Panache-booth API

Authentifizierung

Alle unterstrichenen Anfragen benötigen einen *authorization Header*. Als Wert sind die E-Mail und das zugehörige Passwort via Basic-Auth zu übergeben

Schemas

Die folgenden Tabellen beschreiben die Struktur der von der API verwendeten Objekte.
Anfragen dürfen nur die **dick gedruckten Attribute** enthalten – alle anderen werden ignoriert.

Im Anhang befinden sich beispielhafte Darstellungen der POST HTTP-Request bodies.

User

Die hier ***kursiven und dick markierten Attribute*** werden nur benötigt, wenn der User als Vendor registriert wird.
Sonst werden sie ignoriert.

Attribut	Datentyp	Info
id	String	UUIDv4
userName	String	4 – 32 Zeichen
email	String	Max. 255 Zeichen und im E-Mail-Format, unique
password	String	6 – 12 Zeichen. Min. ein Großbuchstabe, min. eine Zahl
street	String	1 – 255 Zeichen
houseNumber	String	1 – 3 Zeichen
postcode	String	Genau 5 Zeichen
city	String	1 – 50 Zeichen
isVendor	Boolean	true oder false
iban	String	Max. 22 Zeichen
bic	String	Max. 11 Zeichen
shippingCost	Number	1 – 8 Stellen vor dem Komma, 0 – 2 Nachkommastellen (Optional)
shippingFreeFrom	Number	1 – 8 Stellen vor dem Komma, 0 – 2 Nachkommastellen (Optional)
createdAt	String	ISO 8601 Format
updatedAt	String	ISO 8601 Format

Product

Attribut	Datentyp	Info
id	String	UUIDv4
name	String	1 – 32 Zeichen
description	String	1 – 300 Zeichen
category	String	1 – 36 Zeichen
discount	Number	Zahl zwischen 0.00 – 0.99
price	Number	1 – 8 Stellen vor dem Komma, 0 – 2 Nachkommastellen (Optional)
vendorId	String	UUIDv4
purchases	Number	Ganzzahl (erhöht sich automatisch mit jedem Kauf)
inventory	Number	Ganzzahl größer oder gleich 0 -1 falls *inventory* nicht angegeben werden möchte
isVisible	Boolean	true oder false
createdAt	String	ISO 8601 Format
updatedAt	String	ISO 8601 Format
vendor	User Objekt	User Objekt (wird bei GET mit ausgegeben, sonst ignoriert)
quantity	Number	Ganzzahl (Benötigt für update Order, sonst ignoriert)
delivered	Boolean	true oder false (Benötigt für Order, sonst ignoriert)
Paid	Boolean	true oder false (Benötigt für Order, sonst ignoriert)

Order

Das ***kursiv und dick markierte*** Attribut ***product*** benötigt zwingend die productId und quantity des Products.
Genauer Aufbau: siehe Anhang → Order.

Attribut	Datentyp	Info
id	String	UUIDv4
userID	String	UUIDv4
price	Number	1 – 8 Stellen vor dem Komma, 0 – 2 Nachkommastellen (Optional)
createdAt	String	ISO 8601 Format
updated At	String	ISO 8601 Format
products	Products Objekt	Array von Products Objekten mit einem oder mehreren Products Quantity und ProductId werden bei ***products*** benötigt!
User	User Objekt	User Object des bestellenden Users (Wird bei GET ausgegeben, sonst ignoriert)

OrderProduct

Auf das Schema OrderProduct kann nicht direkt über die API zugegriffen werden.
Das Schema wird nur benötigt, um die n:m-Beziehung zwischen Order und Product aufzulösen.

Attribut	Datentyp	Info
orderId	String	UUIDv4
productId	String	UUIDv4
quantity	Number	Ganzzahl größer als 0
deliverd	Boolean	true oder false
paid	Boolean	true oder false
createdAt	String	ISO 8601 Format
updated At	String	ISO 8601 Format
priceProduct	Number	1 – 8 Stellen vor dem Komma, 0 – 2 Nachkommastellen (Optional)
discountProduct	Number	Zahl zwischen 0.00 – 0.99
vendorShippingCost	Number	1 – 8 Stellen vor dem Komma, 0 – 2 Nachkommastellen (Optional)
vendorShippingFreeFrom	Number	1 – 8 Stellen vor dem Komma, 0 – 2 Nachkommastellen (Optional)

Requests

Im Folgenden werden alle verfügbaren API Requests, deren Request Bodies und deren Responses beschrieben. Pfade mit *[ ]* müssen mit entsprechenden Werten ersetzt werden.

Alle API Request besitzen zusätzlich zu den angegebenen *400* Fehlercodes einen *500* Fehlercode mit der Nachricht *Internal server error!*.
Dies wird ausgegeben, falls es einen Serverfehler geben sollte. Darunter fällt beispielsweise ein Datenbankfehler.

Die *400* Fehlercodes enthalten eine Nachricht, die das Problem entsprechend beschreibt.

Standardpfad: http://localhost:3000/api

User

Die Rückgabe eines User Objekts erfolgt grundsätzlich ohne das gespeicherte Passwort.

