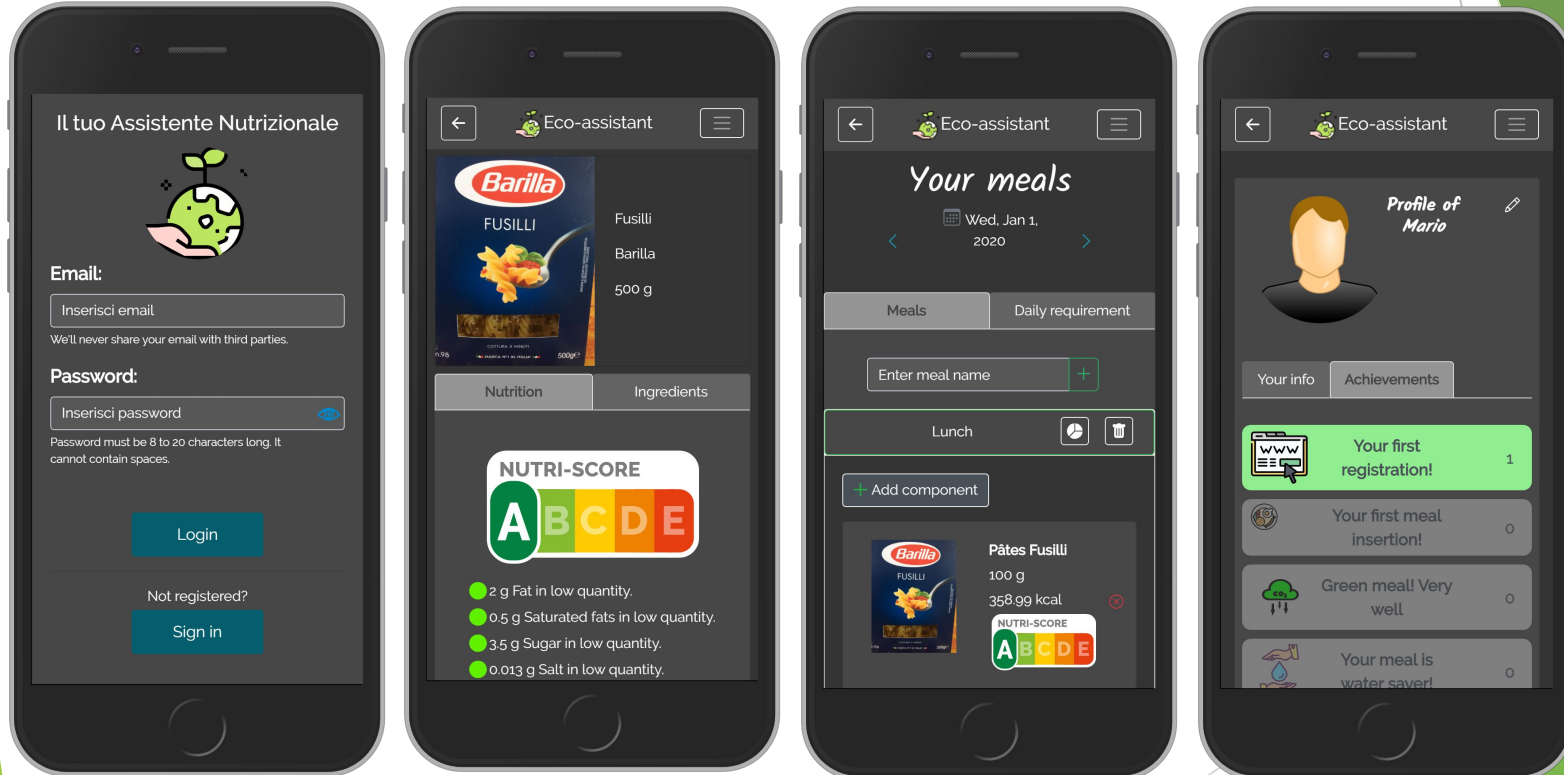
Frontend: Emissioni



Frontend: Fabbisogno nutrizionale



Frontend: Tema scuro



Usability testing

Feedback utenti

Sono stati interrogati gli utenti dopo l'utilizzo dell'applicazione web nella sua prima versione e sono stati riscontrati alcuni problemi:

- Mancanza di un tutorial sull'utilizzo
- Calendario difficile da utilizzare versione mobile
- Mancanza di controlli nei bottoni di rimozione

Con l'ultima versione dell'applicazione sono state colmate le mancanze sopra citate e gli utenti sono rimasti soddisfatti durante l'utilizzo.

Tuttavia alcuni consigliano l'inserimento di tutorial guidati il poter memorizzare ingredienti o ricette frequenti.



Sviluppi
futuri

Sviluppi futuri

Possibili sviluppi futuri:

- Scan della tabella nutrizionale tramite camera OCR
- Memorizzazione ricette
- Registrazione tramite social network (Google, Facebook)
- Tutorial di utilizzo dell'applicazione

Sviluppi futuri

- Miglioramento performance



Performance



Metrics

▲ First Contentful Paint	4.6 s	▲ First Meaningful Paint	4.6 s
■ Speed Index	4.6 s	■ First CPU Idle	4.6 s
■ Time to Interactive	4.6 s	▲ Max Potential First Input Delay	1,400 ms