Aufgabe Praxistag Frontend

Die Firma Bluewave Electronics hat mit ihrem Team vor, einen E-Commerce Shop zu bauen. Das Team von Bluewave Electronics besteht hauptsächlich aus Backend Entwicklern, die mit folgendem Tech Stack eine REST API aufgestellt haben:

- Python
- Python RestFramework
- sqlite3 als Prototyp für die Datenbank (Wird vom Team selbst beim Deployment auf AWS in eine MySQL Datenbank mit MySQL Engine deployed werden)

Da im Entwicklerteam der Bluewave Electronics niemand Erfahrung mit der Frontend-Entwicklung hat (alles Lappen), möchte die Firma Sie mit dem Auftrag, ein vollwertiges Frontend zu erstellen, beauftragen.

Sie können sich aus folgenden Frontend Technologien frei entscheiden:

- HTML (Pflicht)
- CSS (Pflicht)
- Bootstrap (Optional)
- Tailwind (Optional)
- JavaScript (Optional)
- React (Optional)
- Angular (Optional)
- Vue (Optional)

Ziel ist es, Ihnen die Maximale Freiheit in der Wahl der Technologie zu geben, dafür aber eine langfristige Zusammenarbeit mit uns zu realisieren.

Anforderungen Design:

- Seien Sie kreativ, aber bedenken Sie, dass weniger manchmal mehr ist. (Eventuelle Design Richtlinien, nicht zu viele Typografien, Farben usw…)
- Achten Sie auf eine ordentliche Platzverteilung der Elemente. Viel Whitespace ist willkommen, aber die Seite sollte nicht leer aussehen.

Anforderungen Schnittstellen:

- Versuchen Sie, Redundanz zu vermeiden. Ein modulares System ist performanter und sorgt für mehr Wartbarkeit und Erweiterbarkeit des Systems
- Dokumentieren Sie die Schnittstellen, falls Parameter übergeben werden, nutzen Sie vielleicht JS Docs, um eine Beschreibung der Parameter hinzuzufügen.
- Fangen Sie mögliche Fehler ab, wenn mal doch eine Anfrage nicht so läuft wie erwartet. Es wäre nicht gut, wenn daraufhin dass ganze Frontend abstürzt und der Besucher unseren Shop wieder verlässt.

Anforderungen der Seiten:

- Weiter unten finden Sie Bilder der bereitgestellten Schnittstellen von unserem Backend Team. Diese sollen ihnen bei der Orientierung helfen, die Seiten strukturiert und sinnvoll aufzuteilen.
- Erstellen Sie eine Hauptseite, auf der alle Produkte, die unser Backend Team im Datenbank-Prototyp für Sie bereitgestellt hat, in jeweils einer Karte darstellt.
  Jede Karte sollte den Namen, die Kurzbeschreibung, den Preis abbilden, sowie einen Link zur Produkt Detailseite
- Erstellen Sie eine Produkt Detailseite, auf der ein Besucher die Informationen über das Produkt finden kann, wie Preis, Produktbeschreibung, Lagerbestand.
- Erstellen Sie eine weitere Seite, um ein neues Produkt anzulegen, zu löschen oder zu bearbeiten.
  Zum anlegen eines neuen Produkts, senden Sie bitte den Produktnamen und den Preis an das Backend aus der Url in dem Bild unten.

Hinweise:

1. Seien Sie nicht perfektionistisch. Arbeiten Sie vom groben zum feinen. Designs ändern sich sowieso des öfteren und denken Sie daran, sie sind kein Designer und haben auch kein Design eines Designers vorliegen! Wir benötigen lediglich einen Prototypen ;)

2. Stressen Sie sich nicht zu sehr! Scheitern ist menschlich und wenn Sie sich mit einer Aufgabe überfordert fühlen, ist es vielleicht eine gute Idee, eine andere Baustelle abzuarbeiten!

3. Machen Sie sich bewusst, dass alle Anforderungen, die hier gestellt werden, im Unterricht behandelt wurden.Diesmal sind weder das IBB oder die IHK schuld! Aber der Dozent hat … Klappe!!!

4. Nutzen Sie das Internet mit allen Mitteln zur Informationsquelle. Hier geht es um bares Geld und wenn Sie in 5 Stunden nicht abliefern, sind wir Pleite. Fragen Sie auch bitte Ihre Kollegen, ob sie Ihnen weiterhelfen können, um Lösungsansätze zu finden. (Sowohl fachlich als auch finanziell, wenn Sie es nicht schaffen sollten!)

5. Suchen Sie im Internet nach Informationen, ob man Pinguine essen kann. Der Inhaber von Bluewave Electronics hat eine Vorliebe für exotische Speisen. Pinguinfußsüppchen…. Yammiii ….

Viel Erfolg!!!

