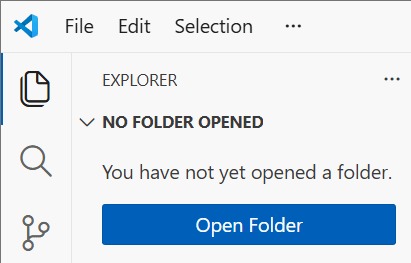
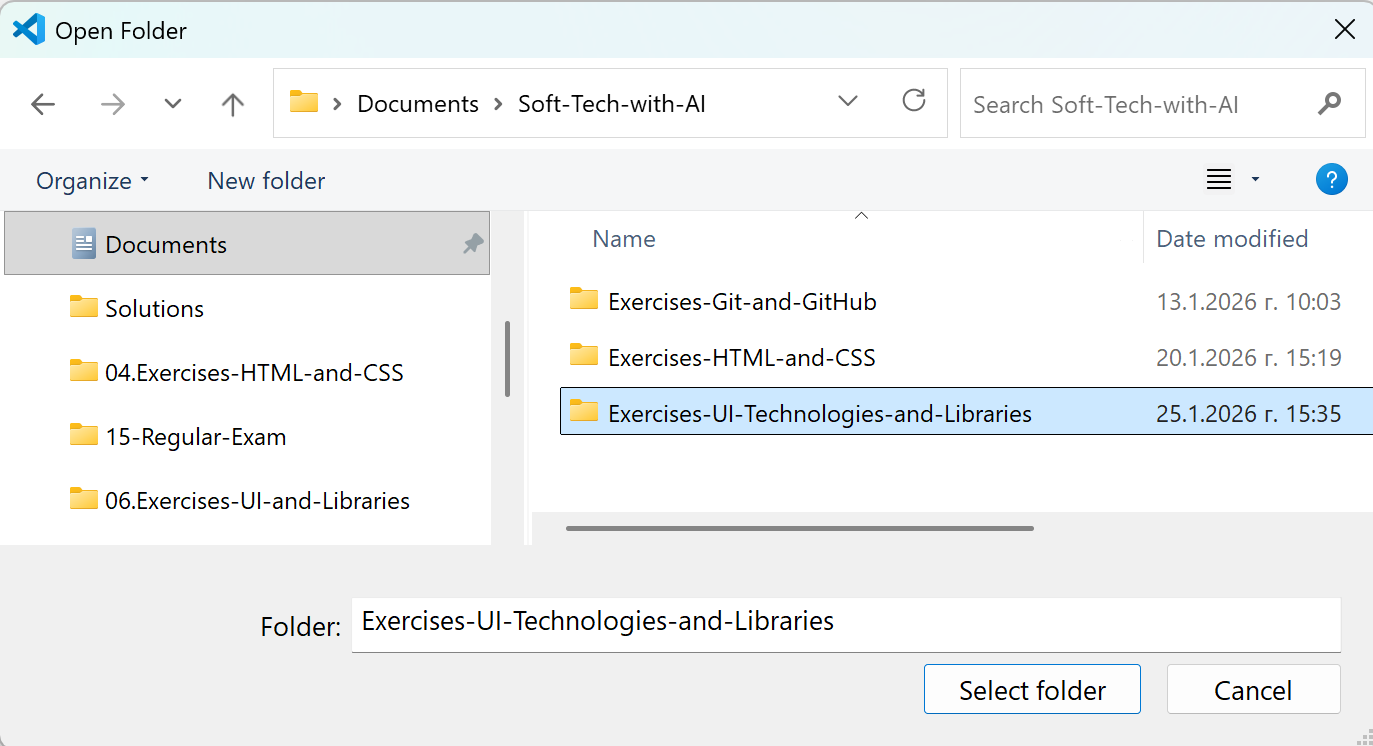
# UI Technologies and Libraries – Exercises

**Exercises** and **homework** assignments for “Software Technologies with AI” course at SoftUni AI.

## Create a Visual Studio Code Project

### Create a New Project

Start **VS Code**. **Create** and **open a folder**, which will hold your exercises about HTML and CSS. Give it a meaningful name, e.g. Exercises-UI-Technologies-and-Libraries.

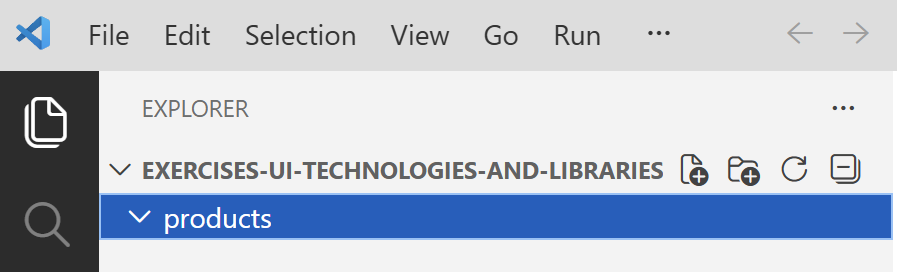
Now you have an empty VS Code project.



## Product Card with Bootstrap

Your task is to **create** **an info card** about a certain **product**, e.g. a smartphone "Samsung Galaxy S25 Ultra".

* Create a separate folder "products" for this exercise.
* Create an info card for Samsung Galaxy S25 Ultra: product-card.html
* Use **HTML** and **Bootstrap**, **no custom CSS**
* Import Bootstrap via CDN



### The AI Prompt

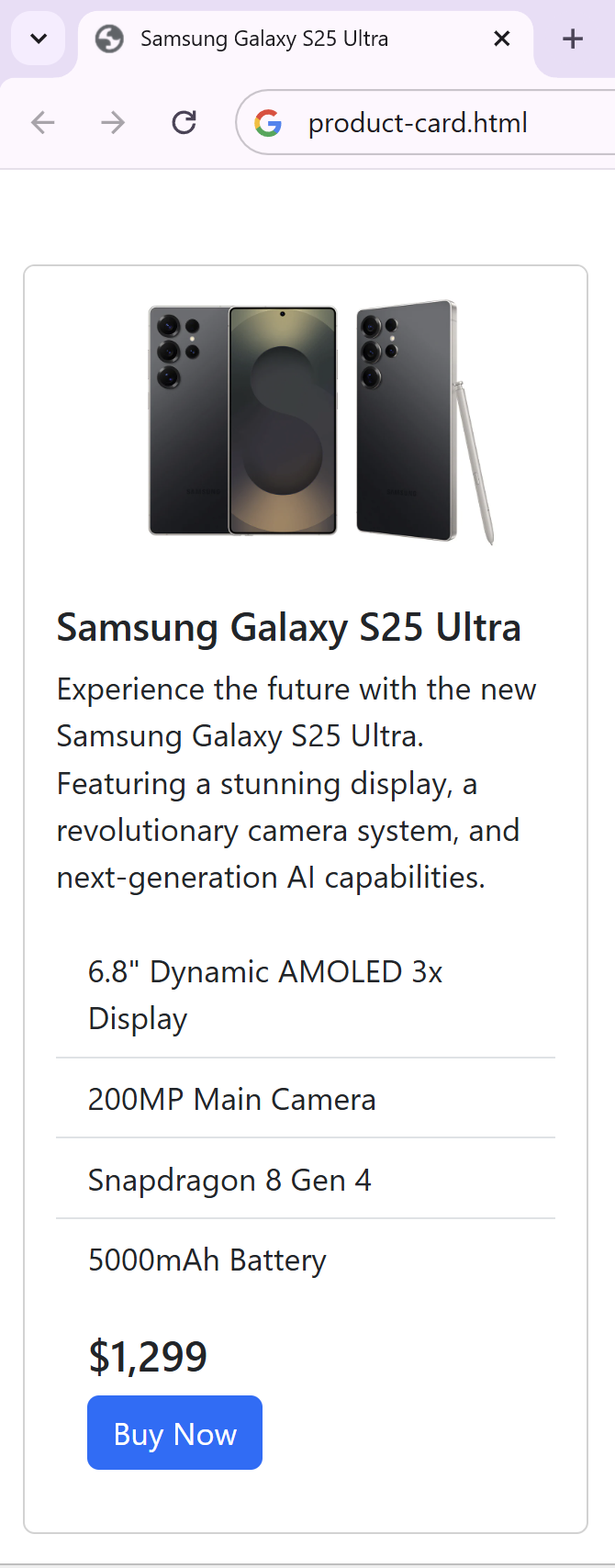
Run the prompt in the "products" folder:



### Inserting Images

Download **images** from Internet, save them to an "images" **sub-folder** of "products", then ask Copilot to insert these images in the HTML code.

### The Product Card in the Browser



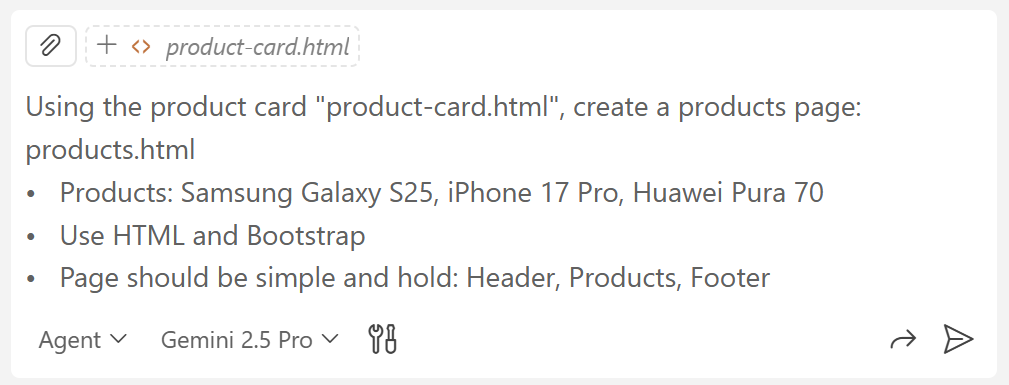
Congratulations, you created your first **product info card** using **Bootstrap** classes!

## Products Page with Bootstrap

Using the product card "product-card.html", **create** a products **page**: products.html

* Products: **Samsung Galaxy S25**, **iPhone 17 Pro**, **Huawei Pura 70**
* Use HTML and Bootstrap
* The page should be simple and should have **header**, **products**, and **footer**

