

Individuelle Praktische Arbeit

Bestellungs-Übersicht | API-Backend in Laravel (PHP-Framework)

Dieses Dokument enthält den Bericht zur individuellen praktischen Arbeit von Julien Rädler.

Inhaltsverzeichnis

AUFGABENSTELLUNG	5
TITEL DER ARBEIT	5
AUSGANGSLAGE	5
DETAILLIERTE AUFGABENSTELLUNG	5
MITTELN UND METHODEN	7
NEUE LERNINHALTE	7
ARBEITEN IN DEN LETZTEN 6 MONATEN	7
PROJEKTAUFBAUORGANISATION	8
PROJEKTORGANISATION	8
AUSFÜHRUNGSZEITRAUM	8
TERMINE	8
INVOLVIerte PERSONEN	8
VORKENNTNISSE	9
VORARBEITEN	9
BENÜTZTE FIRMENSTANDARDS	9
VERSIONIERUNG UND DATENSICHERHEIT	9
ARBEITSPROTOKOLL	14
MONTAG, 13. MÄRZ 2023	14
DIENSTAG, 14. MÄRZ 2023	15
MITTWOCH, 15. MÄRZ 2023	16
FREITAG, 17. MÄRZ 2023	17
MONTAG, 20. MÄRZ 2023	18
DIENSTAG, 21. MÄRZ 2023	19
MITTWOCH, 22. MÄRZ 2023	20
FREITAG, 24. MÄRZ 2023	21
MONTAG, 27. MÄRZ 2023	22
DIENSTAG, 28. MÄRZ 2023	23
KURZFASSUNG	24
AUSGANGSSITUATION	24
UMSETZUNG	24
ERGEBNIS	24
INFORMIEREN	25
AUSGANGSLAGE	25
ABKLÄRUNGEN	25
VERSTANDENE AUFGABENSTELLUNG UND ZIEL DER ARBEIT	25
VERFEINERUNG DES AUFTRAGES	25

PROJEKTUMFELD UND SYSTEMGRENZEN.....	25
PLANEN	26
VERWENDETE PROJEKT MANAGEMENT METHODE.....	26
VERSIONIERUNG UND DATENSICHERHEIT	26
PRIORISIERUNG DER TÄTIGKEITEN	26
KERNFEATURE LARAVEL API	26
KERNFEATURE SWAGGER GUI	30
GEPLANTES VORGEHEN FÜR DIE QUALITÄTSSICHERUNG	32
ANMERKUNGEN ZUM ZEITPLAN	32
BEGRÜNDUNG FÜR ABWEICHUNGEN ZUM ZEITPLAN.....	32
ENTSCHEIDEN	33
VARIANTE BILDER ALS BLOB SPEICHERN	33
VARIANTE BILDER IN DATEISYSTEM SPEICHERN	33
KURZBESCHREIBUNG DER ENTSCHEIDUNGSKRITERIEN.....	33
ENTSCHEIDUNGSMATRIX	33
VARIANTE SHOP-NAME DIREKTER VERGLEICH	34
VARIANTE VERGLEICH MIT PHP-METHODE	34
VARIANTE VERGLEICH MIT SCOUT	34
KURZBESCHREIBUNG DER ENTSCHEIDUNGSKRITERIEN.....	34
ENTSCHEIDUNGSMATRIX	34
REALISIEREN	35
ABBILDUNG DES GESAMTSYSTEMS	35
INTERAKTIONEN ZWISCHEN DEN TEILSYSTEMEN	36
VORGEHENSWEISE	36
ANBINDUNG AN DIE DATENBANK.....	36
IMPLEMENTIERUNG DES KERNFEATURES LARAVEL-API.....	37
IMPLEMENTIERUNG DES KERNFEATURES SWAGGER	39
KONTROLLIEREN.....	42
BESCHREIBUNG DER RANDBEDINGUNGEN / TESTANLAGE (UMFELD)	42
EINGESETZTE TESTMITTEL UND -METHODEN.....	42
BESCHREIBUNG DER TESTSZENARIEN	42
GENERELLE ANFORDERUNGEN	42
EINGESETZTE TESTMITTEL.....	82
RAHMENBEDINGUNGEN.....	82
AUSWERTEN.....	83
REFLEXION DER VORGEHENSWEISE.....	83
BEWERTUNG DES PRODUKTES	83
PERSÖNLICHES SCHLUSSWORT UND BILANZ	84

GLOSSAR	85
QUELLENVERZEICHNIS.....	87
LITERATURVERZEICHNIS	87
ABBILDUNGSVERZEICHNIS.....	87
TABELLENVERZEICHNIS.....	87
ANHANG.....	89
PROJEKTJOURNAL	90
GESPRÄCHSPROTOKOLL VOM 20. MÄRZ 2023	90
GESPRÄCHSPROTOKOLL VOM 27. MÄRZ 2023	90
ERSTER EXPERTENBESUCH	91
ZWEITER EXPERTENBESUCH	92
CODE INFO	94
FILENAME	94
ZUSÄTZLICHE MANUALS, SKRIPTS UND WEITERES	95
HANDBUCH LARAVEL.....	95
HANDBUCH SWAGGER FÜR LARAVEL	95

Teil 1: Umfeld und Ablauf

Aufgabenstellung

Titel der Arbeit

Bestellungs-Übersicht | API-Backend in Laravel (PHP-Framework).¹

Ausgangslage

Wir bei der twofold academy AG bieten Lernenden auf dem Autismus-Spektrum die Möglichkeit eine Lehre zu machen und fit für den ersten Arbeitsmarkt zu werden. Durch unser spezielles Setting und die Finanzierung durch die IV dürfen wir keine gewinnbringenden, externen Aufträge annehmen, weswegen wir auf interne Projekte bauen.

Bei uns in der Firma werden von verschiedenen Personen Bestellungen in diversen Webshops getätigt. Da nicht für jeden Mitarbeiter ein eigenes Login bei diesen Webshops existiert, fehlt eine Möglichkeit diese Bestellungen zuzuordnen, nachzuverfolgen und zu organisieren. Es soll eine Bestellungs-Übersicht erstellt werden, wo oft bestellte Produkte erfasst und zu Bestellungen hinzugefügt werden können. In dieser IPA soll jetzt der erste Schritt dieser Bestellungs-Übersicht, das API-Backend, erstellt werden. Aufbauend auf diese Arbeit wird dann später noch ein Frontend erstellt, was aber nicht zu dieser IPA gehört.²

Detaillierte Aufgabenstellung

Es soll ein API-Backend aufgebaut werden, womit Produkte verwaltet werden können. Die Produkte sollen zudem zu Bestellungen zusammengefasst werden können.

- Es ist ein API-Backend mit dem Laravel Framework in PHP umzusetzen
- Die Daten sollen in einer geeigneten Struktur in einer Datenbank abgelegt werden: Die 3. Normalform muss erreicht sein und die Spalten müssen mit sinnvollen Datentypen definiert werden
- Die Datenbank-Struktur wird mithilfe einer Migration automatisch aufgebaut
- Die einzelnen Tabellen werden mit Model-Klassen in Laravel abgebildet
- Für die verschiedenen Inhalte werden Routen bereitgestellt
- Die Routen werden mit den üblichen Verben angesprochen (Create = POST, Read = GET, Update = PUT/PATCH, Delete = DELETE)
- Die API-Routen sollen in einer geeigneten Form dokumentiert werden für die künftigen Frontend-Entwickler (Swagger o.ä.)
- Die Kommunikation mit der API läuft über JSON
- Anfragen werden in Request-Klassen validiert und aufbereitet
- Antworten werden in Ressourcen-Klassen aufbereitet und zurückgesendet
- Fehler in den Eingaben werden dem Nutzer verständlich, auf Deutsch, zurückgeschickt
- Die Routen und Inhalte sind geschützt, sodass nur eingeloggte User Zugang haben
- Die Routen sollen händisch getestet und in einem Testprotokoll aufgeführt werden
- Das Projekt soll lokal laufen und muss noch nicht veröffentlicht werden

¹ Aufgabenstellung Original gemäss Eingabe aus PkOrg

² Aufgabenstellung Original gemäss Eingabe aus PkOrg

- Das Projekt muss sauber, nachvollziehbar und leicht erweiterbar sein: Die Konventionen, die in der Laravel-Dokumentation festgehalten sind, sollen eingehalten werden, Funktionen/Variablen sollen selbsterklärende Namen haben und der gesamte Code soll einheitlich formatiert/eingerückt sein

Endpoints:

Alle Angaben sind Pflichtangaben, ausser mit "optional" spezifizierte.

Users:

- User sollen sich registrieren und einloggen können (ohne Mail-Validierung)
- Users beinhalten:
 - Username
 - E-Mail
 - Passwort

Orders:

- Bestellungen sollen CRUD-Möglichkeiten haben
- Bestellungen beinhalten:
 - Bestelldatum
 - User, der die Bestellung eingetragen hat (siehe "Users" oben)
 - Produkte (siehe "Products" unten)
 - Erhalt Datum pro Produkt
 - Anzahl pro Produkt
 - Preis pro Produkt zum Bestellzeitpunkt
 - Totalpreis pro Produkt für gesamte Anzahl
 - Totalpreis Bestellung
 - Optionale Bemerkungen zur Bestellung
- Bestellungen können sortiert und gefiltert werden
 - Sortierungen:
Bestelldatum, Status (Offen/Abgeschlossen)
 - Filterung:
Status (Offen/Abgeschlossen)

(Der Status der Bestellung ist "Abgeschlossen", sobald alle Produkte ein Erhalt Datum eingetragen haben. Bis dahin bleibt dieser "Offen". Dieser Status kann dynamisch aus der Datenbank ausgelesen und muss in keiner Tabelle erfasst werden. Das Erhalt-Datum wird von der jeweiligen Person aktualisiert, die zuständig für die Entgegennahme von Paketen ist.)