Ach ja, die Bilder…

localhost:8000/api/products/

Django REST framework test

Product Api

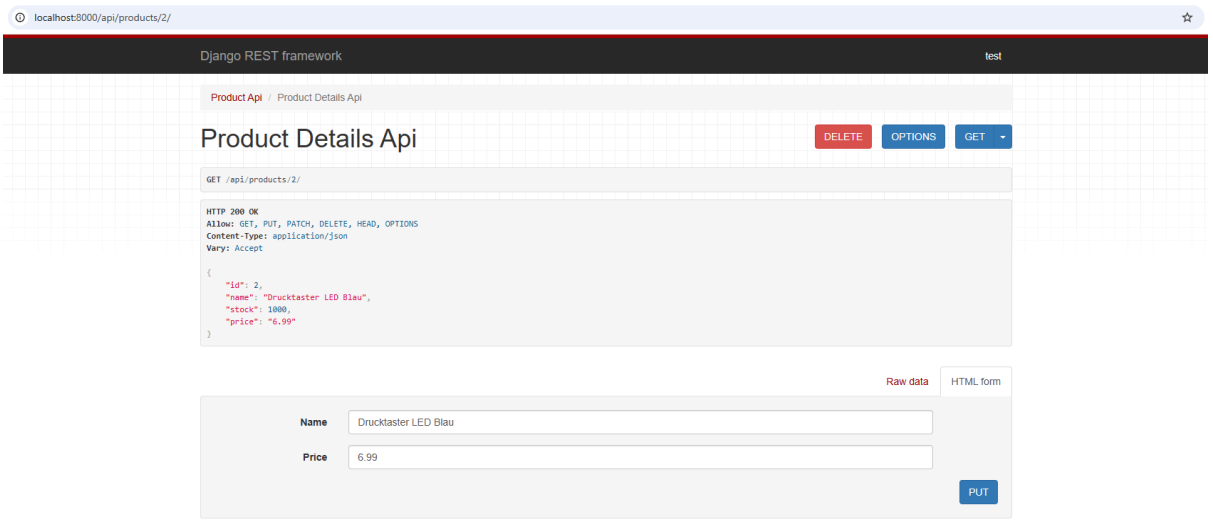
Product Api

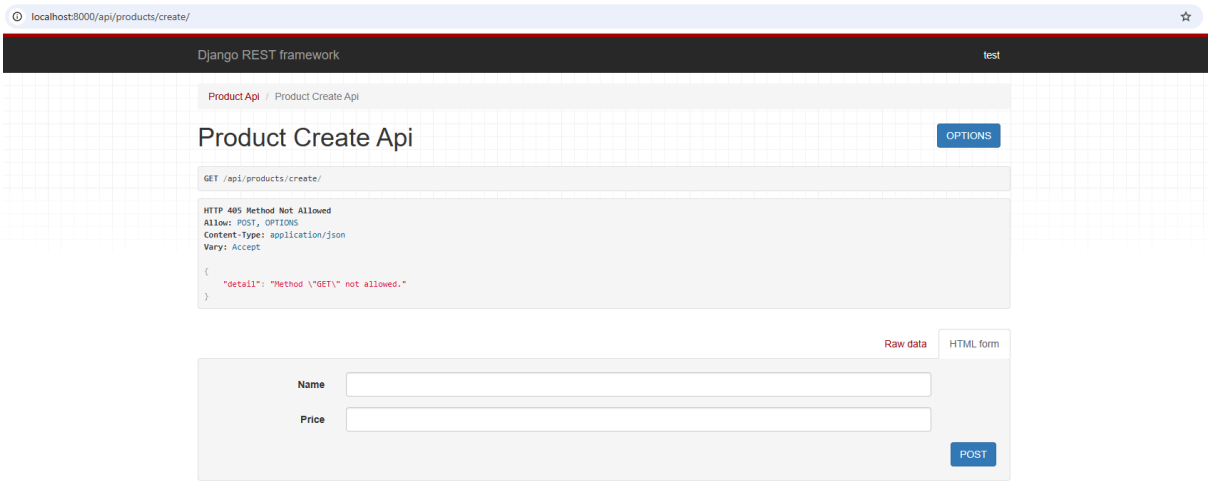
OPTIONS GET

```
GET /api/products/
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept
```


```
[
    {
        "id": 1,
        "name": "LED Display Blau",
        "stock": 100,
        "price": "7.99"
    },
    {
        "id": 2,
        "name": "Drucktaster LED Blau",
        "stock": 1000,
        "price": "6.99"
    },
    {
        "id": 3,
        "name": "Drucktaster LED Gelb",
        "stock": 655,
        "price": "6.99"
    },
    {
        "id": 4,
        "name": "Drucktaster LED Grün",
        "stock": 80,
        "price": "6.99"
    },
    {
        "id": 5,
        "name": "Drucktaster LED Rot",
        "stock": 215,
        "price": "6.99"
    }
]
```


localhost:8000/api/products/2/
Django REST framework
test
Product Api / Product Details Api
Product Details Api
DELETE
OPTIONS
GET
GET /api/products/2/
HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept
{
"id": 2,
"name": "Drucktaster LED Blau",
"stock": 1000,
"price": "6.99"
}
Raw data
HTML form
Name
Drucktaster LED Blau
Price
6.99
PUT


localhost:8000/api/products/create/
Django REST framework
test
Product Api / Product Create Api
Product Create Api
OPTIONS
GET /api/products/create/
HTTP 405 Method Not Allowed
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept
{
"detail": "Method \"GET\" not allowed."
}
Raw data
HTML form
Name
Price
POST