### Sample AI Prompt

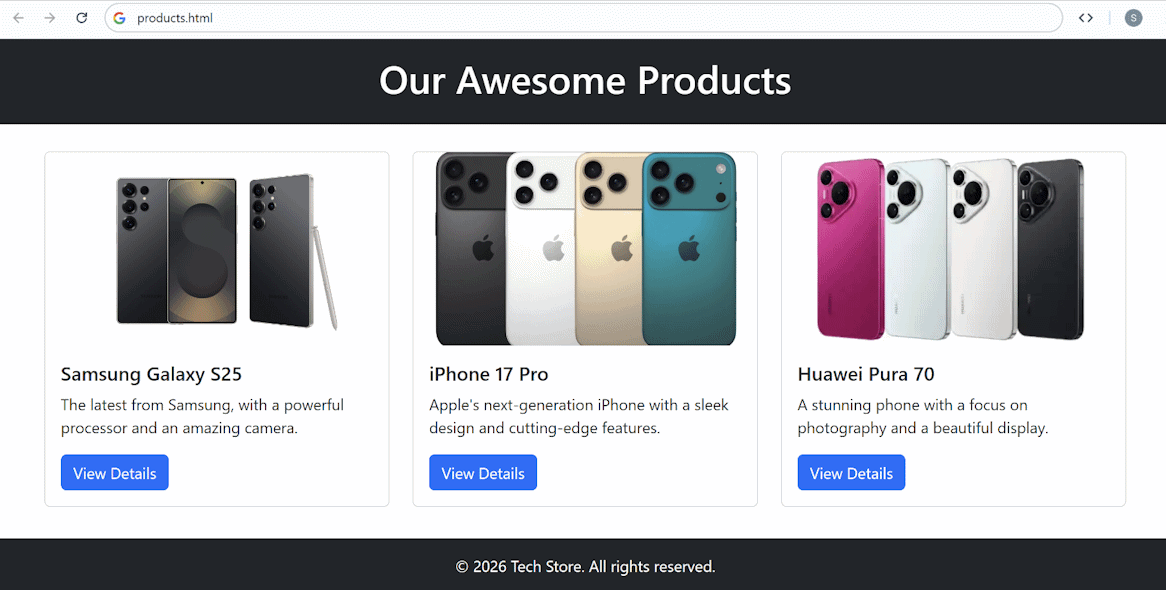


### Inserting Images

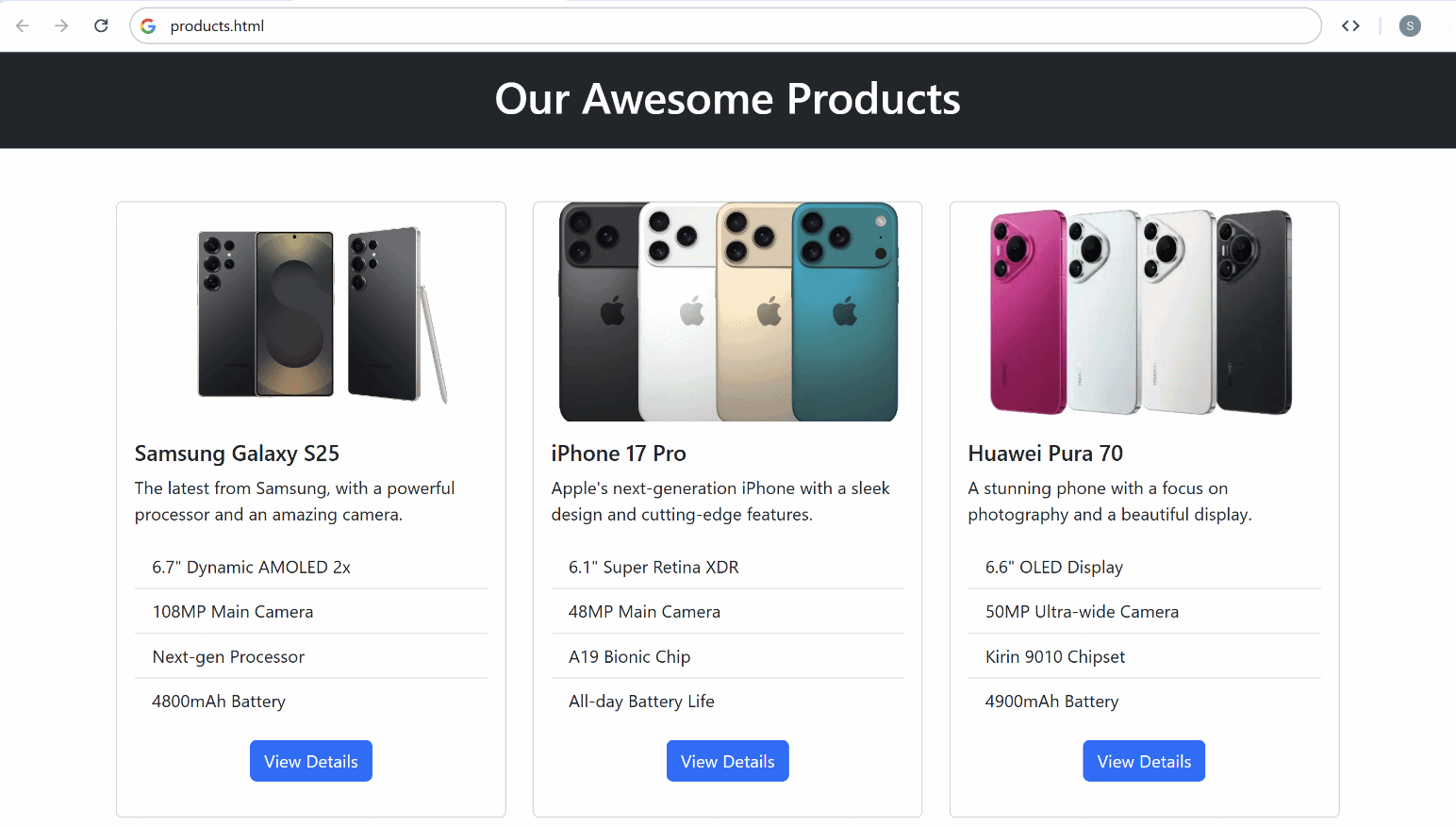
Download **images** from Internet, then ask Copilot to insert these images in the HTML code for every product.

### The Product Page in the Browser

You may want a simplified use of the card style like this one.



The product page can also contain more details like this:

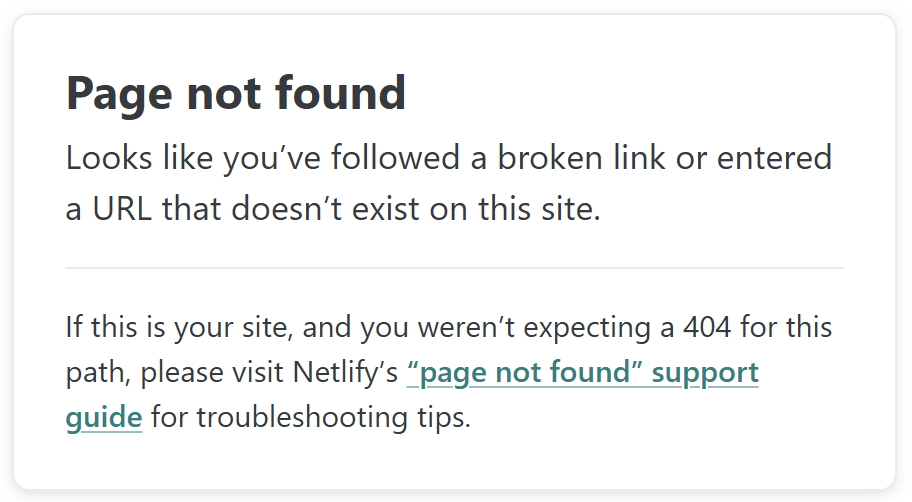


### The App on a Mobile Device

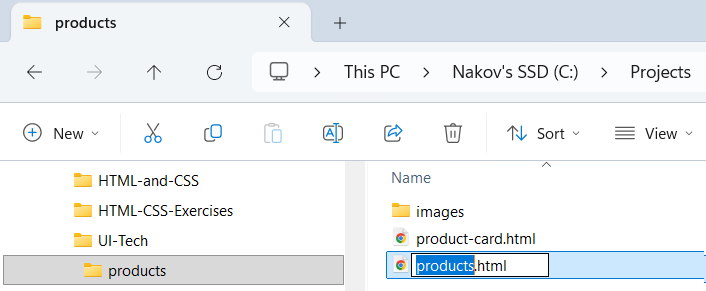
### Deploying to Netlify

To **see** how the **website looks** on **mobile devices**, you can **deploy** it on Netlify. **Log in** your profile, go to **Projects** and **drop the folder with the application files** directly. Use the link to open the application on a mobile device.

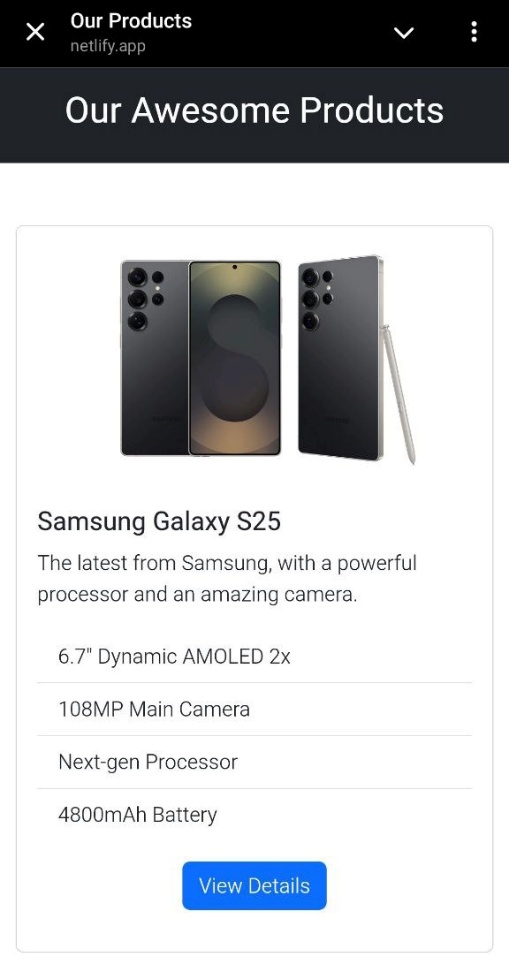
Keep in mind that **Netlify looks for** a file named "index.html" when your deployed Web site is open from Internet. If this file is missing, after you deploy your Web site files, you will get the following error:



You may need to rename the **main html** file to index.html (instead of products.html / cars.html / etc.). **Change it back** to its **original name** after you deploy your code to Netlify:



This is how your site will look like on **mobile devices**, when open from Netlify:



## Dog Breeds Website

In a **new folder**,e.g. "dogs", **create** a **website** about **dogs** and different **breeds**.

* Create a page "dogs.html" to display info about dogs and different breeds.
* Use **HTML**, **CSS**, **JS** and **Bootstrap**
* Create a **breed card** component
  + Top **image** + **title** + **description**
* Add **hover effects** (button changes color, card slightly lifts-up)
* Implement a standard **layout**: **header**, **navigation bar**, **main**, **footer**
* Start with a "**Hero**" section
* Include section "**Featured Breeds**": use carousel
* Include section "**All Breeds**": display all product cards
* Include section "**FAQ**": use accordion
* Hold all breeds in JavaScript array and render them dynamically at runtime

### The AI Prompt

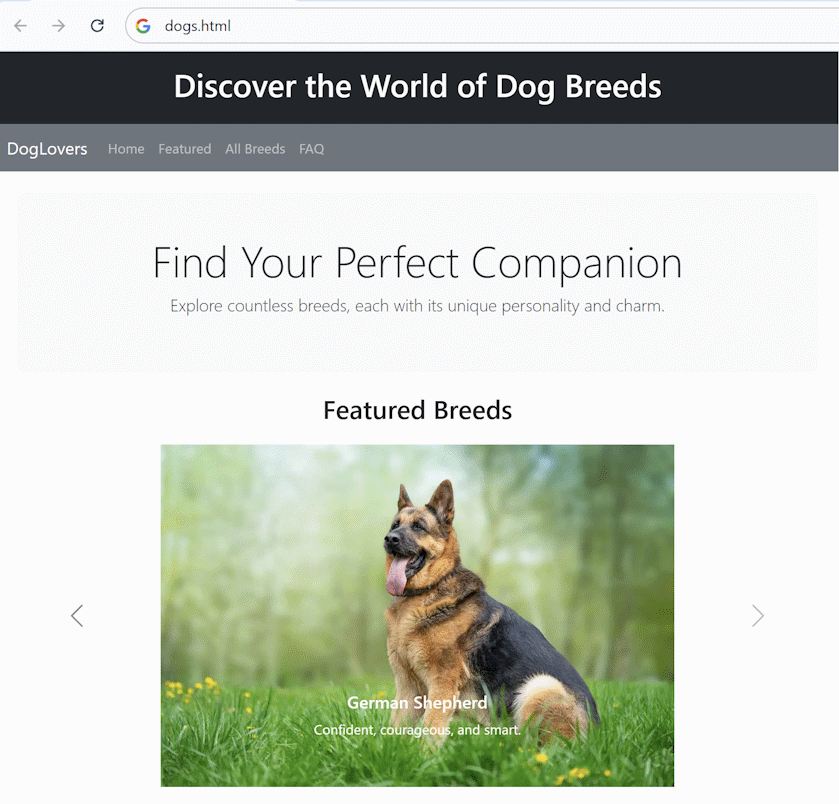


### Inserting Images

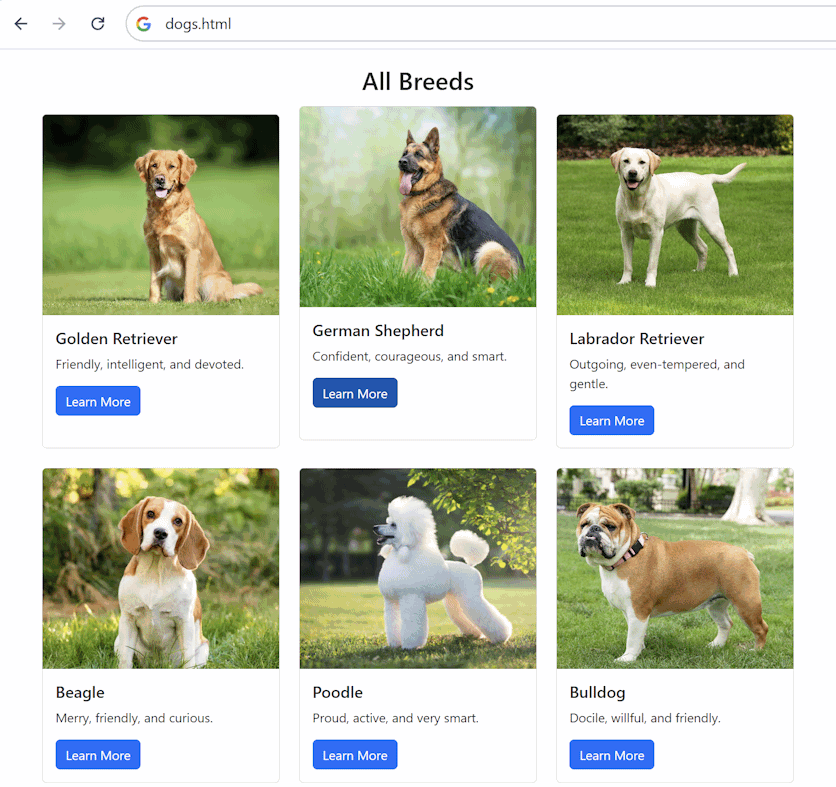
Download **images** from Internet, save them to an "images" **sub-folder** of "Dog Breeds" project, then ask Copilot to insert these images in the HTML code.