Products:

- Produkte sollen CRUD-Möglichkeiten haben
- Wird ein Produkt in mehreren Shops bestellt, muss es mehrfach erstellt werden
- Produkte beinhalten:
 - Name
 - optionales Bild
 - Webshop
 - Link zu Produkt

- Produkte können sortiert und gefiltert werden
- Sortierungen:
Name, Webshop
- Filterung:
Webshop

Die Einkäufe werden in den jeweiligen Webshops getätigt und können danach mithilfe dieser IPA in Bestellungen zusammengefasst eingetragen und somit organisiert und nachvollzogen werden. Diese IPA dient dazu, den Überblick zu bewahren, was, wann, von wem bestellt und was bereits erhalten wurde, ohne sich in alle Webshops einloggen zu müssen.

Erwartet wird ein funktionierendes API-Backend, aufgebaut in Laravel, mit welchem bspw. über Postman/Insomnia kommuniziert werden kann. Es soll eine funktionierende Registrierung eines Nutzers möglich sein, mit anschliessendem Login. Die Authentifizierung soll via Bearer-Token funktionieren. Der eingeloggte Nutzer sollte dann die Möglichkeit haben, Produkte zu erstellen, bearbeiten und zu löschen. Zudem soll der Nutzer die Möglichkeit haben, Produkte einer Bestellung zuzuweisen und mit allen Inhalten zu speichern, bearbeiten und zu löschen. Inhalte sollen mit der gewünschten Sortierung und Filterung abgefragt werden können.³

Mitteln und Methoden

XAMPP, Laravel, PHP, Postman/Insomnia.⁴

Neue Lerninhalte

Keine.⁵

Arbeiten in den letzten 6 Monaten

Ausschliesslich Aufgaben und Projekte in Laravel.⁶

³ Aufgabenstellung Original gemäss Eingabe aus PkOrg

⁴ Aufgabenstellung Original gemäss Eingabe aus PkOrg

⁵ Aufgabenstellung Original gemäss Eingabe aus PkOrg

⁶ Aufgabenstellung Original gemäss Eingabe aus PkOrg

Projektaufbauorganisation

Projektorganisation

Auszubildender	Lehrbetrieb
Julien Rädler Im Dreispitz 235 8050, Zürich	twofold academy AG Thurgauerstrasse 54 8050 Zürich

Tabelle 1 - Adressinformation Auszubildender

Ausführungszeitraum

Mo 13. März 2023	Fr 17. März 2023	Mi 22. März 2023	Di 28. März 2023
Di 14. März 2023	Mo 20. März 2023	Fr 24. März 2023	
Mi 15. März 2023	Di 21. März 2023	Mo 27. März 2023	

Tabelle 2 – Ausführungszeitraum

Termine

Was?	Wann?
1. Expertenbesuch	15.03.2023 16:00
2. Expertenbesuch	22.03.2023 13:30
Präsentation, Demonstration, Fachgespräch	05.04.2023 13:30

Tabelle 3 – Termine

Involvierte Personen

Kontaktinformation	Rolle, Aufgabe, Verantwortung
Hansruedi Menzi hasu.menzi@bewegdi.com 079 242 08 37	Rolle: Hauptexperte Aufgabe: Hauptexperte Verantwortung: Hauptexperte
Sascha Gysel sascha@gysel.biz +41 79 240 45 23	Rolle: 2. Experte Aufgabe: 2. Experte Verantwortung: 2. Experte
Domenik Hofer domenik.hofer@twofold.swiss 044 533 42 38	Rolle: Verantwortliche Fachkraft Aufgabe: Verantwortliche Fachkraft Verantwortung: Verantwortliche Fachkraft
Domenik Hofer domenik.hofer@twofold.swiss 044 533 42 38	Rolle: Berufsbildner Aufgabe: Berufsbildner Verantwortung: Berufsbildner
Julien Rädler julien.raedler@twofold.swiss 076 526 25 52	Rolle: IPA-Prüfungskandidat Aufgabe: IPA-Prüfungskandidat Verantwortung: IPA-Prüfungskandidat

Tabelle 4 - Involvierte Personen

Vorkenntnisse

Laravel, PHP

Laravel API Authentifizierung via Bearer-Token

Ganze Lehrzeit verwendet.⁷

Vorarbeiten

Keine.⁸

Benützte Firmenstandards

Dokumentation:

Vorlage der twofold AG.

Coding-Vorgaben:

Github zur Speicherung des Projektes online.⁹

Versionierung und Datensicherheit

Zur Versionierung wurde GitHub verwendet. Dabei benutzte ich sowohl meinen Firmenaccount (joinbean), als auch meinen Schulaccount (bbw19).