Methode	Pfad	Request Body	Response Body
GET	/user	-	200: Array von User Objekten
GET	/user/[userId]	-	200: Ein User Objekt 400: User nicht gefunden
GET	/user/login	-	200: Login-Daten korrekt 401: Login-Daten inkorrekt
POST	/user	User Objekt (Vendor Bedingungen beachten)	200: User Objekt (ohne Passwort) 400: Falsche Syntax 400: Doppelte E-Mail
PUT	/user/[userId]	User Objekt (Vendor Bedingungen beachten)	200: User Objekt (ohne Passwort) 400: Syntax Fehler 400: Doppelte E-Mail 401: Unautorisiert
DELETE	/user/[userId]	-	200: falls User gelöscht 400: User nicht gefunden 401: Unautorisiert

Product

Methode	Pfad	Request Body	Response Body
GET	/product	-	200: Array von Products, zusätzlich der Vendor des jeweiligen Products als User Objekt
GET	/product/[productId]	-	200: Ein Product Objekt, zusätzlich der Vendor des Products als User Objekt 400: Product nicht gefunden
POST	/product	Product Objekt	200: Product Objekt 400: Syntaxfehler 401: Unautorisiert
PUT	/product/[productId]	Product Objekt	200: Product Objekt 400: Syntaxfehler 401: Unautorisiert
DELETE	/product/[productId]	-	200: falls gelöscht 400: Product nicht gefunden 401: Unautorisiert

Order

Der als **dick markierte Pfad** benötigt zusätzlich zu der userId noch den query Parameter isVendor.
Dieser Parameter kann entweder true oder false sein.

Mit Order PUT kann nur festgelegt werden ob die jeweiligen Produkte durch den Vendor verschickt worden sind.
Die Order an sich kann nachträglich **nicht verändert werden!**
Ebenso kann eine erstellte Order im Nachgang **nicht mehr gelöscht werden!**

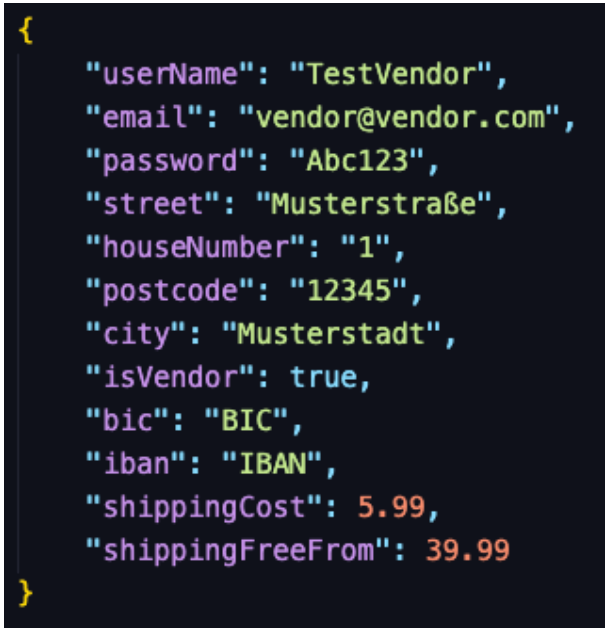
Methode	Pfad	Request Body	Response Body
GET	/order	-	200: Array von Order Objekten
GET	**/order/[userId]?isVendor=true/false**	-	200: isVendor=true → Array von Order Objekten des Vendors, mitsamt einem Array der darin enthaltenen Products Objekten des Vendors + User Objekt des Käufers und dem User Objekt des Bestellenden Users 200: isVendor=false → Array von Order Objekten des Users, mitsamt einem Array der darin enthaltenen Products Objekten des Users und dem User Objekt des Bestellenden Users 401: Unautorisiert
POST	/order	Order Objekt + Array von Product Objekten	200: Order Objekt 400: Syntaxfehler 401: Unautorisiert
PUT	/order /[orderId]	Array von Products (Form siehe Anhang)	200: Order Objekt 400: Syntaxfehler 401: Unautorisiert

Anhang

User

Beispielhafter Aufbau des POST Requests.

```
{
    "userName": "TestUser",
    "email": "user@user.com",
    "password": "Abc123",
    "street": "Musterstraße",
    "houseNumber": "1",
    "postcode": "12345",
    "city": "Musterstadt"
}
```



```
{
    "userName": "TestVendor",
    "email": "vendor@vendor.com",
    "password": "Abc123",
    "street": "Musterstraße",
    "houseNumber": "1",
    "postcode": "12345",
    "city": "Musterstadt",
    "isVendor": true,
    "bic": "BIC",
    "iban": "IBAN",
    "shippingCost": 5.99,
    "shippingFreeFrom": 39.99
}
```

Product

Beispielhafter Aufbau des POST Requests.

```
{
    "name": "Artikelname",
    "description": "Beschreibung",
    "discount": 0,
    "category": "Kategorie",
    "vendorId": "VendorId",
    "purchases": 5,
    "inventory": 2,
    "isVisible": true,
    "price": 29.99
}
```

Order

Beispielhafter Aufbau des POST Requests.