### The Website in the Browser

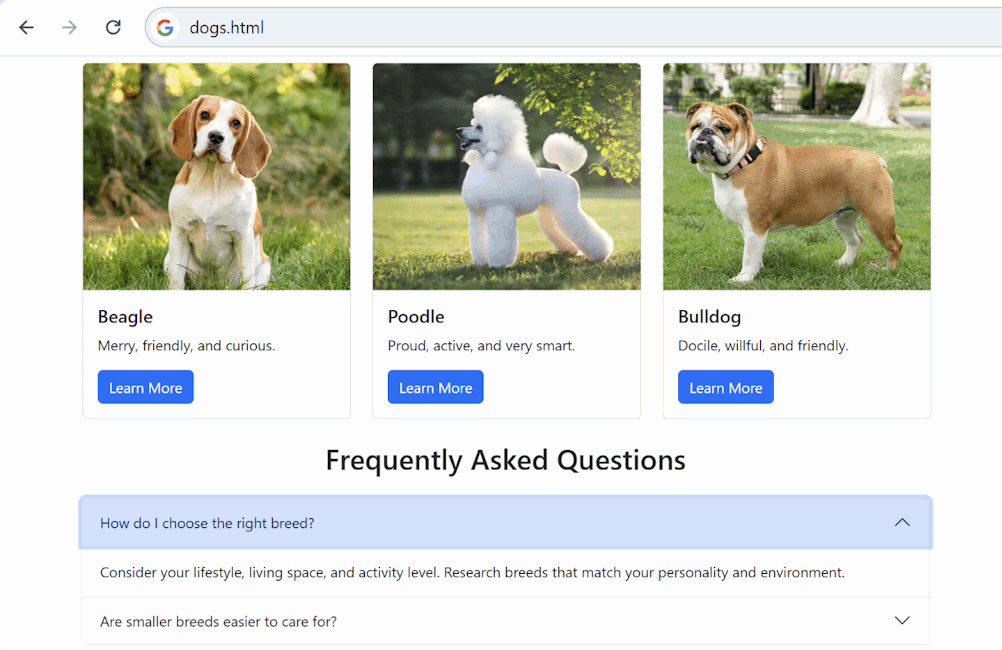
#### The Hero Section and the Carousel



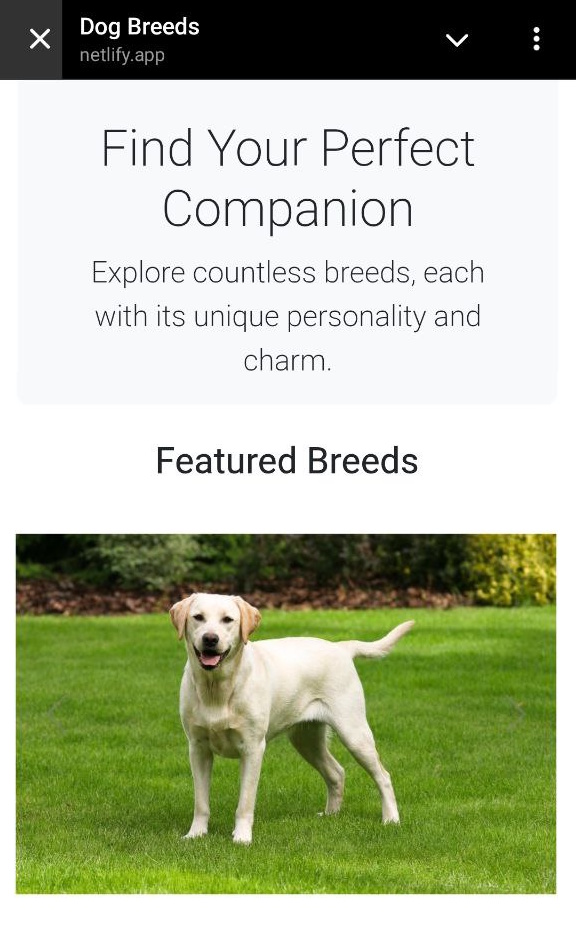
#### All Breed Cards



#### The Accordion FAQ Section



### The App on a Mobile Device



## Cars and Tires Marketplace

In a **new folder**, **create** a **website** for selling cars and tires.

**Note**: this task is complex!

* **Use** **a frontier LLM model** in the coding agent.

In this exercise you will not be using cars / tires photos.

### The AI Prompt

The requirements can be divided into a few subcategories:

#### Cars and Tires Marketplace

**Implement** a **web page** for "Cars and Tires Marketplace", named "cars.html".

* Page structure: **header** with navigation, **main** content, **about** section, **contacts**, footer
* **Navigation bar** with the following tabs: **Home** | **Cars** | **Tires** | **About** | **Contacts**
* Use **breadcrumbs**: Home -> Cars | Home -> Tires
* Show **tabs** in the page main content: **Cars** | **Tires**
* Implement **navigation**: scroll to selected page section or tab

#### Cars Tab

Implement **tab "Cars"**:

* Section "**Hot Cars**": **carousel** with promoted cars
* **"Search" form** -> include the following filters:
  + by brand
  + by model
  + by year
  + by maximum price
  + by kilometers
* "**Search Results" section**:
  + Display the search results as **cards** with **paging**
  + Display cars info in **accordion "View Details"**

#### Tires Tab

Implement **tab "Tires"**: keep it empty for now.

* **Section "Hot Tires"**: **carousel** with promoted tires
* **"Search" form**:
  + by tire
  + by width
  + by tire height
  + by wheel
* **"Search Results" section**:
  + Display the search results as **cards** with **paging**
  + Display tire info in **accordion "View Details"**

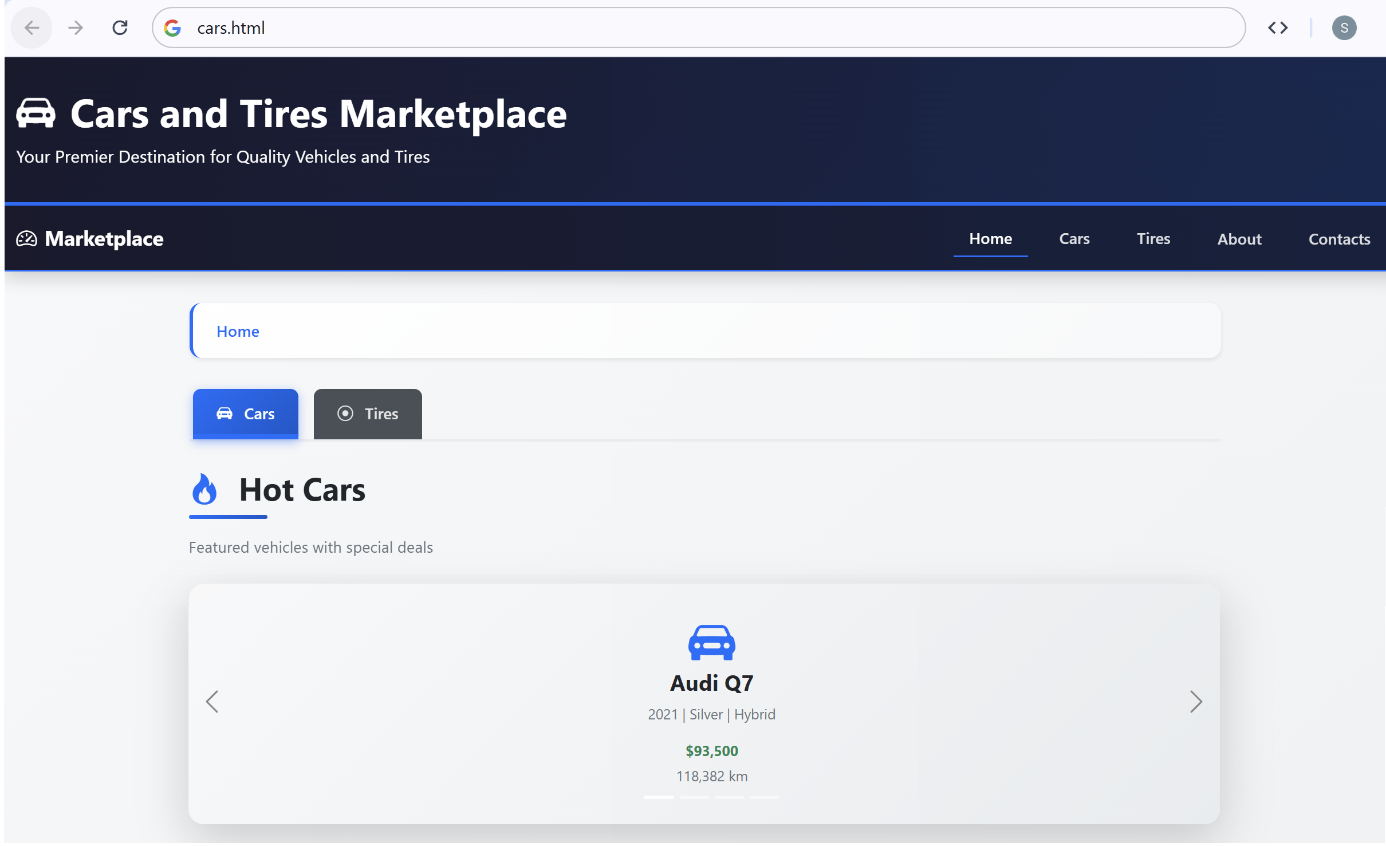
#### Data

* Hold the **cars** and **tires** in the **browser local storage** and render them dynamically at runtime
* Initially generate **20 sample cars** and **20 sample tires** in the local storage

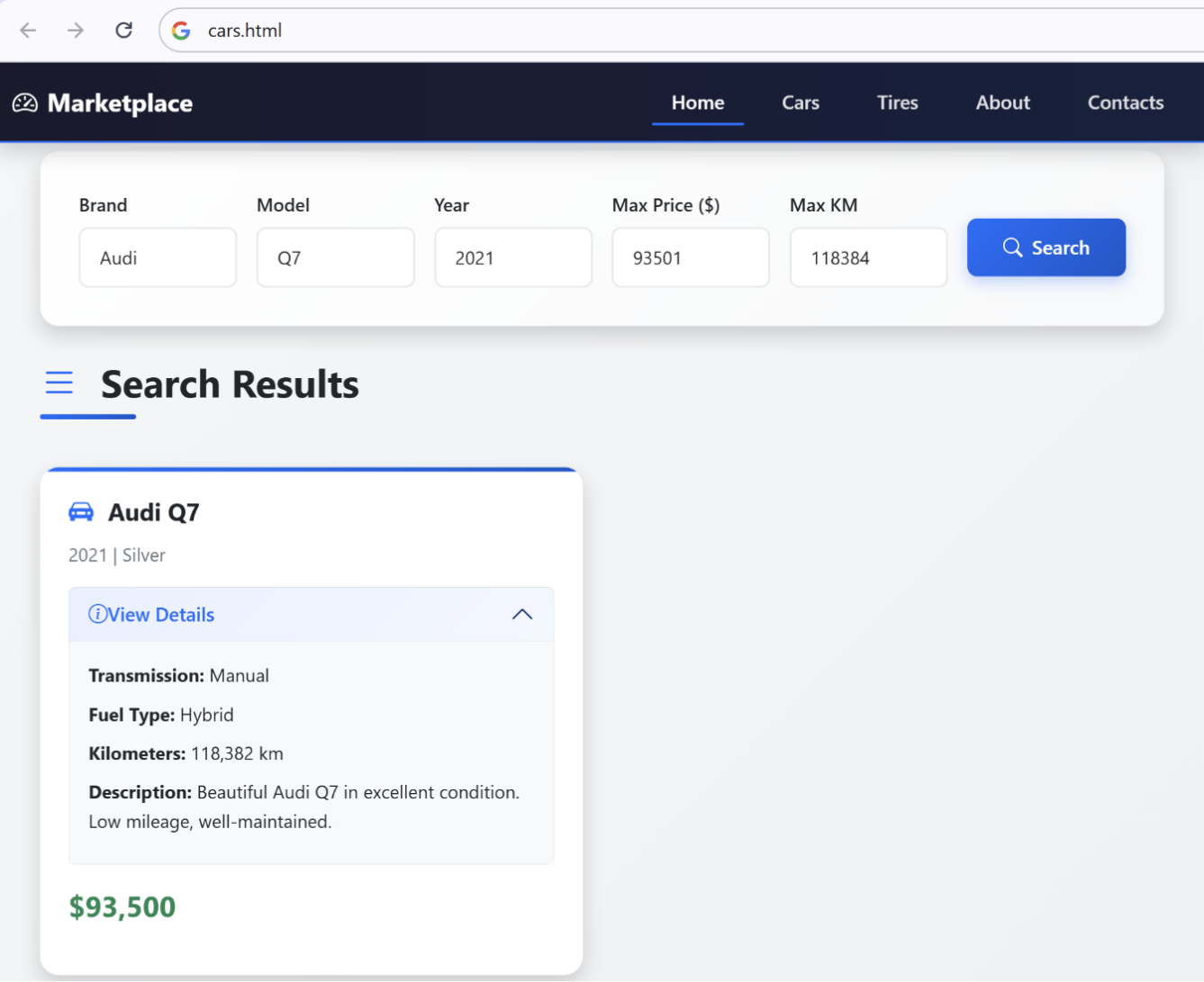
#### Technologies and UI

* Use **HTML**, **CSS**, **JS** and **Bootstrap**
* Implement **modern U**I, with **icons**, **transitions**, **effects**, etc.