commit ea305c022e7140c646fd9e8935ac507f199a88ef

Author: joinbean <julien.raedler@twofold.swiss>

Date: Tue Mar 28 12:03:06 2023 +0200

finishing documentation

commit 84dcad0aae8720c3cee85a9642f09c5c83d42320

Author: joinbean <julien.raedler@twofold.swiss>

Date: Mon Mar 27 16:48:16 2023 +0200

daily protocol added

commit 7ee2397692aeeb666a4648df339f5d756cbd6840

Author: joinbean <julien.raedler@twofold.swiss>

Date: Mon Mar 27 16:42:14 2023 +0200

table and graphic names in documentation

⁷ Aufgabenstellung Original gemäss Eingabe aus PkOrg

⁸ Aufgabenstellung Original gemäss Eingabe aus PkOrg

⁹ Aufgabenstellung Original gemäss Eingabe aus PkOrg

commit 7bfcec422ac49da96659e748ffea6254f9cb4c47

Author: joinbean <julien.raedler@twofold.swiss>

Date: Mon Mar 27 11:59:51 2023 +0200

document implementation

commit f52675114f792ffcd173faa3c4bb2425e9541e0

Author: bbw19 <71121888+bbw19@users.noreply.github.com>

Date: Sun Mar 26 12:21:22 2023 +0200

finished test protocol

commit e93f23eb36798044a8eb6bd0163a94e4df54cc8f

Author: bbw19 <71121888+bbw19@users.noreply.github.com>

Date: Sat Mar 25 12:59:07 2023 +0100

prodgress test protocol

commit 91770feeb05517bb89d7c509346957bf3e19be70

Merge: 158a7b9 f4cf666

Author: bbw19 <71121888+bbw19@users.noreply.github.com>

Date: Sat Mar 25 11:14:10 2023 +0100

Merge branch 'main' of <https://github.com/joinbean/IPA>

commit 158a7b9f22cc64071a6b3f3dbc4dd1ff5527b98e

Author: bbw19 <71121888+bbw19@users.noreply.github.com>

Date: Sat Mar 25 11:12:09 2023 +0100

set up project on laptop

commit f4cf666f3669ff1e00ce6b2365990b2ec58beec0

Author: joinbean <julien.raedler@twofold.swiss>

Date: Fri Mar 24 16:52:35 2023 +0100

started documentation of tests

commit b15f2169e80de5214603bf8441152e7d629c8ff1

Author: joinbean <julien.raedler@twofold.swiss>

Date: Fri Mar 24 11:35:11 2023 +0100

finished swagger

commit 7a8f1123d30a10457117ffd8bebb0dd0ded46ed6

Author: joinbean <julien.raedler@twofold.swiss>

Date: Wed Mar 22 16:51:56 2023 +0100

second expert visit

commit c6c7cdfa4c85e0cf942797ae244b56a5634a75d8

Author: joinbean <julien.raedler@twofold.swiss>

Date: Wed Mar 22 11:58:58 2023 +0100

work on swagger

commit c201652292923d58156736cdcc873c8d440bdf9f

Author: joinbean <julien.raedler@twofold.swiss>

Date: Tue Mar 21 16:52:37 2023 +0100

create controller, request and resource

commit c9dfe4580ba0809717f97bee000f8930bd219866

Author: joinbean <julien.raedler@twofold.swiss>

Date: Tue Mar 21 11:40:00 2023 +0100

controller for shop and product

commit bc4fe2dbd5edf3c531f9c52913faf072c3d45d3a

Author: joinbean <julien.raedler@twofold.swiss>

Date: Mon Mar 20 16:42:10 2023 +0100

finish test concept

commit c4058c25ef1af7ac98361f3cab8967e9a0dca5af

Author: joinbean <julien.raedler@twofold.swiss>

Date: Mon Mar 20 11:33:26 2023 +0100

update test protocol

commit 1d1c92c63557e124e05dea9ffd961c6c643527ee

Author: joinbean <julien.raedler@twofold.swiss>

Date: Fri Mar 17 16:39:39 2023 +0100

test concept documentation

commit 77ff47418f8d298803a29c05274d639812c56fb6

Author: joinbean <julien.raedler@twofold.swiss>

Date: Fri Mar 17 12:50:42 2023 +0100

update swagger

commit 88e266dd1f9c023b6c6ffe4c6b889334a0621f4b

Author: joinbean <julien.raedler@twofold.swiss>

Date: Wed Mar 15 15:47:08 2023 +0100

add laravel model to documentation

commit 99f303ff0f309025f7b0bea2c1cbe85aaf53e64c

Author: joinbean <julien.raedler@twofold.swiss>

Date: Wed Mar 15 14:17:45 2023 +0100

class diagramm

commit 64595526246ea8f8fff42eb598b9fcce3793691c

Author: joinbean <julien.raedler@twofold.swiss>

Date: Tue Mar 14 16:44:00 2023 +0100

start planing phase

commit cafc539df5c3630feac13abc9be064406ca33c4d

Author: joinbean <julien.raedler@twofold.swiss>

Date: Tue Mar 14 12:24:48 2023 +0100

info phase documentation and update time table

commit 2d0476f512e344bcde3eb6f16c1fd1aeb6977eee

Author: joinbean <julien.raedler@twofold.swiss>

Date: Mon Mar 13 16:28:57 2023 +0100

create documentation and create time table

commit d2687d17ab0594fd35101bb19cb8c1ff8d12f760

Author: joinbean <julien.raedler@twofold.swiss>

Date: Mon Mar 13 14:04:51 2023 +0100

first commit

Zeitplan

plan		
Bezeichnung	SOLL Aufwand	IST Aufwand
Täglichejobs	6	6
Arbeitsjournal führen	6	6
Vorbereitung	12	10
Zeitplan erstellen	6	6
GIT-Projekt erstellen	2	1
IPA-Bericht Vorlage erstellen	2	1
IPA-Vorbereitung dokumentieren	2	2
Informationen	2	3
Informationen sammeln	2	3
Planen	12	25
Prioritätentabellen erstellen	2	1
Aktivitätsdiagramm erstellen	2	2
Klassendiagramm erstellen	2	2
Laravel-Aufbaudiagramm erstellen	2	4
Mockups erstellen	2	5
Testkonzept erstellen	2	11
Entscheiden	3	2
Entscheidungsmatrix erstellen	2	1
Entscheidungen treffen	1	1
Realisieren	19	15
Laravel-Projekt vorbereiten	0.5	0.5
API-Routen erstellen	0.5	0.5
Controller erstellen	2	1.5
Requests erstellen	2	1.5
Ressourcen erstellen	2	1.5
Validierungen erstellen	2	1.5
Swagger Aufsetzen	6	4
Authentifizierung implementieren und in Swagger anwenden	4	4
Kontrollieren	4	8
Tests durchführen	4	8
Auswerten	4	2
Arbeit auswerten und Reflexion schreiben	4	2
Dokumentieren	18	14
Implementation dokumentieren	6	2
Dokumentation abschliessen	6	8
Pufferzeit	6	4
Gesamtstunden	80	85

Arbeitsprotokoll

Montag, 13. März 2023

Tagesziele gemäss Zeitplan	Zeitplan erstellen, GIT-Projekt erstellen
Ungeplante Arbeiten / geleistete Überstunden	Keine
Erreichte Ziele, Erfolgserlebnisse	Fertigstellen des Zeitplans und aufsetzen von Repository
Herausforderungen Probleme	Ich hatte ein kleines Problem mit dem Zeitplan. Ich wusste nicht genau wie ich die Zeit (15 min), für das Protokollschreiben, im Zeitplan repräsentieren soll.
Lösungen	Ich habe beschlossen, die Zeitblöcke nicht zu addieren.
Durchgeführte Tests	Keine
Wissensbeschaffung	Pkorg Aufgabenstellung
Beanspruchte Hilfeleistung	Keine
Vergleich mit dem Soll-Zeitplan	Stimmt überein.
Persönliche Tagesreflexion	Ich hatte etwas mit Excel zu kämpfen. Das war jedoch zu erwarten, weshalb ich etwas mehr Zeit für den Zeitplan eingerechnet habe. Ich konnte gut in die IPA starten und liege genau im Zeitrahmen.

Tabelle 5 - Arbeitsprotokoll Tag 1

Dienstag, 14. März 2023

Tagesziele gemäss Zeitplan	IPA-Bericht erstellen, Vorbereitung dokumentieren, Informationen sammeln, Prioritätentabellen erstellen
Ungeplante Arbeiten / geleistete Überstunden	Keine
Erreichte Ziele, Erfolgserlebnisse	Ich konnte die Informationsphase abschliessen und bereits mit der Planungsphase beginnen.
Herausforderungen Probleme	Ich hatte Probleme mit draw.io. Ich will das Klassendiagramm mit SQL erstellen, habe aber Probleme beim Import.
Lösungen	Ich muss es von Hand auf der Seite eingeben.
Durchgeführte Tests	Keine
Wissensbeschaffung	Pkorg, Laravel Dokumentation
Beanspruchte Hilfeleistung	Keine
Vergleich mit dem Soll-Zeitplan	Leichter Vorsprung
Persönliche Tagesreflexion	Ich bin gut vorangekommen, glaube aber, dass ich meinen Fortschritt von heute noch einmal Überarbeiten muss.

Tabelle 6 - Arbeitsprotokoll Tag 2

Mittwoch, 15. März 2023

Tagesziele gemäss Zeitplan	Aktivitätsdiagramme erstellen, Klassendiagramm erstellen, Aufbaudiagramm erstellen, Expertenbesuch
Ungeplante Arbeiten / geleistete Überstunden	PDF von Zeitplan erstellen
Erreichte Ziele, Erfolgserlebnisse	Ich konnte das Aufbaudiagramm erstellen und den ersten Expertenbesuch durchführen.
Herausforderungen Probleme	Ich habe Probleme, den Zeitplan als PDF sauber darzustellen und konnte das deshalb noch nicht beenden.
Lösungen	Leider muss ich wohl den Zeitplan erneut machen.
Durchgeführte Tests	Keine
Wissensbeschaffung	Laravel-Webseite
Beanspruchte Hilfeleistung	Fragen an den Experten.
Vergleich mit dem Soll-Zeitplan	Ich liege genau im Zeitplan
Persönliche Tagesreflexion	Die Diagramme für den Aufbau von Laravel sind sehr gut herausgekommen. Der Zeitplan scheint so weit gut aufzugehen.

Tabelle 7 - Arbeitsprotokoll Tag 3

Freitag, 17. März 2023

Tagesziele gemäss Zeitplan	Mockups erstellen, Testkonzept erstellen, Entscheidungsmatrix erstellen, Entscheidungen treffen, Laravel-Projekt vorbereiten, API-Routen erstellen
Ungeplante Arbeiten / geleistete Überstunden	Keine
Erreichte Ziele, Erfolgserlebnisse	Mockups erstellt mit einigen erklärenden Diagrammen extra.
Herausforderungen Probleme	Testkonzept war etwas unübersichtlich, was aus dem Auftrag geprüft werden soll.
Lösungen	Ich konnte die prüfbaren Vorgaben herausfinden.
Durchgeführte Tests	Keine
Wissensbeschaffung	Laravel-Webseite, DarkOnLine Github
Beanspruchte Hilfeleistung	Keine
Vergleich mit dem Soll-Zeitplan	Rückstand von etwa vier bis sechs Stunden.
Persönliche Tagesreflexion	Ich habe das Erstellen der Dokumentation für die Mockup-Phase etwas unterschätzt. Stark unterschätzt habe ich das Erstellen des Testkonzepts.

Tabelle 8 - Arbeitsprotokoll Tag 4

Montag, 20. März 2023

Tagesziele gemäss Zeitplan	Controller erstellen, Requests erstellen, Ressourcen erstellen, Validierung erstellen
Ungeplante Arbeiten / geleistete Überstunden	Keine
Erreichte Ziele, Erfolgserlebnisse	Ich konnte das Testkonzept endlich abschliessen.
Herausforderungen Probleme	Ich war mir nicht sicher, wie ausführlich das Testkonzept sein sollte.
Lösungen	Domenik konnte mir Input geben.
Durchgeführte Tests	Keine
Wissensbeschaffung	Keine
Beanspruchte Hilfeleistung	Domenik Hofer, mein Berufsbildner
Vergleich mit dem Soll-Zeitplan	Ich liege substanziell hinter dem Zeitplan, nämlich eineinhalb Tage.
Persönliche Tagesreflexion	Ich hatte Schwierigkeiten beim Erstellen des Zeitplans, weil ich nicht ganz wusste, wie ich ihn aufbauen sollte. Glücklicherweise konnte ich den Zeitplan heute fertigstellen.

Tabelle 9 - Arbeitsprotokoll Tag 5

Dienstag, 21. März 2023

Tagesziele gemäss Zeitplan	Swagger aufsetzen, Authentifizierung implementieren und in Swagger anwenden
Ungeplante Arbeiten / geleistete Überstunden	Keine
Erreichte Ziele, Erfolgserlebnisse	Laravel-Projekt vorbereiten, API-Routen erstellen, Controller erstellen, Requests erstellen, Ressourcen erstellen, Validierung erstellen
Herausforderungen Probleme	Ich hatte ein paar Probleme die Validierung richtig zu definieren.
Lösungen	Mit Hilfe der Laravel-Dokumentation habe ich schlussendlich den korrekten Aufbau verstanden.
Durchgeführte Tests	Keine
Wissensbeschaffung	Laravel-Dokumentation, laraveldaily.com
Beanspruchte Hilfeleistung	Keine
Vergleich mit dem Soll-Zeitplan	Ich konnte gute Fortschritte machen, bin aber nichtsdestotrotz weiter im Verzug. Ich bin zirka ein Tag hinter dem Zeitplan.
Persönliche Tagesreflexion	Normalerweise mache ich zuerst ein Controller ohne Requests und Resources, um die Funktionalität zu gewährleisten. Um Zeit zu sparen, habe ich in diesem Projekt direkt die ganze Struktur aufgebaut.

Tabelle 10 - Arbeitsprotokoll Tag 6

Mittwoch, 22. März 2023

Tagesziele gemäss Zeitplan	Authentifizierung implementieren und in Swagger anwenden, Expertenbesuch
Ungeplante Arbeiten / geleistete Überstunden	Keine
Erreichte Ziele, Erfolgserlebnisse	Ich konnte den zweiten Expertenbesuch erfolgreich abschliessen.
Herausforderungen Probleme	Ich hatte ein Problem Multipart-Formdata per PUT-Befehl in Swagger zu versenden.
Lösungen	Im Body des Request die PUT-Methode mitgeben.
Durchgeführte Tests	Keine
Wissensbeschaffung	Laravel-Dokumentation, L5-Swagger-Dokumentation
Beanspruchte Hilfeleistung	Domenik Hofer
Vergleich mit dem Soll-Zeitplan	Obwohl ich weiter Zeit gutmache, liege ich weiterhin etwas hinter dem Soll-Zustand. Ich bin zirka vier bis sechs Stunden im Rückstand.
Persönliche Tagesreflexion	Beim Implementieren von Swagger traten einige kleine Probleme auf. Ich konnte aber trotzdem gute Fortschritte machen, unter anderem weil die Autorisierung problemlos war.

Tabelle 11 - Arbeitsprotokoll Tag 7

Freitag, 24. März 2023

Tagesziele gemäss Zeitplan	Tests durchführen, Arbeit auswerten und Reflexion schreiben, Implementation dokumentieren
Ungeplante Arbeiten / geleistete Überstunden	Keine
Erreichte Ziele, Erfolgserlebnisse	Ich konnte die Punkte Swagger aufsetzen und Authentifizierung implementieren und in Swagger anwenden, abschliessen. Der Realisierungsteil ist damit beendet.
Herausforderungen Probleme	Einige Swagger Annotationen sind etwas knifflig, es ist aber kein grösseres Problem aufgetreten.
Lösungen	Wenn die Dokumentation fehlt, einfach probieren und raten.
Durchgeführte Tests	Test für Tag User, Shop und die Hälfte von Produkt.
Wissensbeschaffung	L5-Swagger-Dokumentation
Beanspruchte Hilfeleistung	Keine
Vergleich mit dem Soll-Zeitplan	Ich liege weiterhin circa sechs Stunden hinter dem Zeitplan. Dabei habe ich erst die Hälfte der Tests abgeschlossen. Nach Abschluss der Tests werde ich ungefähr zehn Stunden hinter dem Zeitplan liegen.
Persönliche Tagesreflexion	Ich bin sehr zufrieden, die Realisierungsphase abgeschlossen zu haben. Das Testen ist eine Fleissarbeit, ich bin aber auch dort gut vorwärtsgekommen.

Tabelle 12 - Arbeitsprotokoll Tag 8

Montag, 27. März 2023

Tagesziele gemäss Zeitplan	Implementation dokumentieren, Dokumentation abschliessen
Ungeplante Arbeiten / geleistete Überstunden	In Absprache mit meinem Berufsbildner, Domenik Hofer, habe ich am Wochenende an der IPA gearbeitet. Ich habe circa sechs Stunden am Projekt gearbeitet. In dieser Zeit habe ich das Projekt auf meinem Computer aufgesetzt, was zuvor nicht funktioniert hatte. Ausserdem habe ich die Testdokumentierung abgeschlossen.
Erreichte Ziele, Erfolgserlebnisse	Ich konnte die Dokumentation inhaltlich abschliessen.
Herausforderungen Probleme	Die Word-Inhaltsverzeichnisse hatten diverse Anzeigefehler.
Lösungen	Nach dem Löschen und neu Erstellen wurden sie korrekt angezeigt.
Durchgeführte Tests	Keine
Wissensbeschaffung	-
Beanspruchte Hilfeleistung	-
Vergleich mit dem Soll-Zeitplan	Ich liege wieder gut im Zeitplan. Die Soll- und Ist-Werte stimmen überein.
Persönliche Tagesreflexion	Dank der geleisteten Überstunden am Wochenende, kann ich dem Abgabetag entspannt gegenüberstehen. Auch kleine Probleme mit Word etc. sollten der rechtzeitigen Abgabe nicht im Weg stehen.

Tabelle 13 - Arbeitsprotokoll Tag 9

Dienstag, 28. März 2023

Tagesziele gemäss Zeitplan	Dokumentation abschliessen
Ungeplante Arbeiten / geleistete Überstunden	Keine
Erreichte Ziele, Erfolgserlebnisse	Ich konnte die Dokumentation erfolgreich abschliessen.
Herausforderungen Probleme	Es war etwas Probieren nötig, um den Zeitplan leserlich in Word abzubilden.
Lösungen	Umstrukturieren, um die Höhe zu verringern.
Durchgeführte Tests	Keine
Wissensbeschaffung	-
Beanspruchte Hilfeleistung	-
Vergleich mit dem Soll-Zeitplan	Ich liege genau im Zeitplan.
Persönliche Tagesreflexion	Ich konnte die alle Arbeiten stressfrei abschliessen und hatte weder zu wenig noch zu viel Zeit. Ich bin sehr zufrieden mit dem Verlauf.

Tabelle 14 - Arbeitsprotokoll Tag 10

Teil 2: Projekt

Kurzfassung

Ausgangssituation

Um Bestellungen von verschiedenen Personen für die Firma detailliert festzuhalten, soll eine Bestellungs-Übersicht erstellt werden. Ziel ist es, Bestellungen mit ihren Produkten organisiert darzustellen und Benutzern zuzuordnen. In dieser Arbeit wird zu diesem Zweck eine API entwickelt, welche diese Daten speichert und sie als JSON ausgibt. In einem späteren Schritt soll ein Frontend entwickelt werden, welche diese Daten konsumiert. Das ist jedoch nicht Teil dieser Arbeit.

Umsetzung

Das Projekt folgt den im Zeitplan definierten Schritten, der anhand der IPERKA-Methode aufgebaut ist. In der Informationsphase wurden die nötigen Informationen gesammelt und der Auftrag spezifiziert. Bei der Planungsphase wurde das Konzept erarbeitet und die Diagramme dazu erstellt. Dann wurde über das Vorgehen entschieden. In der Realisierungsphase wurde das Geplante umgesetzt. Zum Schluss wurde anhand der Testspezifikation getestet. Die Auswertung beendet das Projekt.

Ergebnis

Es wurde eine API mit dem PHP-Framework Laravel erstellt, auf die über Swagger zugegriffen werden kann. Diese API erlaubt es, sich als Nutzer zu registrieren und anzumelden. Es können Shops, Produkte, Bestellungen und bestellte Produkte erstellt, verändert, angesehen und gelöscht werden. Bei falschen Eingaben werden sinnvolle Fehlermeldungen im JSON-Format ausgegeben.

Informieren

Ausgangslage

Mit einer API sollen Bestellungen in Online-Shops und dazu gehörigen Produkte erstellt und bearbeitet werden können. Die API soll das Registrieren von Benutzern erlauben. Nur registrierte Benutzer sollen Zugang zu den Endpoints der API haben.

Abklärungen

Autorisierung und Datenzugang. Jeder angemeldete Benutzer hat vollen Zugang zu allen Daten. Die Produkte und Online-Shops von anderen Benutzern können für die eigenen Bestellungen verwendet werden. Es können Bestellungen von anderen Benutzern verändert und gelöscht werden.

Verstandene Aufgabenstellung und Ziel der Arbeit

Es muss ein sinnvolles Datenbankschema erstellt werden. Routen für die Funktionen der API müssen bereitgestellt werden und die dazugehörige Controller-Logik muss erstellt werden. Es sollen Antworten in JSON zurückgegeben werden.

Verfeinerung des Auftrages

Es ist ein Bildupload für Produkte erforderlich. Ich werde das mit einer Speicherung von URLs in der Datenbank und dem Speichern von Bildern auf dem Server lösen. Das ist das Standardvorgehen mit Laravel, anstatt der Benutzung eines Blobs.

Aus praktischen Gründen könnte man argumentieren, dass Redundanzen in der Datenbank die Logik vereinfachen würden. Ich habe mich dagegen entschieden, weil die dritte Normalform eine Vorgabe ist. Für die Filterung und Sortierung von Modellen existiert ein Framework, das in Laravel integriert werden kann. Es heisst Scout. Dieses ist jedoch sehr umfangreich und daher aufwendig zu implementieren. Da ich nur sehr wenige solche Endpoints habe, habe ich mich gegen die Verwendung entschieden.

Projektumfeld und Systemgrenzen

Die API läuft auf einem XAMPP Server und soll lokal auf demselben Computer verfügbar sein. Der Zugriff erfolgt über den Browser via Localhost. Zu diesem Zeitpunkt ist keine Veröffentlichung geplant.

Planen

Verwendete Projekt Management Methode

Das Vorgehen während der IPA wird in die Schritte der IPERKA-Projektmanagementmethode eingeteilt.

- Informieren
- Planen
- Entscheiden
- Realisieren
- Kontrollieren
- Auswerten

Versionierung und Datensicherheit

Änderungen werden in einem GIT-Repository festgehalten, so wird auch die Datensicherheit gewährleistet.

Priorisierung der Tätigkeiten

Beschreibung der Tätigkeit	Priorität
Datenbank planen	Sehr hoch
Endpoints planen	Sehr hoch
Use Cases beschreiben	Niedrig
Authentifizierung sicherstellen	Hoch
Swagger implementieren	Hoch
Tests durchführen	Hoch
Dokumentation erstellen	Normal

Tabelle 15 - Tätigkeiten Liste

Kernfeature Laravel API

User sollen sich registrieren und anmelden können. Daten sollen gespeichert und erhalten werden können. Die Daten sollen im JSON-Format ausgegeben werden. Authentifizierung und Datensicherheit sind gewährleistet. Es sollen verständliche Fehlermeldungen ausgegeben werden.