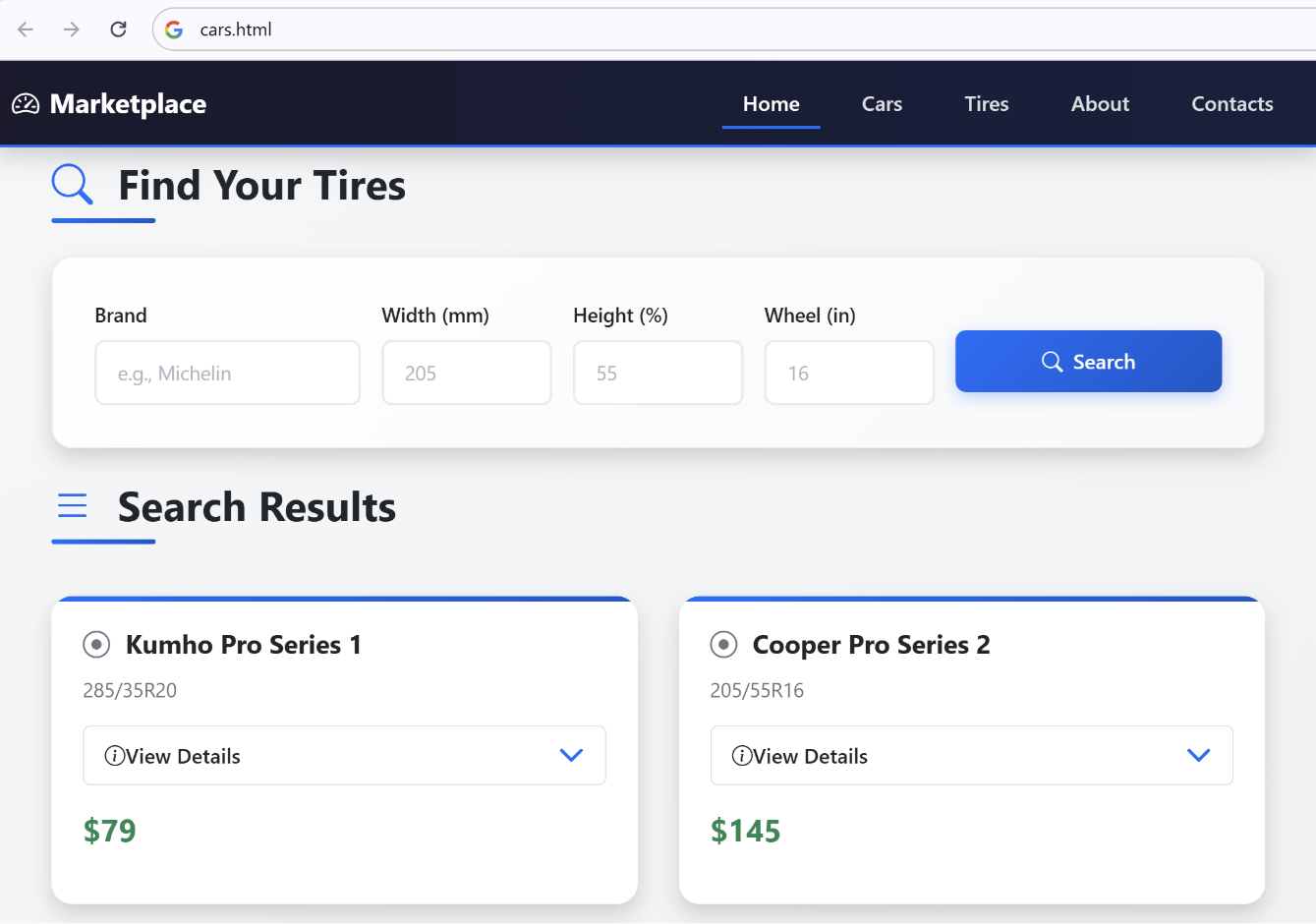
### The Website in the Browser



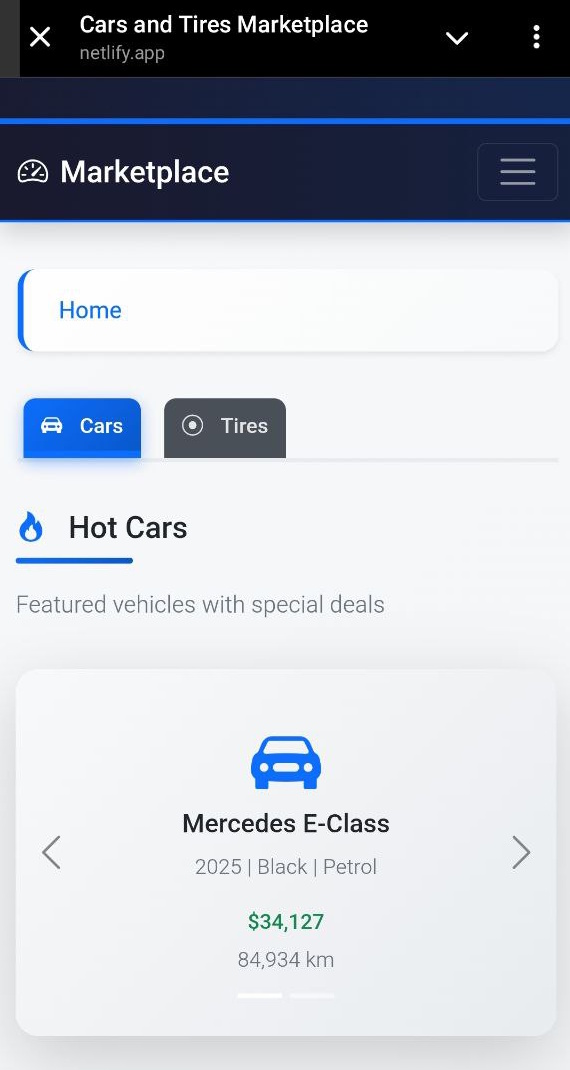
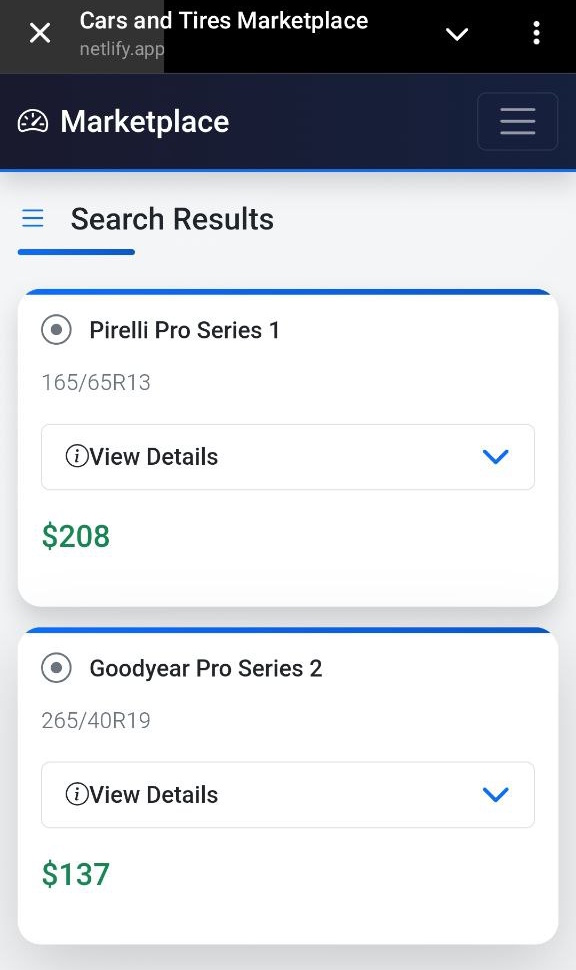
#### The Accordion Car Details



#### The Tire Search Menu



### The App on a Mobile Device

## Cars and Tires Marketplace Admin

**Implement** **Admin Panel** for the Cars and Tires Marketplace: cars-admin.html

* Page structure: header, navigation sidebar, main content, footer
* Tabs (in the navigation sidebar): Cars | Tires

### Cars Tab

* List all **cars** as cards
* Implement a [View Details] button
* Implement the commands [**Add]**, [**Edit]**, [**Delete]** accessible through **modal popups** requiring for the user to press a [Confirm] button in the end

### **Tires Tab**

* List tires as cards
* Implement a [View Details] button
* Implement the commands [**Add]**, [**Edit]**, [**Delete]** accessible through **modal popups** requiring for the user to press a [Confirm] button in the end

### Data

* Use the data model from the previous task "cars.html"
* Cars and tires data is stored in the **browser local storage**

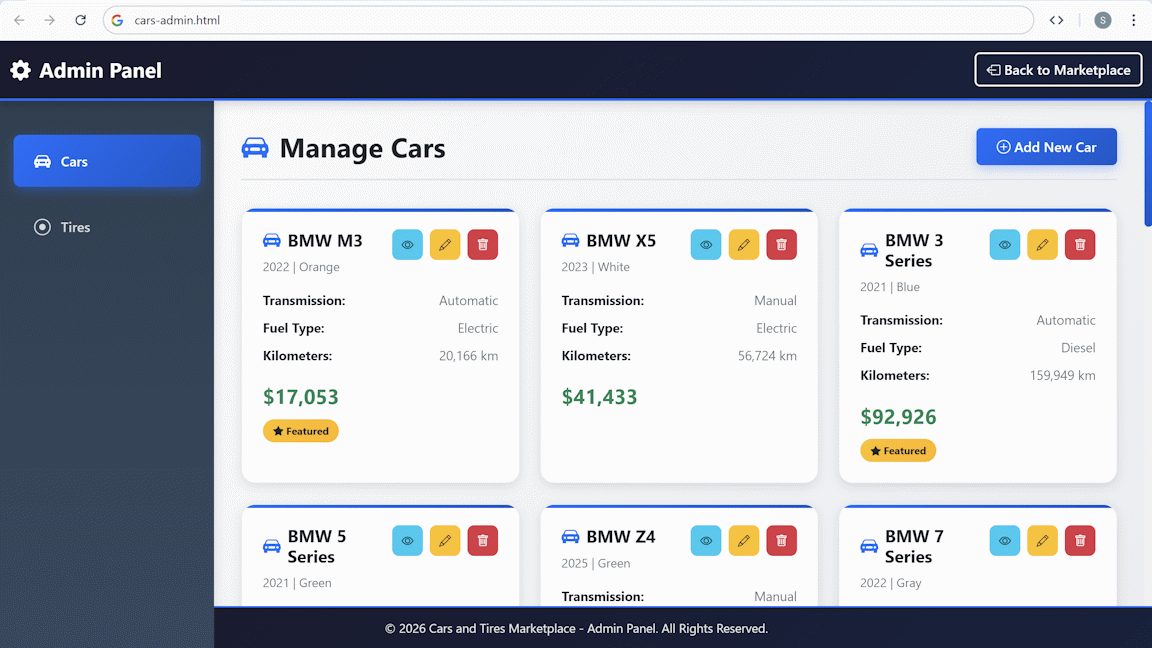
### Technologies and UI

* Use **HTML**, **CSS**, **JS** and **Bootstrap**
* Implement **modern UI**, with **icons**, **transitions**, **effects**, etc.
* Action **buttons**: implement as **icon** + **tooltip**
* Implement **alert notifications** after each operation (auto close after 3 seconds)

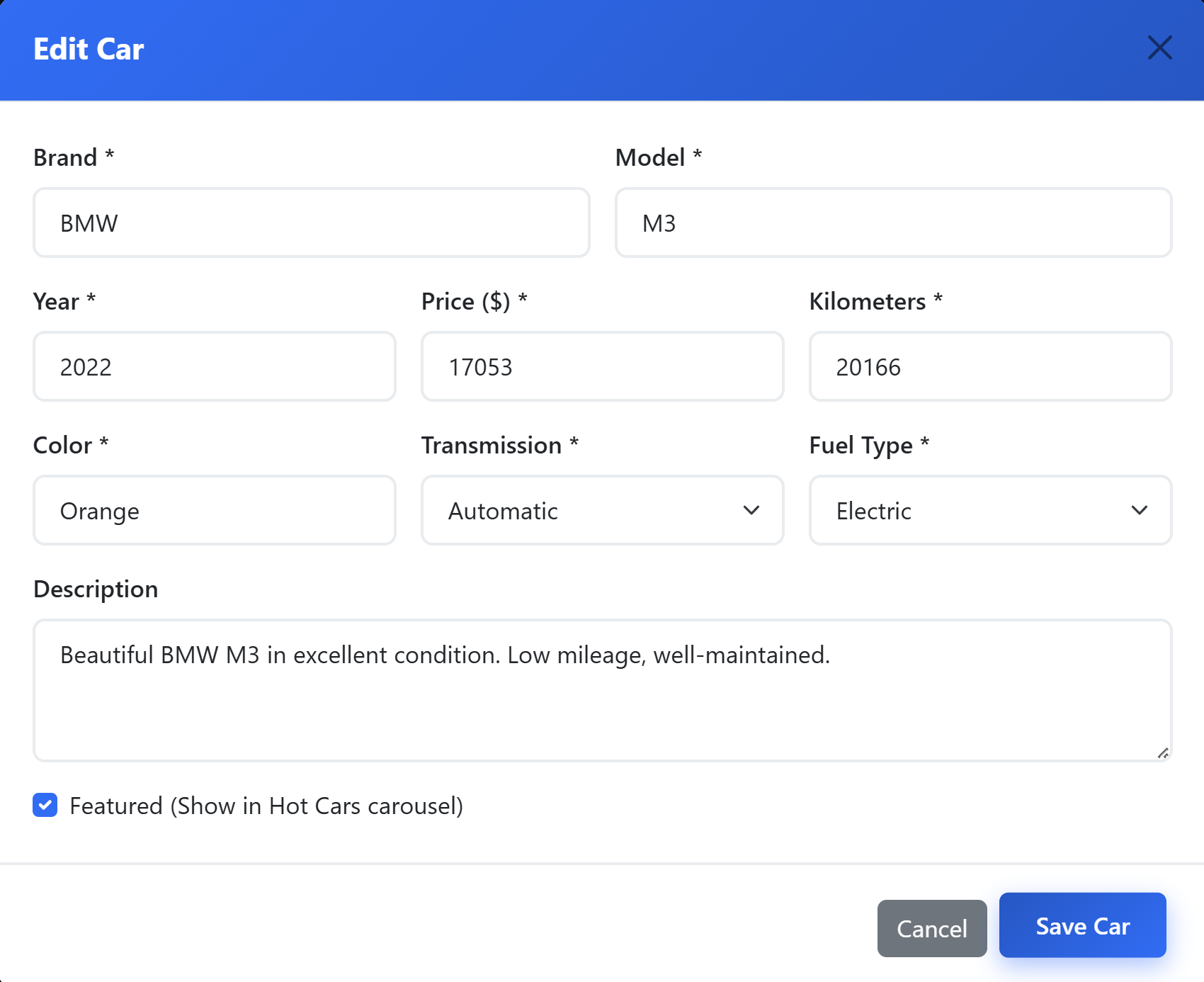
### The Website in the Browser

This is an example of how your admin panel may look like.