Aktivitätsdiagramm Laravel API

Bei dieser API wird jede Aktivität stets von einem Benutzer angestossen werden. Jeder End Point repräsentiert dabei eine Aktivität. Wie zum Beispiel das Hinzufügen von Produkten:

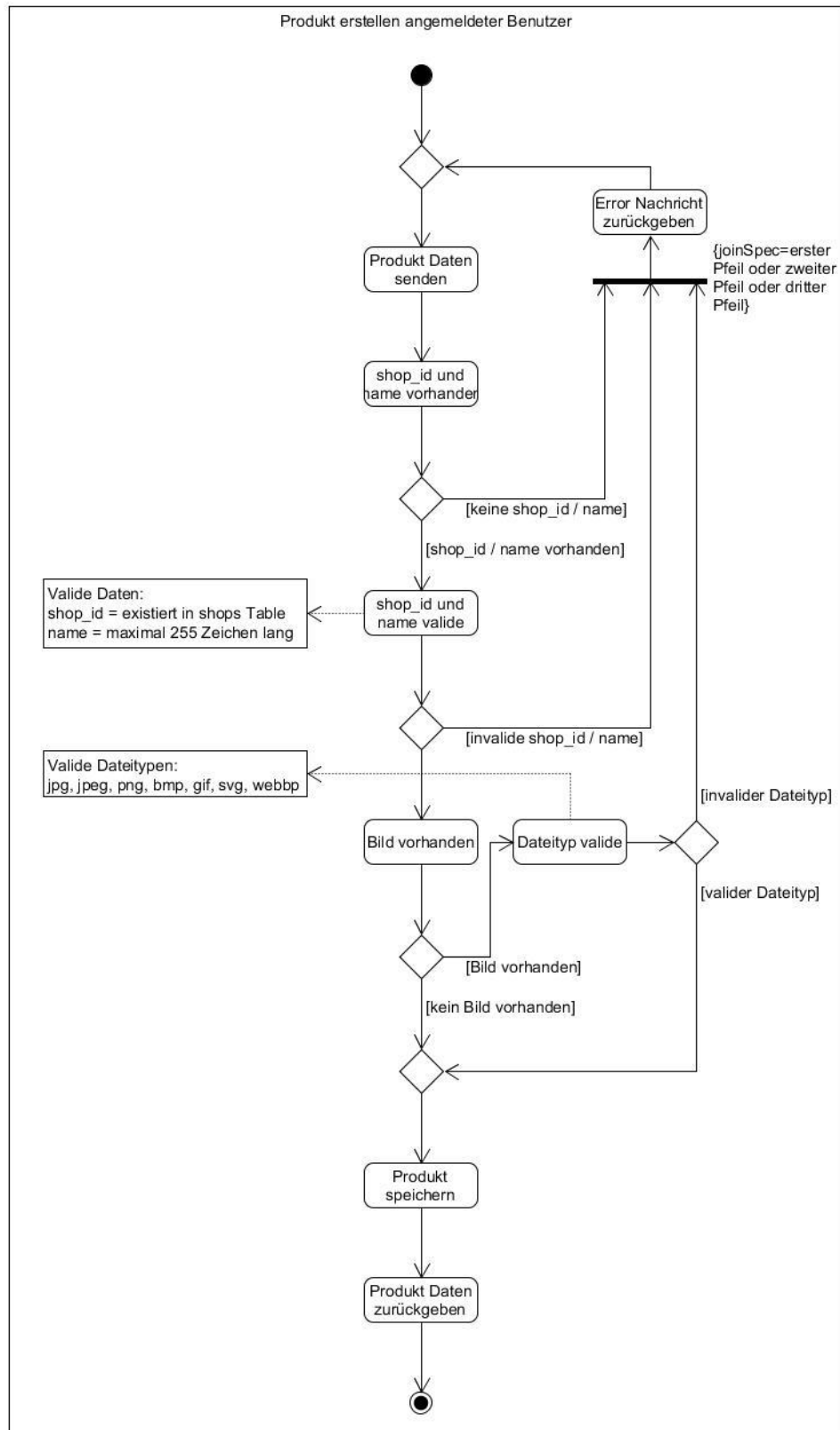


Abbildung 1: Aktivitätsdiagramm Produkt erstellen

Klassendiagramm

Untenstehende Abbildung zeigt das Klassendiagramm für die Datenbank. Die Tabellen für Migrationen, Tokens und Jobs sind hier nicht abgebildet, um die Übersichtlichkeit zu erhöhen und weil sie hier keine grosse Relevanz haben.

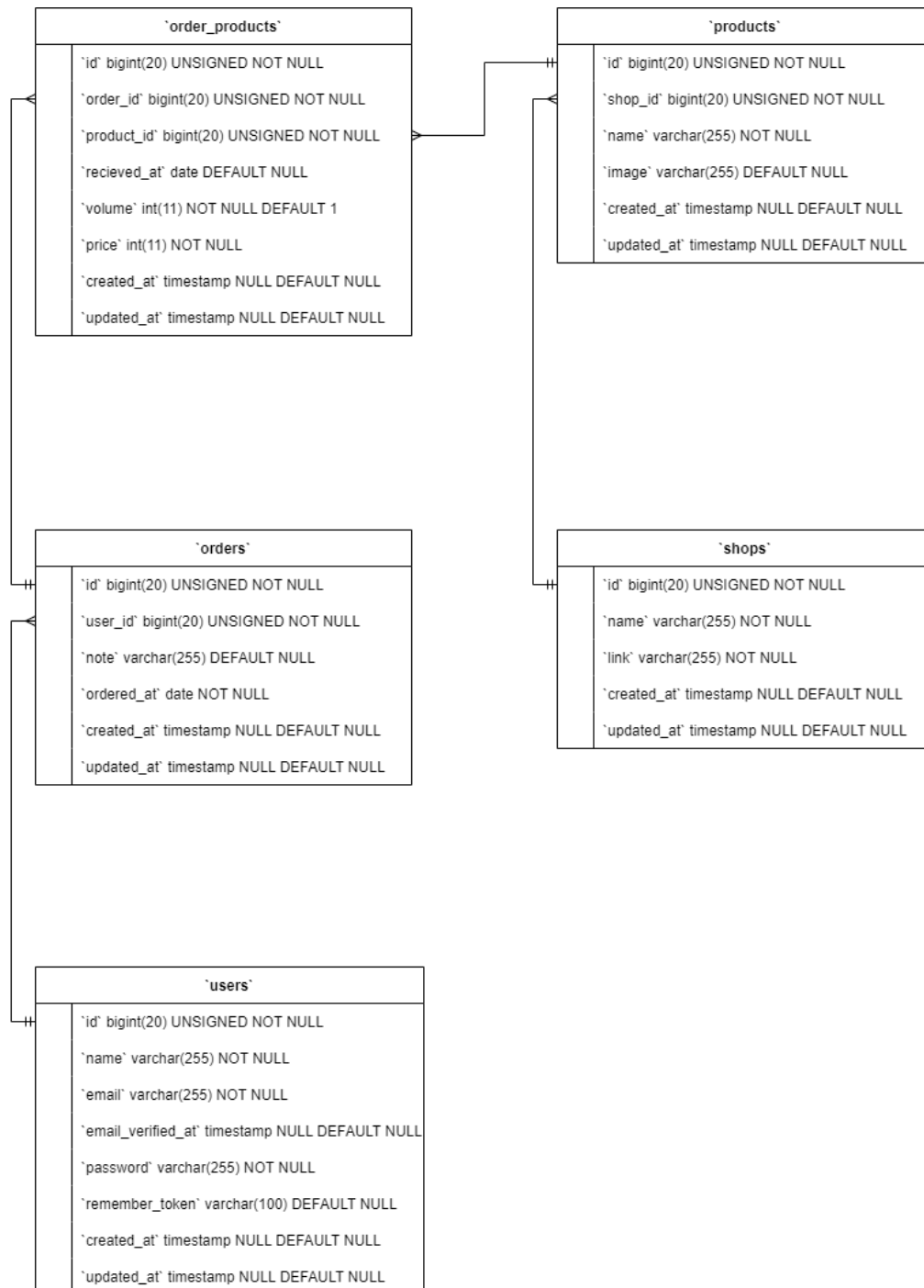


Abbildung 2: Klassendiagramm

Konzeptioneller Aufbau

Folgendes Diagramm stellt den konzeptionellen Aufbau dar. Es zeigt, wie ein Request die Struktur von Laravel durchläuft.

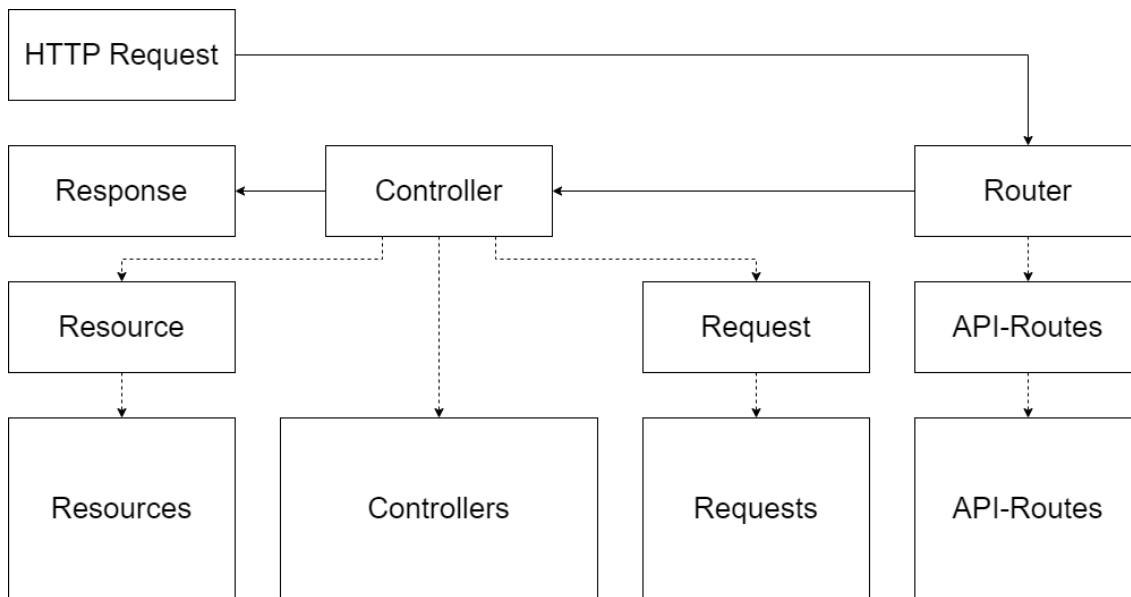


Abbildung 3: Aufbaudiagramm Laravel Konzeptionell

Hier abgebildet ist der vereinfachte Lifecycle eines Requests. Die Stationen werden nach diesem Konzept aufgerufen.

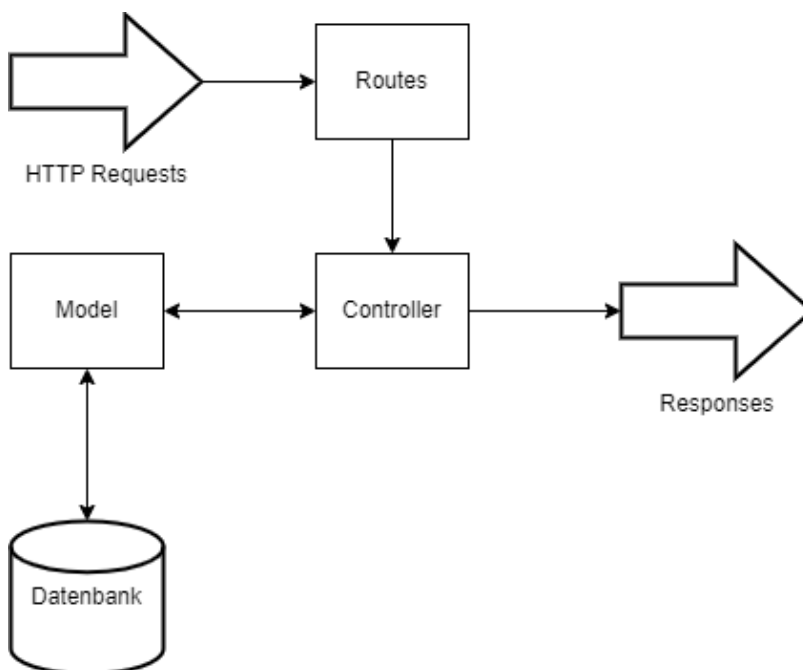


Abbildung 4: Lifecycle-Diagramm Request

Laravel ist ein objektorientiertes Framework. Es benutzt Modelklassen, um auf die Datenbank zuzugreifen. Modelle können auch Attribute haben, welche Beziehungen zwischen Modellen repräsentieren können.

Kernfeature Swagger GUI

Swagger ist ein Format zur Beschreibung einer REST-API. Es ist Open Source und benutzt OpenAPI Spezifikationen. Swagger kann für verschiedene Programmiersprachen und Frameworks verwendet werden. Da diese Anwendung auf dem Laravel-Framework aufgebaut ist, wurde das eine Version von Swagger spezifisch für Laravel verwendet. Das GitHub Repository von DarkaOnLine, mit dem Namen L5 Swagger¹⁰. Diese Version kann über Composer installiert werden.

GUI-Design / Mockup

Swagger stellt ein Frontend zur Verfügung, mit dem auf die Endpoints der API zugegriffen werden kann.

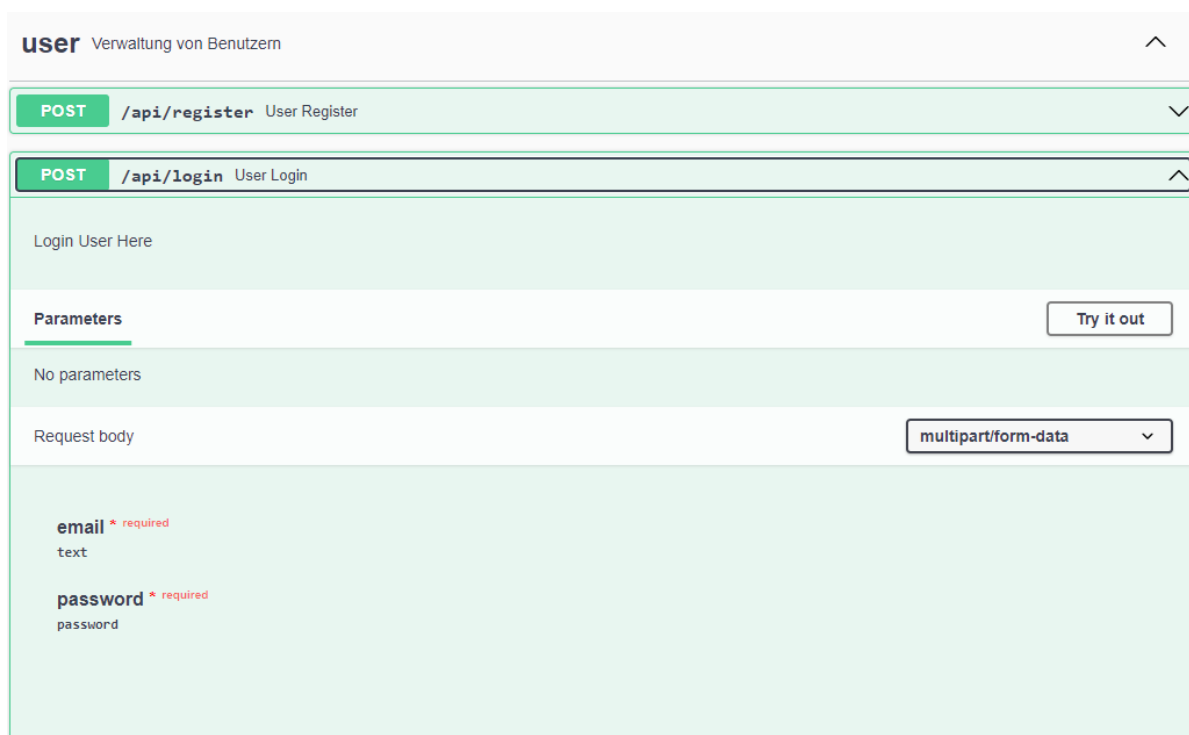


Abbildung 5: Mock-Up Swagger

Die Endpoints können über Tags in Gruppen eingeteilt werden, somit können sie nach Controller geordnet angezeigt werden. Mit Schemas können ausserdem die Klassen definiert werden, die zurückgegeben werden.

¹⁰ (DarkaOnLine, 2023)

user Verwaltung von Benutzern	▼
shop Verwaltung von Geschäften	▼
product Verwaltung von Produkten	▼
order Verwaltung von Bestellungen	▼
orderProduct Verwaltung von bestellten Produkten	▼
Schemas	▼

Abbildung 6: Mock-Up Swagger Schemas

Konzeptioneller Aufbau

Swagger für Laravel benutzt Annotationen im Code, um Endpoints und Schemas zu definieren. Anhand dieser wird dann eine HTML-Seite mit JavaScript generiert, die über den Browser aufgerufen werden kann.

```
/**
 * @OA\Info(
 *   description="Diese Web-App verwaltet Bestellungen und die dazugehörenden Produkte und Geschäfte.
 *   Dieses Projekt wurde im Laufe einer IPA erstellt.",
 *   version="1.0.0",
 *   title="Bestellungs-Übersicht | API-Backend in Laravel",
 *   @OA\Contact(
 *     email="julien.raedler@twofold.swiss"
 *   ),
 *   @OA\License(
 *     name="Apache 2.0",
 *     url="http://www.apache.org/licenses/LICENSE-2.0.html"
 *   )
 * )
 */
```

In unserer Applikation sind die Swagger Annotationen und die generierte Seite auf dem gleichen Server wie unsere API. Das muss aber nicht immer der Fall sein. In anderen Anwendungen kann Swagger auch benutzt werden, um auf eine API auf einem anderen Server zuzugreifen.