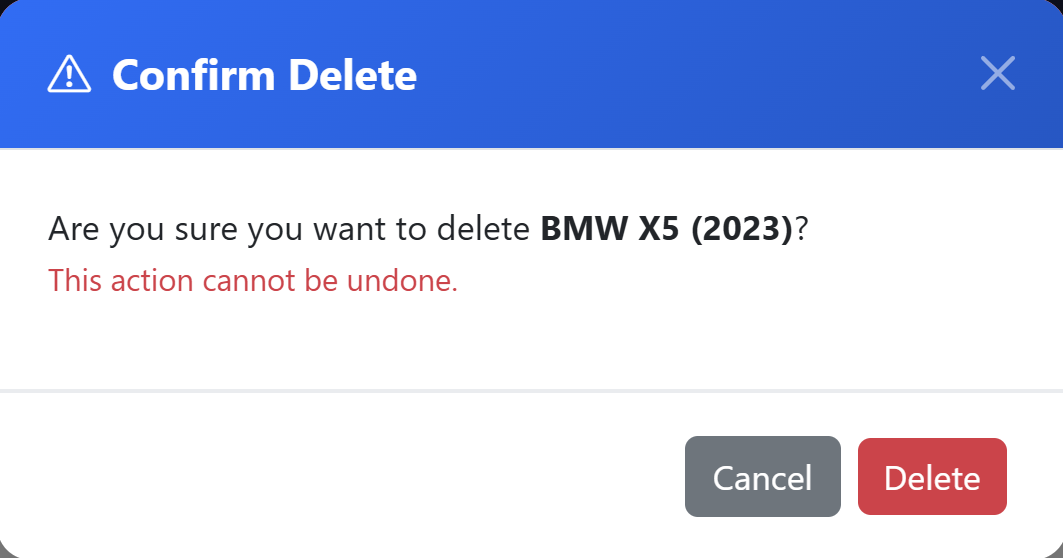
#### The Car Admin Panel



#### The Car Editing Pop-up



#### The Car Deletion Pop-up



## Wedding Invitation

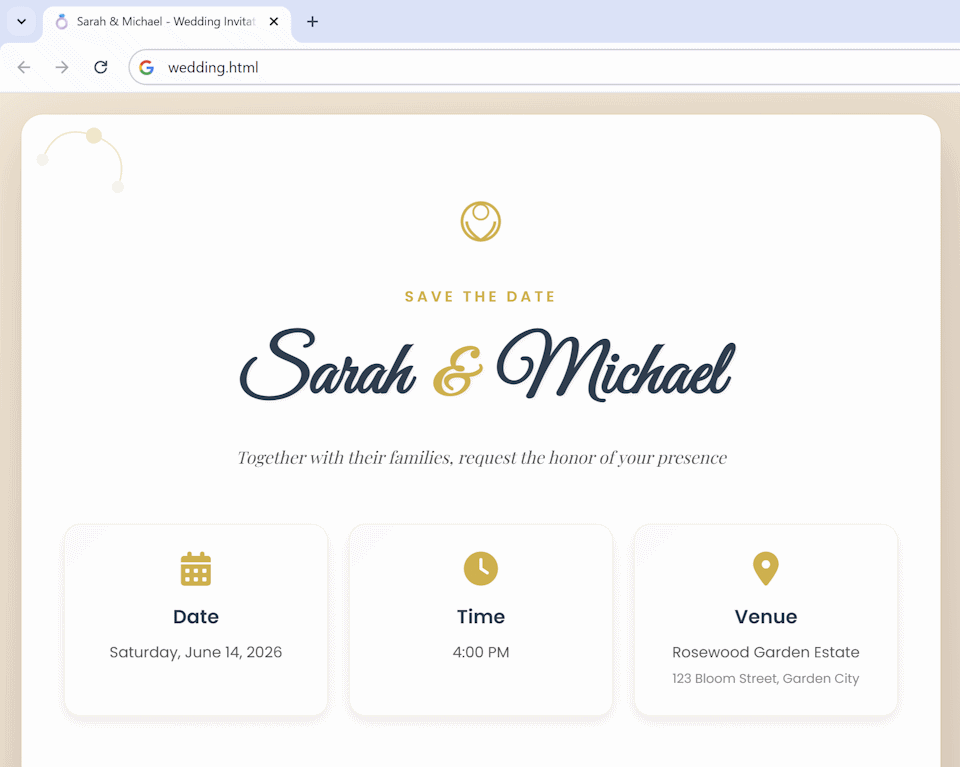
In a new folder, create a **modern wedding invitation page** containing wedding.html and wedding.css.

* Design the wedding **invitation** with **information** and **registration form** for the guests
* Use **HTML** and **CSS**
* Use **icons**, **SVG art**, **effects**, **transitions**, **web fonts**

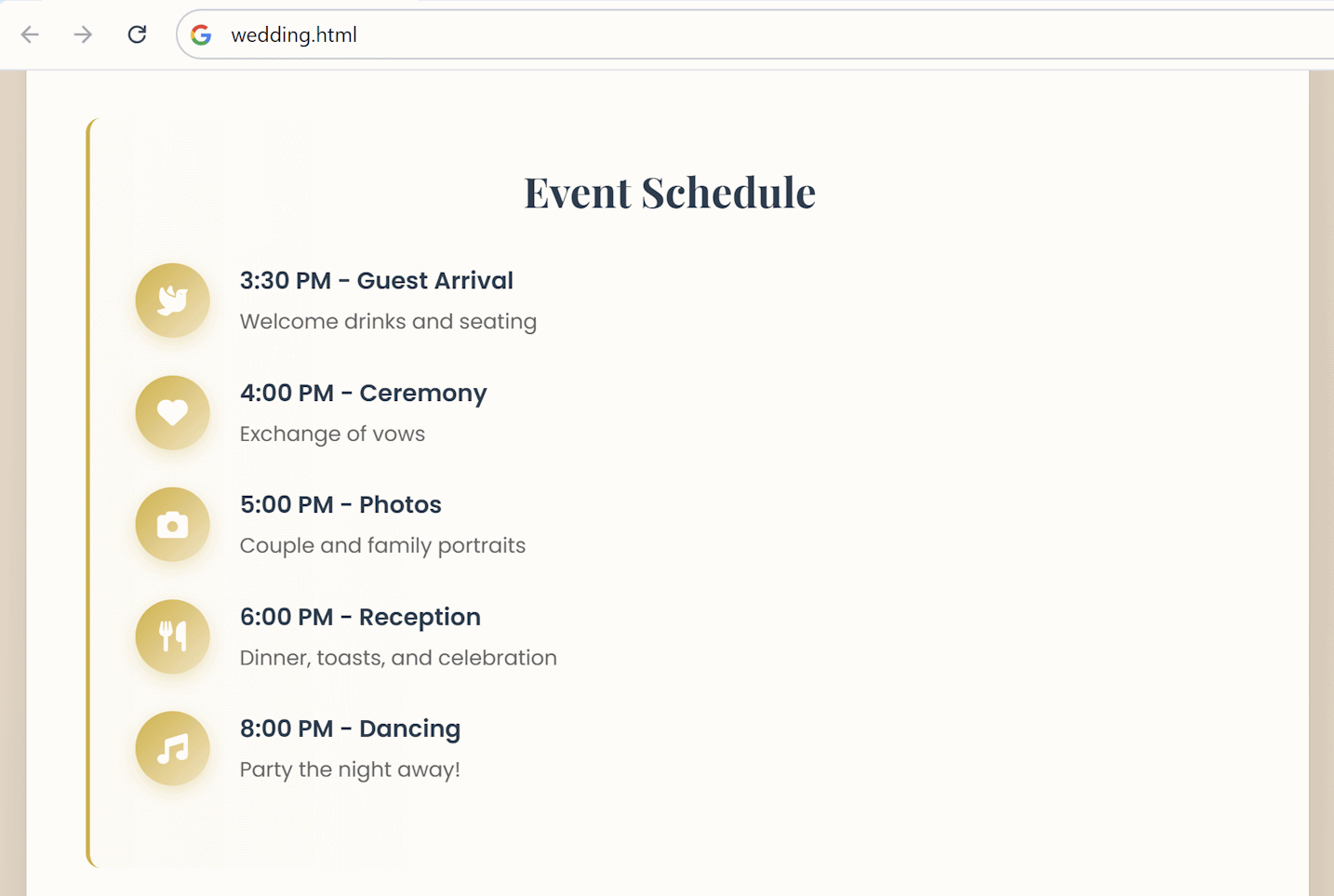
Be **creative**! Experiment and write a prompt of your own.

### The Invitation Page in the Browser

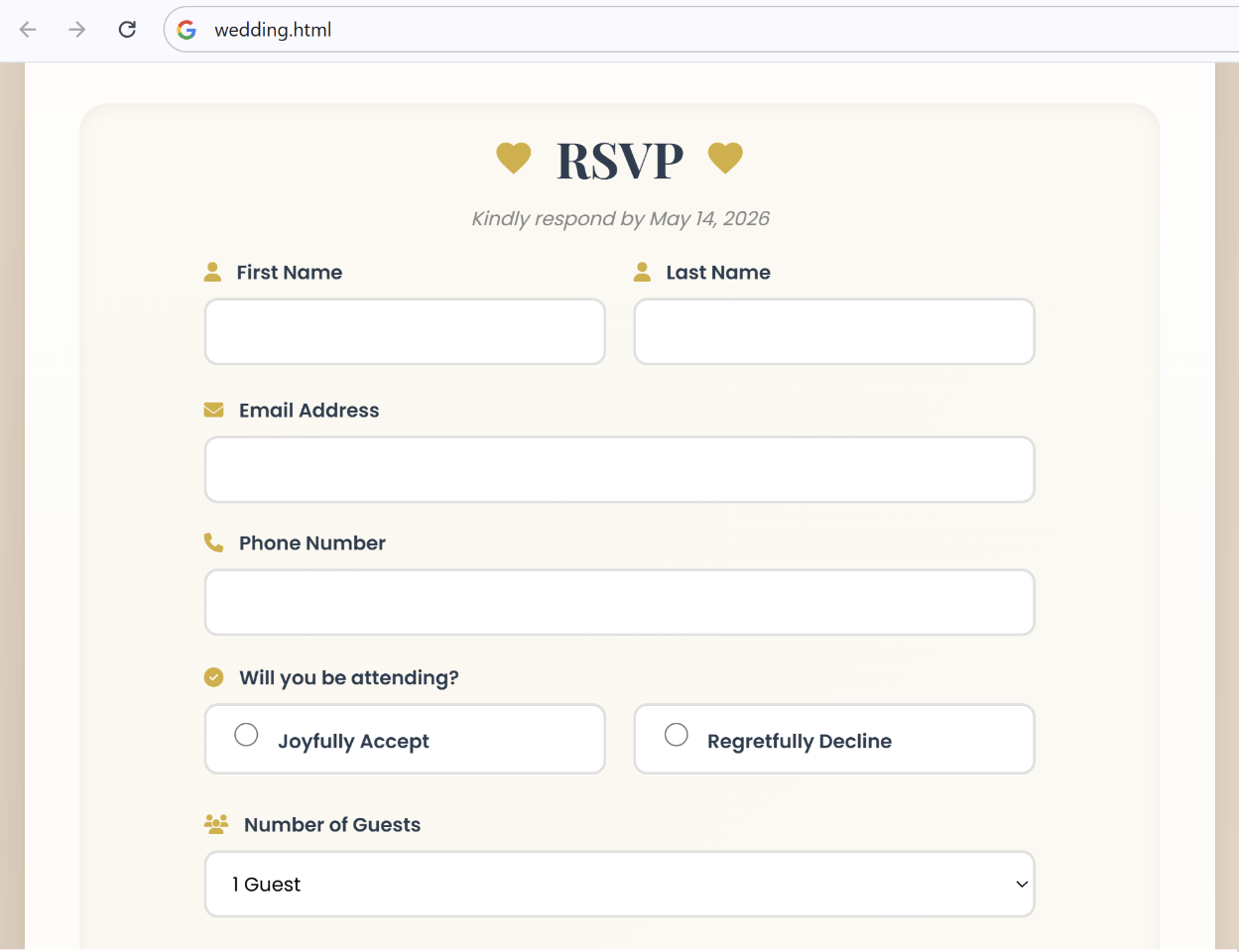
#### The Information Section



#### The Event Schedule



#### The Registration Form



### The Invitation Page on a Mobile Device

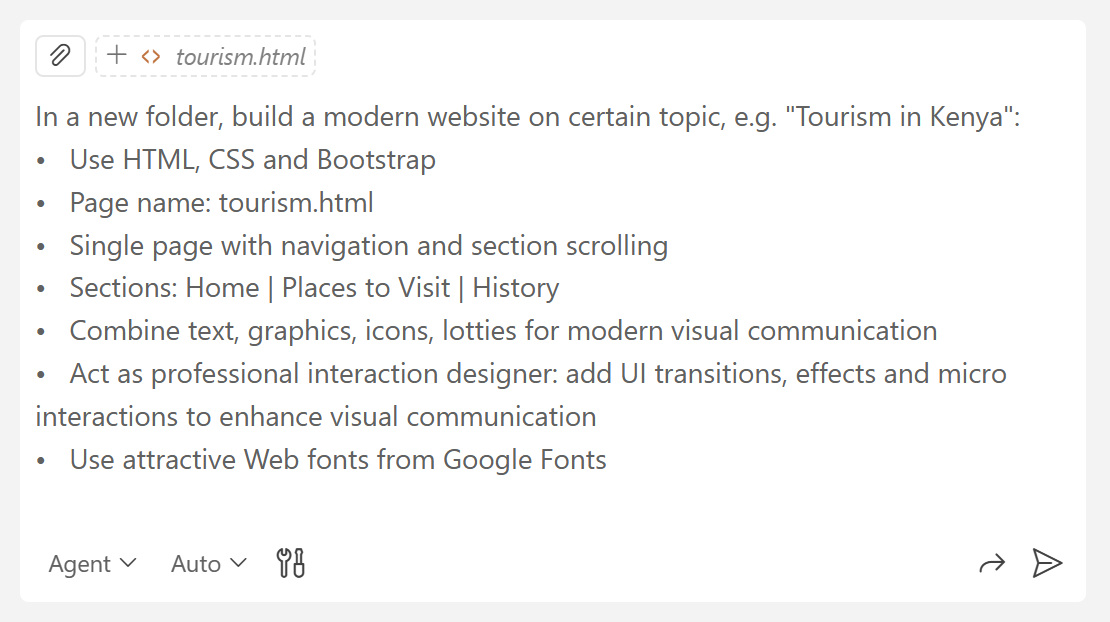
 

## Modern Web Site

In a **new folder**, **build** a modern **website** on certain topic, e.g. "**Tourism in Kenya**":

* Use **HTML**, **CSS** and **Bootstrap**
* Page name: tourism.html
* Single page with **navigation** and section scrolling
* Sections: **Home** | **Places to Visit** | **History**
* Combine **text**, **graphics**, **icons**, **lotties** for modern visual communication
* Act as professional interaction designer: add UI **transitions**, **animations**, **effects** and **micro interactions** to enhance visual communication
* Use attractive Web fonts from **Google Fonts**

### Sample AI Prompt



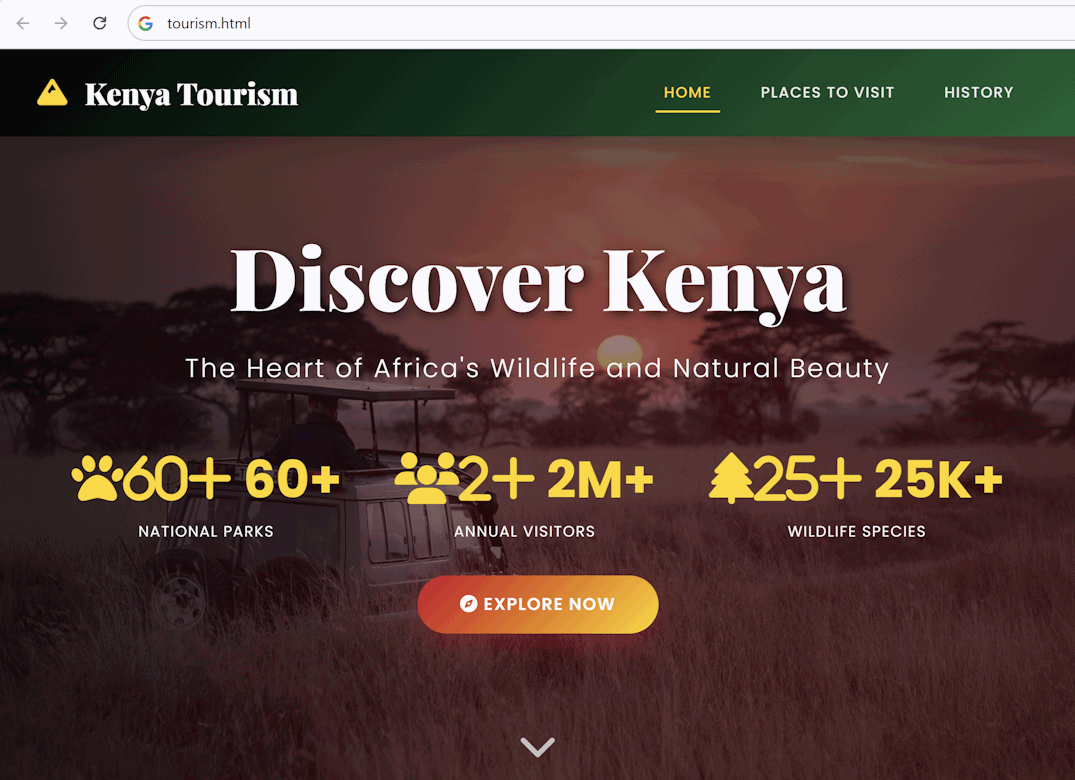
### Inserting Images

Download **images** from Internet, save them to an "images" **sub-folder** of "Tourism in Kenya", then ask Copilot to insert these images in the HTML code.

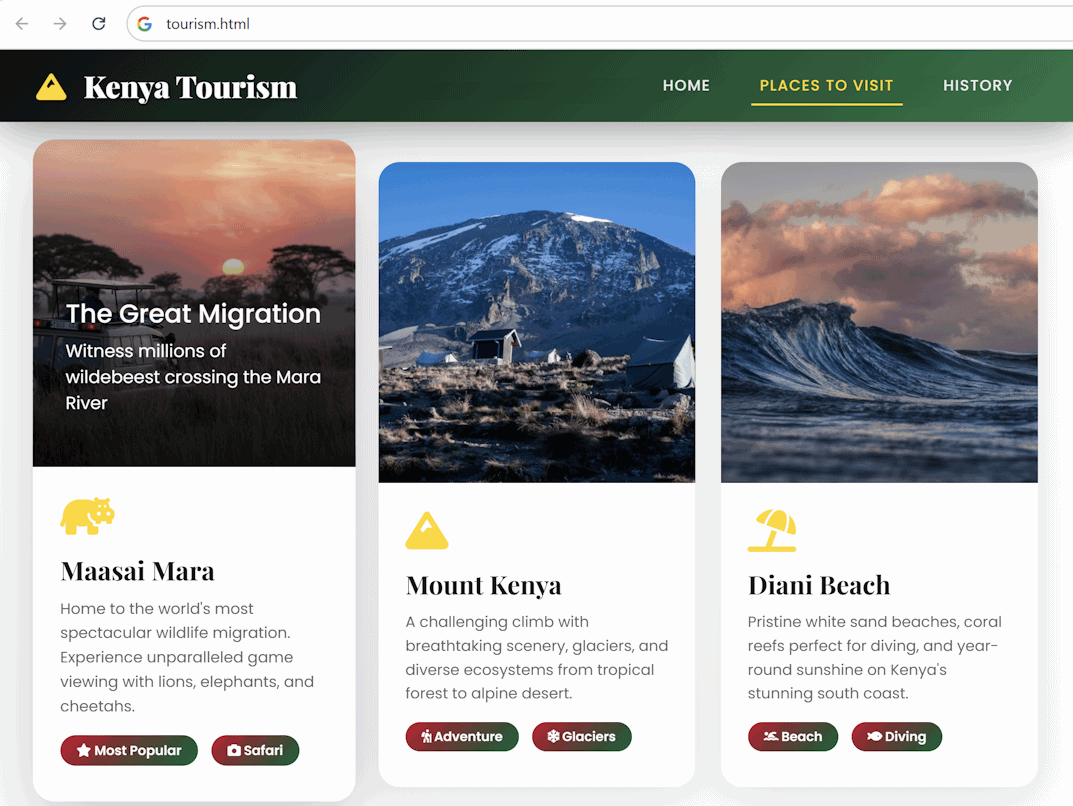
### The Website in the Browser

This is an example of how your modern Web site may look like.