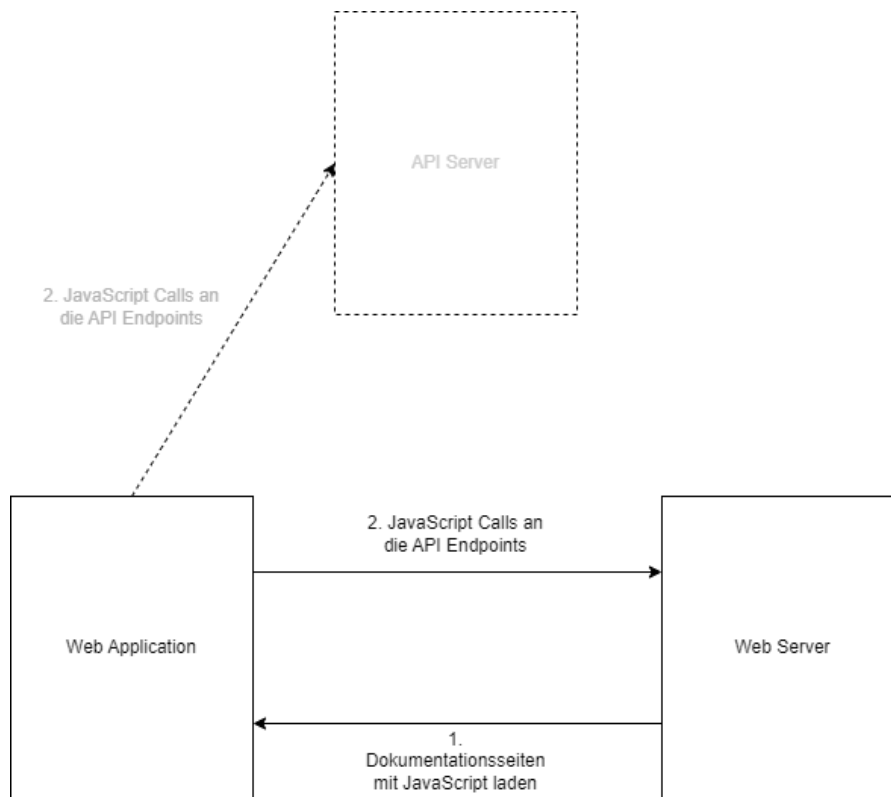


Abbildung 7: Aufbaudiagramm Swagger konzeptionell

Geplantes Vorgehen für die Qualitätssicherung

Durch die Einhaltung der Laravel und Swagger Konventionen, ist der Code für eine kompetente aussenstehende Person einfach nachvollziehbar und erweiterbar.

Weiter wird durch das händisch erstellte Testkonzept eine korrekte Funktionsweise sichergestellt.

Anmerkungen zum Zeitplan

Die Soll- und Ist-Zeit waren häufig in Übereinstimmung und die Zeitblöcke waren meist richtig bemessen. Ich bin alles in allem zufrieden mit dem Zeitplan. Es war eine besonders gute Entscheidung eine Pufferzeit einzuplanen.

Begründung für Abweichungen zum Zeitplan

Ich habe den Dokumentationsaufwand in der Planungsphase ein wenig unterschätzt, wodurch ein Rückstand entstand. Das Erstellen der Ressourcen nahm dabei weniger Zeit in Anspruch als das Einbinden und Erklären in der Dokumentation. Dies stellte aber dank einer eingeplanten Pufferzeit kein grosses Problem dar. Um den entstandenen Rückstand aufzuholen, habe ich am Wochenende fünf Stunden Überzeit geleistet. Dies fand während den folgenden zwei Tagen statt: 25.03.2023 und 26.03.2023

Entscheiden

Variante Bilder als Blob speichern

Eine Bilddatei kann vollständig als sein sogenannter Blob in der Datenbank gespeichert werden. Der Vorteil dieser Methode ist, dass es nicht zwei unterschiedliche Referenzen gibt. Wenn die Datei gelöscht oder bearbeitet wird, passiert das alles in der Datenbank. Es besteht so kein Risiko, das ein URL nicht mit einer separaten Datei übereinstimmt. Die Datenbank entspricht dem ACID Standard. Backups und Transfers der Datenbank sind somit problemlos möglich.

Variante Bilder in Dateisystem speichern

Bilder werden als Bilddateien gespeichert und nur der URL des Bildes wird in der Datenbank referenziert. Der Vorteil dieser Methode ist, dass das Laden eines Strings aus der Datenbank viel schneller ist als eine grosse Datei zu laden. Falls etwas gestreamt werden muss, wie bei Video, ist das auch nur mit dieser Variante möglich.

Kurzbeschreibung der Entscheidungskriterien

- Performance
Wie gut ist die Performance?
- Laravel-Style
Was ist das von der Laravel-Dokumentation vorgeschlagene Weg?
- Komplikation
Wie viel extra Komplikation entsteht in der Logik?
- Integrität und Datensicherheit
Wie robust ist die Datensicherung?

Performance sollte immer in Betracht gezogen werden, ist aber in einem kleinen Projekt mit nur einem Bild pro Produkt weniger wichtig. Ob es dem Laravel-Style entspricht wird hier schwerer gewichtet, weil es auch Komplikation verringert und sauberen Code erzeugt. Es sind keine Migrationen oder Backups vorgesehen, trotzdem sollte die Integrität und Datensicherheit nicht vernachlässigt werden.

Entscheidungsmatrix

Kriterium	Gewichtung	Blob		Datei	
		Wert	G-Wert	Wert	G-Wert
Performance	10%	2	20	8	80
Laravel-Style	50%	4	200	10	500
Komplikation	30%	6	180	10	300
Integrität und Datensicherheit	10%	10	100	4	40
Total			500		920

Tabelle 16: Entscheidungsmatrix Bild

Variante Shop-Name direkter Vergleich

Der eingegebene Such-String wird direkt benutzt, um in der Shop-Tabelle nach Namen zu suchen. Wenn der String genau dem Namens-String entspricht, wird der Shop als Suchergebnis zurückgegeben.

Variante Vergleich mit PHP-Methode

PHP stellt die `similar_text` Methode zur Verfügung. Die Methode vergleicht zwei Strings und gibt einen Prozentwert der Übereinstimmung zurück.

Variante Vergleich mit Scout

Scout ist ein Textsuche-Framework, das gut in Laravel integriert ist. Es stellt volle Textsuche über jedes gewünschte Model zur Verfügung.

Kurzbeschreibung der Entscheidungskriterien

- Laravel-Style
Was ist das von der Laravel-Dokumentation vorgeschlagene Weg?
- Komplikation
Wie viel extra Komplikation entsteht in der Logik?
- Benutzerfreundlichkeit
Wie Benutzerfreundlich ist die Lösung?
- Erweiterbarkeit
Wie erweiterbar ist diese Lösung?

Ob es dem Laravel-Style entspricht, wird hier weniger gewichtet, weil es in diesem Fall die Komplikation erhöht. Die Erweiterbarkeit ist weniger wichtig, weil nichts in diesem Bereich vorgesehen ist. Selbst wenn Erweiterungen kommen würden, wäre es immer noch eine kleine Applikation und würde nicht zu einer grossen App werden.

Entscheidungsmatrix

Kriterium	Gewichtung	Direkter Vergleich		PHP-Methode		Scout	
		Wert	G-Wert	Wert	G-Wert	Wert	G-Wert
Laravel-Syle	10%	6	60	6	60	10	100
Komplikation	40%	10	400	10	400	2	80
Benutzerfreundl.	40%	2	80	8	320	10	400
Erweiterbarkeit	10%	10	100	2	20	10	100
Total			640		800		680