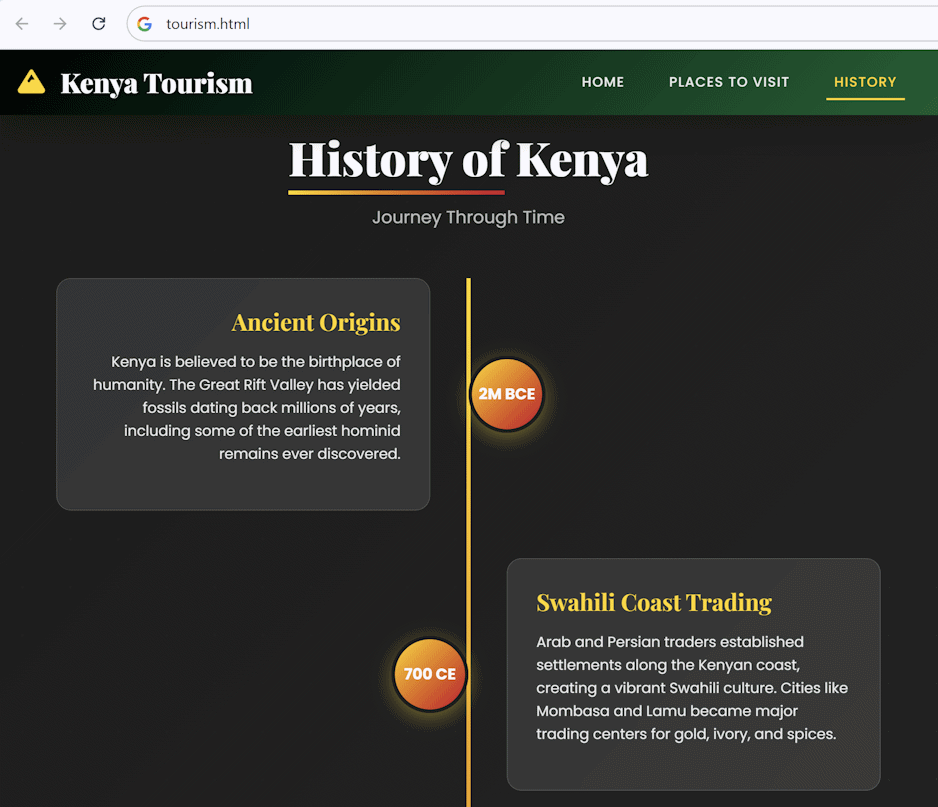
#### The Hero Section



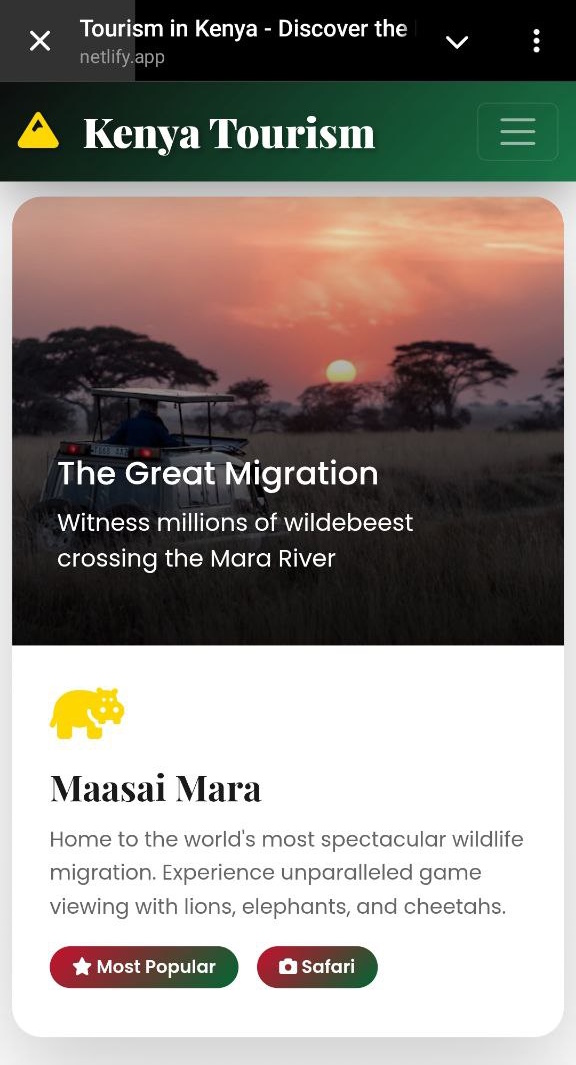
#### Cards with Places to Visit



#### History of Kenya Tree



### The App on a Mobile Device

## App from Screenshots

**Create** a "Password Generator" **app** in plain HTML + CSS + JS (see the attached screenshots):

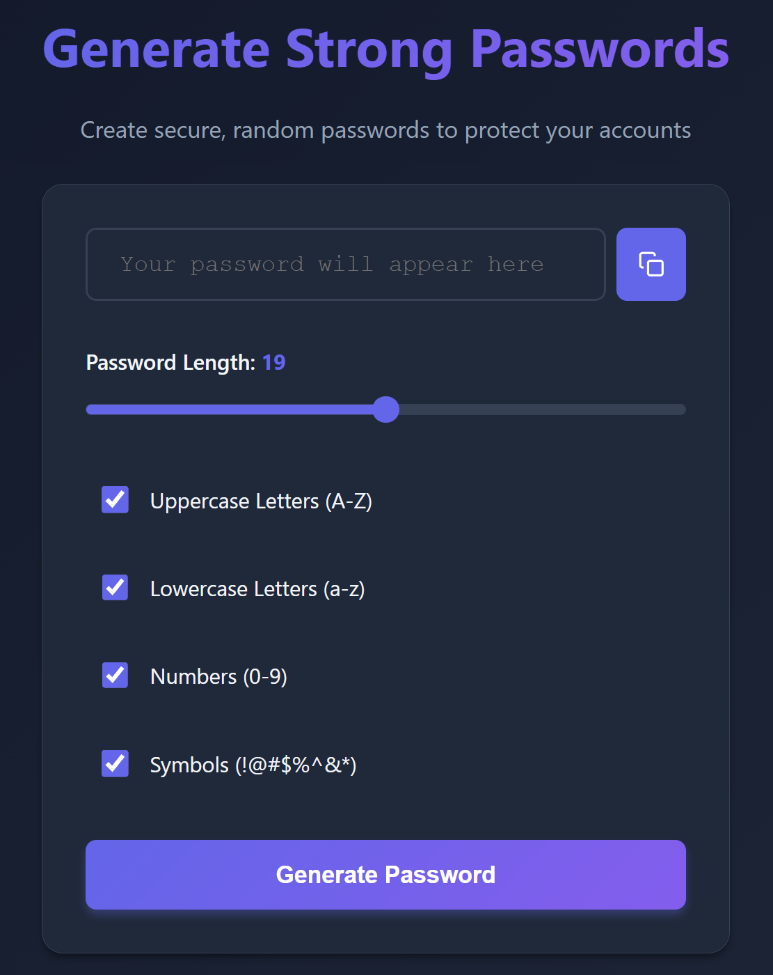
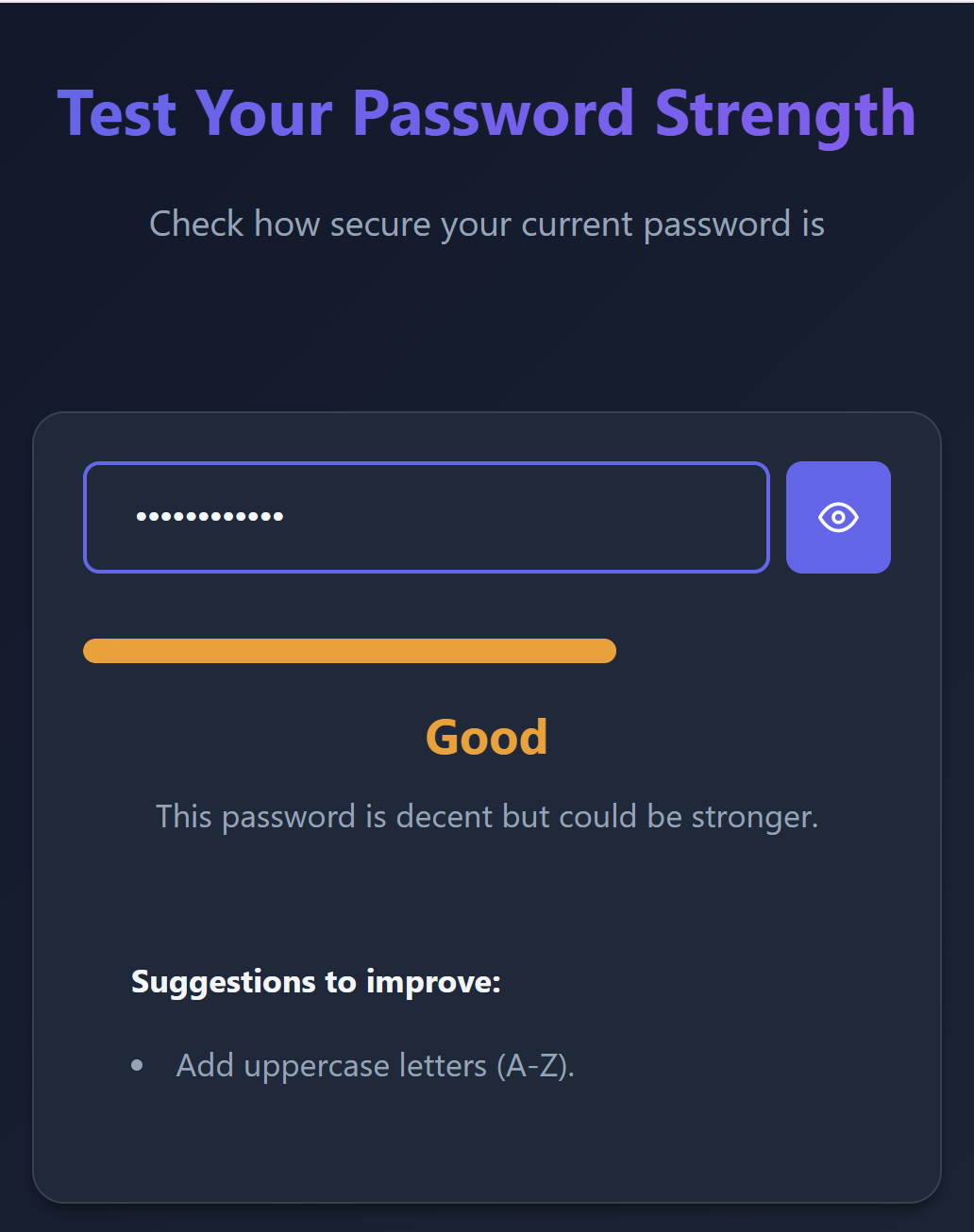
* Implement "**Home**" page (with Password Generator section), "**About**" page, "**Password Test**" page
* It should look like the **screenshots** provided (layout, fonts, colors, UI styles, URLs)
* The app allows users to **generate random passwords** with the **Password Generator** and test their own passwords with the **Password Test**
* In the About section there is an article on how to create strong passwords
* Keep app data in the **browser local storage**

### The Screenshots

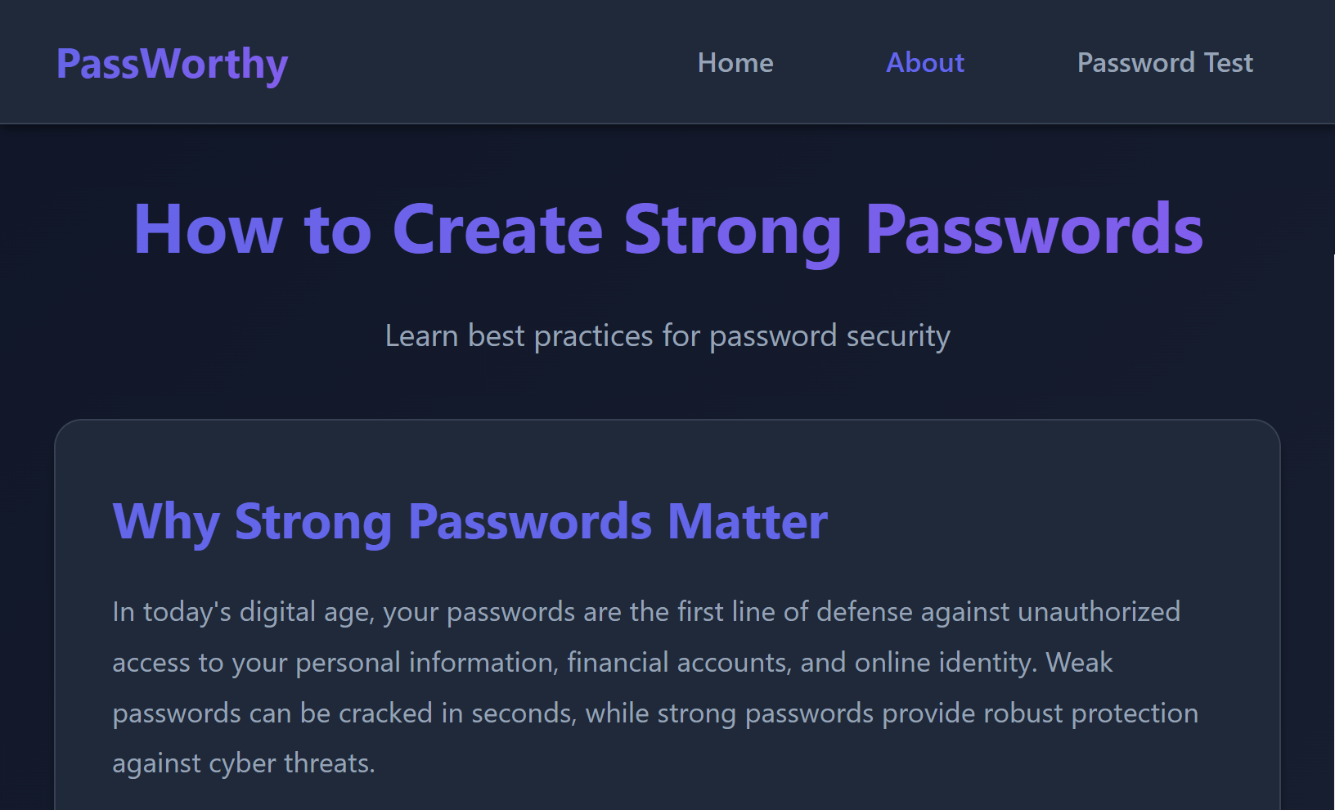
#### The Navigation Bar



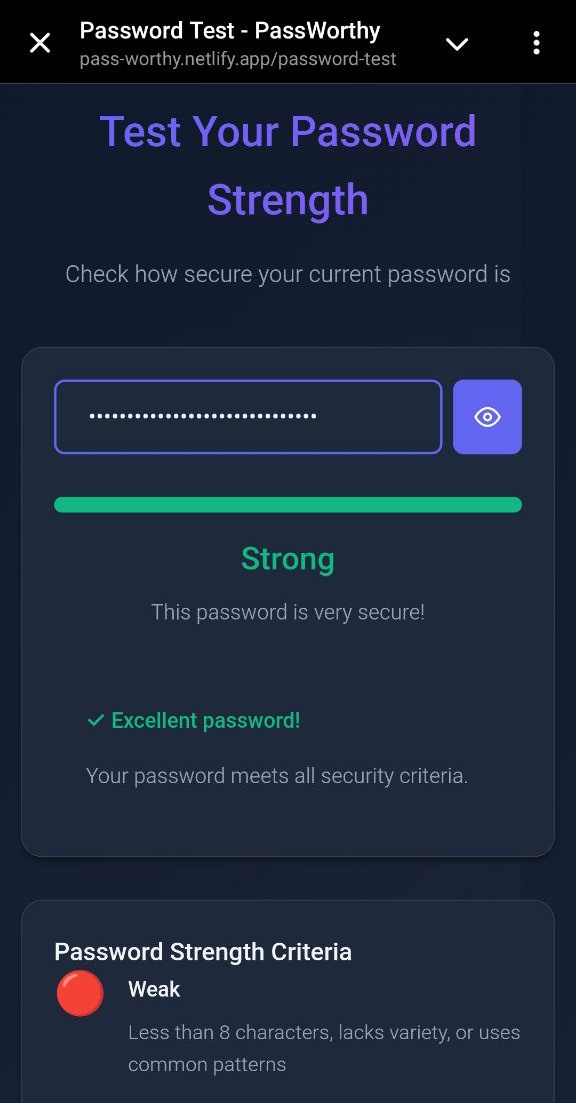
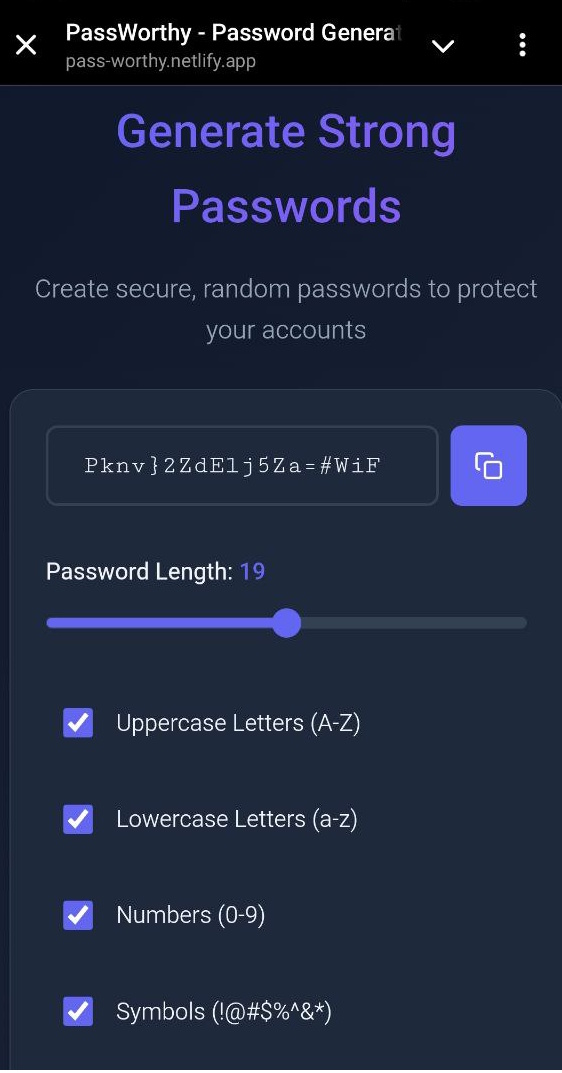
#### The Password Generator and the Password Test

#### The About Page



### The App on a Mobile Device

## Multi-Page App with Vite \*

In this exercise you will **build** a **multi-page web application** using **Vite**, **HTML**, **Bootstrap**, and **JavaScript**. The app represents a **Dogs Marketplace**, where **users browse dog listings** and **administrators manage them**.

The application represents a **simple marketplace for dogs**:

* Dogs for **sale**
* Dogs for **adoption** (free)
* **Puppies**

Each **dog** has:

* Breed
* Age
* Gender (male / female)
* Description
* Purpose (sale / adoption)
* Adopt or Buy button (with a price)

### Pages

* **Home** – welcome page
* **Dogs** – list dogs and view dog details
* **About** – information about the platform
* **Contact** – contact information
* **Admin** – manage dogs (list, add, edit, delete)
* Create a **multi-page app** with **HTML**, **Bootstrap** and **JavaScript**

### Steps to Follow / AI Prompt

Use the following steps as a prompt:

* Setup a **Vite project** in VS Code – it will serve the app
* Generate all needed **Vite files**
* Do not use a starter kit
* Set up a **multi-page app** with navigation
* Pages: **Home**, **Dogs**, **About**, **Contact**, **Admin**
* Implement **navigation** between pages
  + **Change the browser URL** when the active page changes
  + Use clean URLs without hashes (Example: <http://localhost:5173/dogs>)
* Structure the app into **UI components**:
* header
* footer
* pages
  + Use **HTML fragments** for each UI component
  + Use a folder with **separate HTML**, **CSS** and **JS file** for each UI component

### Data Model

* + Keep app data in the **browser local storage**

### Dogs Page

* + Display a **list of dogs** as cards
  + Implement **viewing dog details** (page or modal)
  + Implement forms with **popups**: Buy, Sell and Adopt Dogs

### Admin Panel

* + List of all dogs
  + Add new dogs
  + Offer dogs for free (for adoption) or sell dogs (and set price)
  + Edit existing dogs
  + Delete dogs
  + Use confirmation dialogs for destructive actions

### Dev Agent Instructions

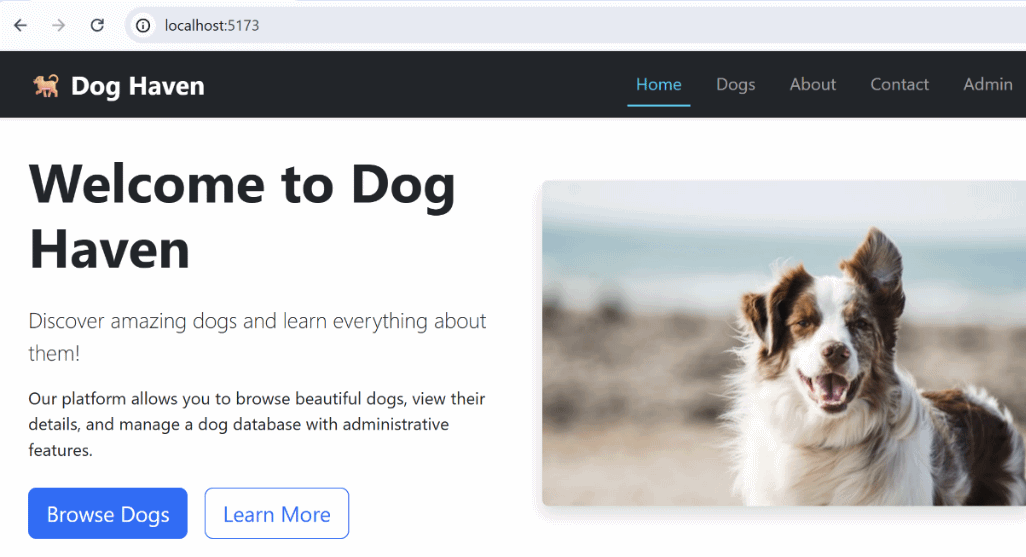
Define short **dev agent instructions**:

* + Create the GitHub Copilot instructions file:
    - .github/copilot-instructions.md
  + Insert instructions about the app structure and components

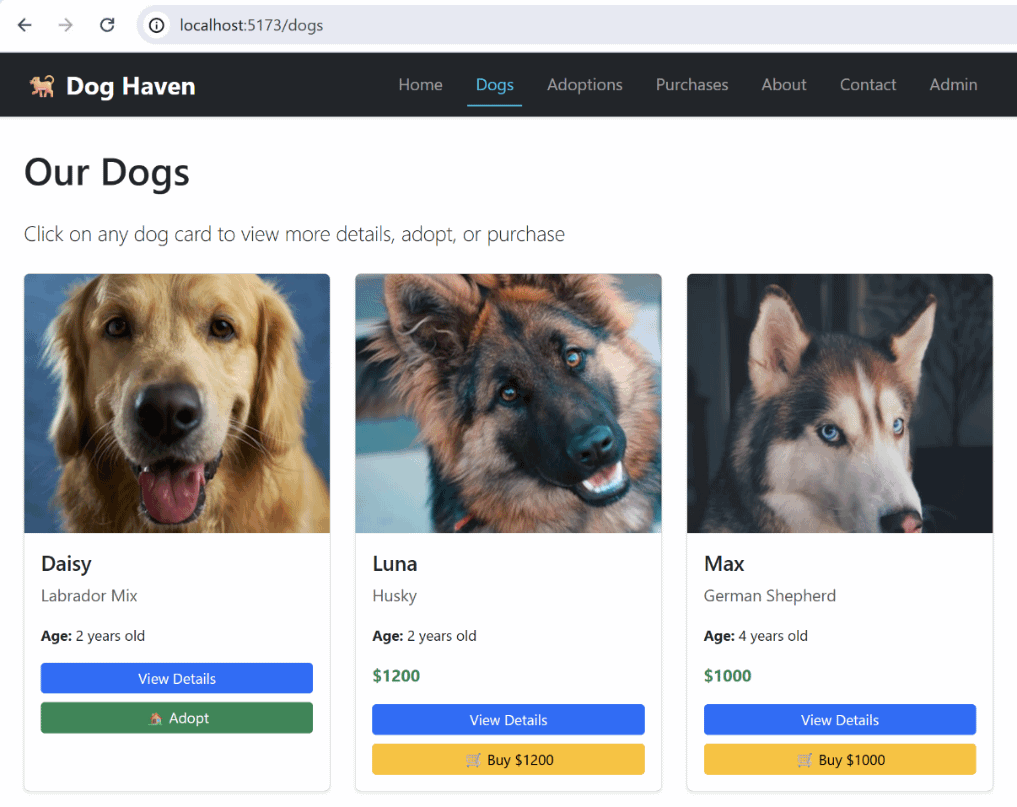
These instructions will **help the AI agent to work correctly** within the defined **app structure**, without making irrelevant changes and incorrect implementations (e.g. generate full page instead of a page fragment).

### The Website in the Browser

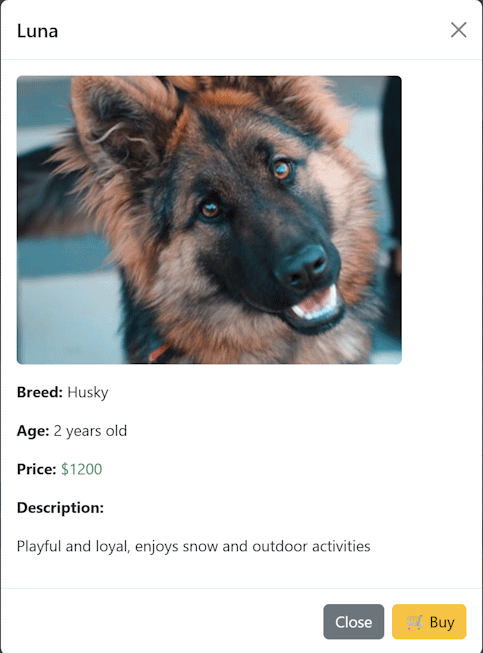
#### The Home Page



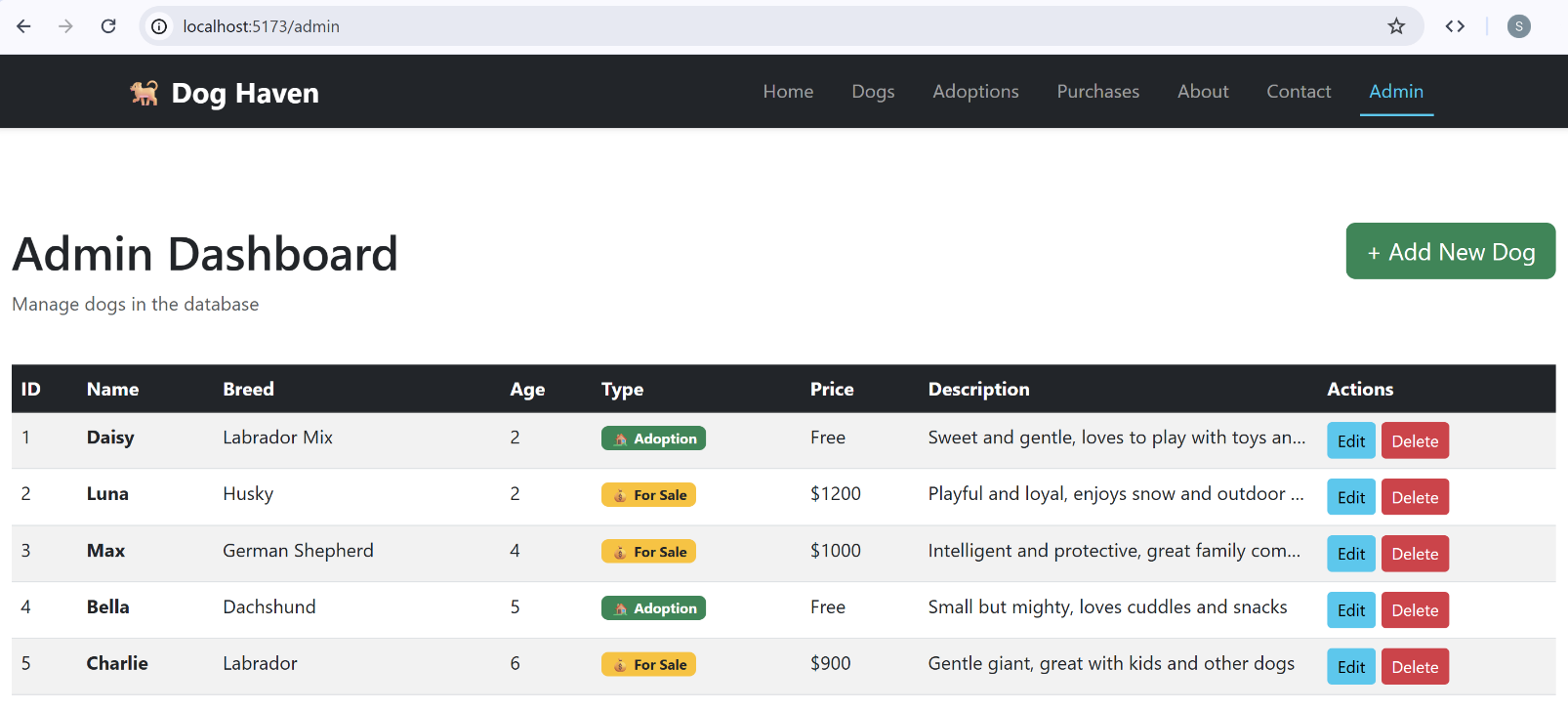
#### The Dogs Page



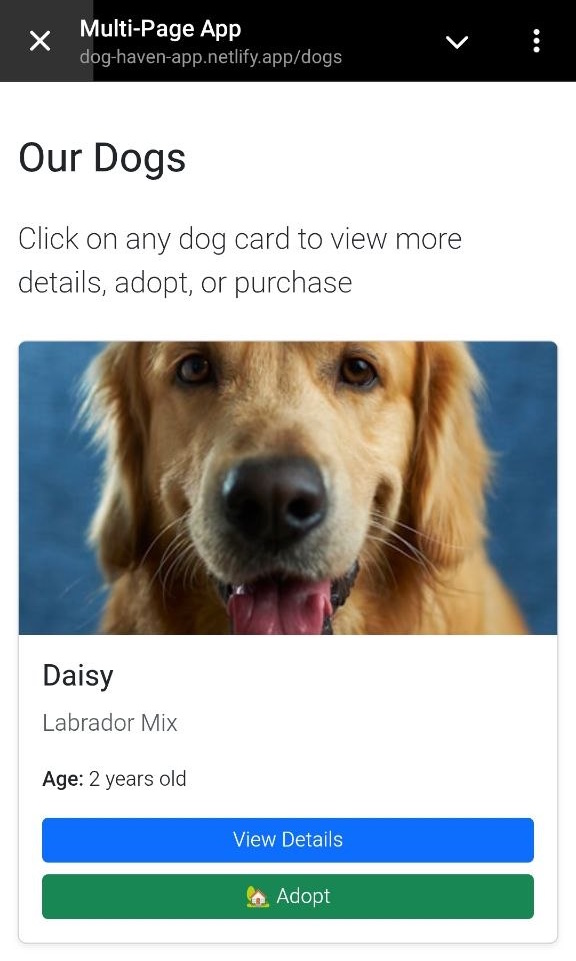
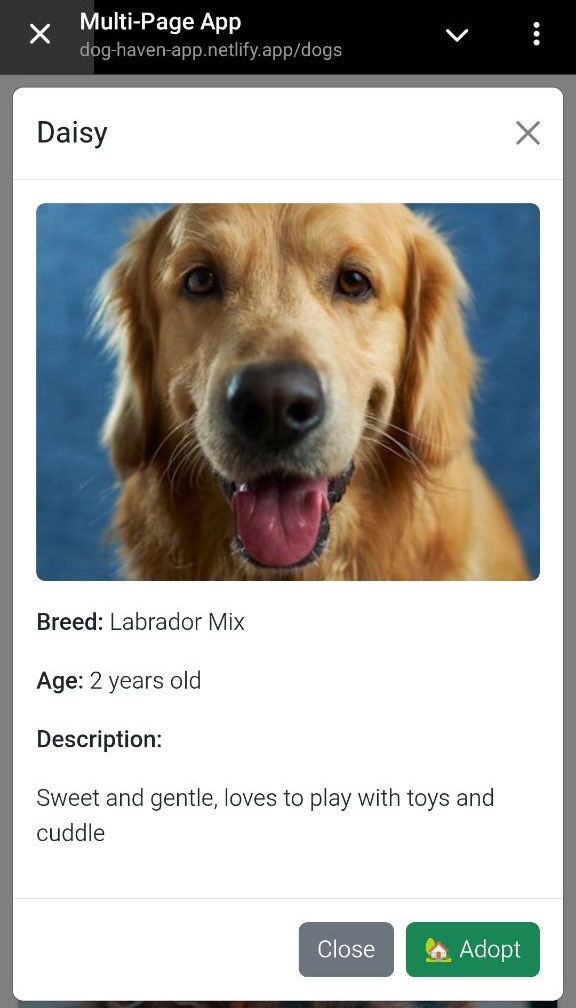
#### A Dog’s Info Card



#### The Admin Panel



### The App on a Mobile Device

### Alternative Sequence of Steps

1. Setup **Vite project** with pages, navigation, routing and UI components, header, footer
2. Create app **data model**
3. Create **Home page**: list dogs
4. Create **Details Page**: view dog details (use routing \details\{id})
5. Create **admin panel**: list / add / edit/ delete dogs (with table + popups) (use routing \admin)