Tabelle 17: Entscheidungsmatrix Textsuche

Realisieren

Abbildung des Gesamtsystems

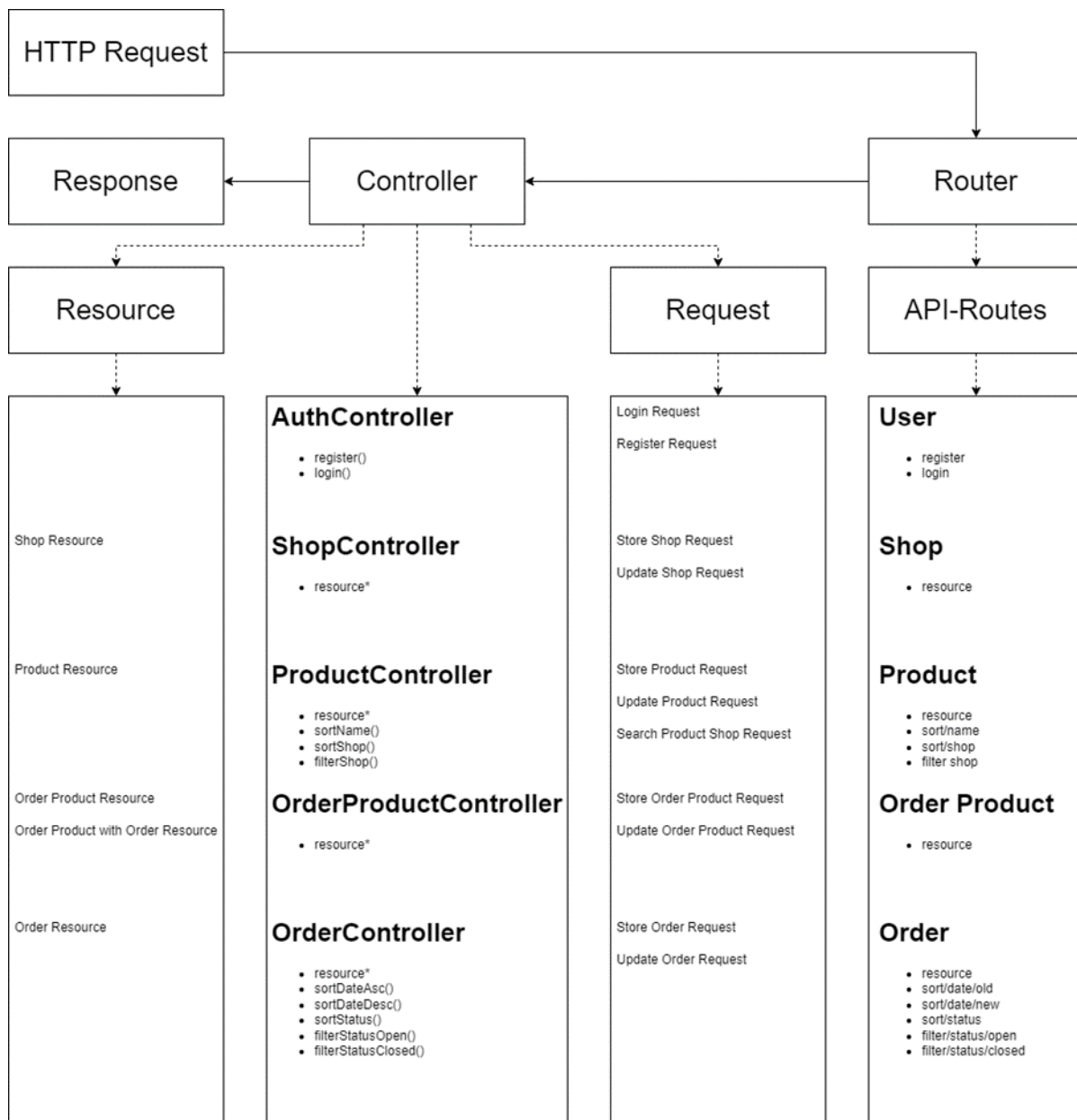


Abbildung 8: Aufbaudiagramm Laravel

Sobald ein Request eintrifft, der einen in den API-Routes definierten URL benutzt, wird der Request an den dazugehörigen Controller weitergeleitet. Im Controller werden Requests von den entsprechenden Methoden aufgerufen. Je nach Request werden verschiedene Validierungsregeln angewendet. Diese sind in den Requests definiert. Die Daten werden dann an Ressourcen weitergegeben, die als Antworten zurückgesendet werden.

*Aufbau einer Laravel-Ressource:

Verb	URI	Action	Route Name
GET	/products	Index	Products.index
GET	/products/create	Create	Products.create
POST	/products	Store	Products.store
GET	/products/{product}	Show	Products.show
GET	/products/{product}/edit	Edit	Products.edit
PUT/PATCH	/products/{product}	Update	Products.update
DELETE	/products/{product}	Destroy	Products.destroy

Tabelle 18: Laravel Ressource

Diese Tabelle zeigt die Routen, die durch eine Laravel-Ressource Route erstellt werden, am Beispiel der Produktklasse.

Erstellen einer Laravel-Ressource:

```
Route::apiResource('products', 'App\Http\Controllers\ProductController');
```

Hier werden die Routes erstellt und dem Produkt-Controller zugeordnet.

Interaktionen zwischen den Teilsystemen

Das System funktioniert lokal. Die Kommunikation findet zwischen dem XAMPP Server und dem Browser statt.

Vorgehensweise

- Erstellung eines neuen Laravel-Projektes mit dem PHP-Packet Management Systems Composer.
- Zeitplan und Dokumentation zum Projekt hinzufügen.
- Projekt zu GIT-Repository hinzufügen.
- Laravel-Sanctum über Composer installieren.
- Logik für das Projekt schreiben.
- Swagger über Composer installieren.
- Swagger Logik schreiben.

Anbindung an die Datenbank

Laravel stellt mit ORM Eloquent Modelklassen zur Verfügung, welche mit der Datenbank kommunizieren können. Die Datenbank selbst wird über Migrationen erstellt und mit einem Seeder mit Testdaten gefüllt.

Implementierung des Kernfeatures Laravel-API

CRUD-Funktionalität:

Die Filterung der Produkte nach Shops setzt eine Textsuche für Shops voraus. Es gab drei Möglichkeiten, wie diese Funktionalität erreicht werden konnte. Der Name wird wortwörtlich in der Shop-Tabelle gesucht und muss genau richtig geschrieben sein, ein selbst geschriebener Vergleich des Suchbegriffs mit dem Namen oder das Verwenden von Scout, einem Framework, das in Laravel integriert ist, das volle Textsuche über alle Modelle ermöglicht. Ich habe mich gegen die Benutzung von Scout entschieden, weil es sich um ein relativ kleines Projekt handelt. Es hat nur eine einzelne Text-Suche. Hätte es mehrere oder kompliziertere Suchen, wäre es angemessen, ein ausführlicheres Suchsystem zu verwenden.

```
public function filterShop(SearchProductShopRequest $request)
{
    $product = Product::all();
    $product = $product->filter(function (Product $p) use ($request) {
        similar_text(strtoupper($p->shop->name), strtoupper(
            $request->shop), $percent);
        return $percent > 50;
    });
    return response(ProductResource::collection($product));
}
```

Ich habe mich für einen minimalistischen Textvergleich entschieden. Ich verwende dazu die PHP-Funktion `similar_text`, diese vergleicht zwei Strings und gibt einen Prozentwert zurück. Ich habe eine Hürde von 50 Prozent Übereinstimmung gewählt, ab dem ein Shop-Name als Ergebnis zurückgegeben wird.

Fehlermeldungen:

Eine Vorgabe für diese Arbeit, ist es, verständliche Fehlermeldungen im JSON-Format auszugeben. Das ist grösstenteils kein Problem, da die Meldungen in den Request-Klassen frei definiert werden können. Da ich für die Routes aber das Route Model Binding, das von Laravel zur Verfügung gestellt wird benutze, muss ich diese Fehlermeldungen separat abfangen. Diese Fehler werden nämlich noch bevor meine Request aufgerufen werden ausgelöst.

```
// Catch not found exception for shop routes and return API friendly error
message.
$this->renderable(function (NotFoundException $e, $request) {
    if ($request->is('api/shops/*')) {
        return response()->json([
            'message' => 'Shop-Datensatz nicht gefunden.'
        ], 404);
    }
});
```

Ich bin so vorgegangen, dass ich für jeden API-URL der Routen, die auch existieren die NotFoundException abfange und durch meine eigene Fehlermeldung ersetze. Diese Lösung habe ich auf der Seite Laraveldaily¹¹ gefunden.

Authentifizierung:

Benutzer können sich registrieren und anmelden, wobei ihnen jeweils ein Bearer-Token zugewiesen wird. Dieses Token muss bei jedem Request mitgeschickt werden, um Zugang zur API zu erhalten.

Funktionsweise des Kernfeatures Laravel-API

Dieses Feature erlaubt das Erstellen, Ansehen, Verändern und Löschen von Shops, Produkten, Bestellungen und bestellten Produkten. Weiter umfasst es Benutzerregistrierung und Login. Bei fehlendem Bearer-Token oder ungültigen Eingaben, werden verständliche Fehlermeldungen zurückgegeben.

Ziel des Kernfeatures Laravel-API

Das Ziel ist, einen benutzerfreundlichen und logisch aufgebauten Zugang zur Verwaltung von Bestellungen, Produkten und Shops gewährleisten.

¹¹ (Laraveldaily, 2023)

Implementierung des Kernfeatures Swagger

Kompromisse:

Die Laravel-Ressource für Routen sieht vor, dass die Update-Methode im Controller mit einer PUT-Methode angesprochen wird. Bei dieser Methode wird aber multipart formdata nicht unterstützt. Das führte dazu, dass der Request zwar ausgeführt wurde aber keine Daten mitgeschickt wurden. Ich musste aber multipart formdata benutzen, um ein Bild mitzuschicken.

Die Lösung war schlussendlich die PUT-Methode als eine Variable im Request-Body mitzugeben. Der Request im Frontend ist aber trotzdem ein POST-Request.

```
@OA\Property(property="_method", type="string", example="PUT"),
```

POST

/api/products/{productId} Update a product

Funktionsweise des Kernfeatures Swagger

Swagger benutzt Annotationen im Laravel-Code und generiert daraus eine HTML und Javascript Seite. Diese greift dann auf die beschriebenen Controller-Funktionen zu.

Annotation im Product-Controller:

```
/**
 * @OA\Get(
 *     path="/api/products",
 *     operationId="getProducts",
 *     tags={"product"},
 *     summary="Get all products",
 *     description="Returns list of products",
 *     security={ {"bearerAuth": {}} },
 *     @OA\Response(
 *         response=200,
 *         description="Successful operation",
 *         @OA\JsonContent(
 *             @OA\Property(property="products", type="array",
collectionFormat="multi",
 *                 @OA\Items(ref="#/components/schemas/ProductResource"))
 *         ),
 *     ),
 *     @OA\Response(
 *         response=401,
 *         description="Unauthenticated",
 *     ),
 *     @OA\Response(
```

```

*         response=403,
*         description="Forbidden"
*     )
*     )
*/
public function index()
{
    $product = Product::all();
    return response(ProductResource::collection($product));
}

```

Generierter Swagger-Code:

```

"/api/products": {
  "get": {
    "tags": [
      "product"
    ],
    "summary": "Get all products",
    "description": "Returns list of products",
    "operationId": "getProducts",
    "responses": {
      "200": {
        "description": "Successful operation",
        "content": {
          "application/json": {
            "schema": {
              "properties": {
                "products": {
                  "type": "array",
                  "items": {
                    "$ref":
"#/components/schemas/ProductResource"
                  },
                  "collectionFormat": "multi"
                }
              },
              "type": "object"
            }
          }
        },
        "401": {
          "description": "Unauthenticated"
        },
        "403": {

```



```

        "description": "Forbidden"
      },
    },
    "security": [
      {
        "bearerAuth": []
      }
    ]
  },
},

```

Ziel des Kernfeatures Swagger

Durch Swagger wird eine nützliche Schnittstelle geboten, die es ermöglicht, alle Endpoints auszuprobieren. Es werden weiter ausführliche Informationen zur Funktionsweise geboten. So wird eine benutzerfreundliche Dokumentation erstellt.

Effektives GUI / Produkt

Das Endprodukt ist eine interaktive Webseite mit allen Endpoints.

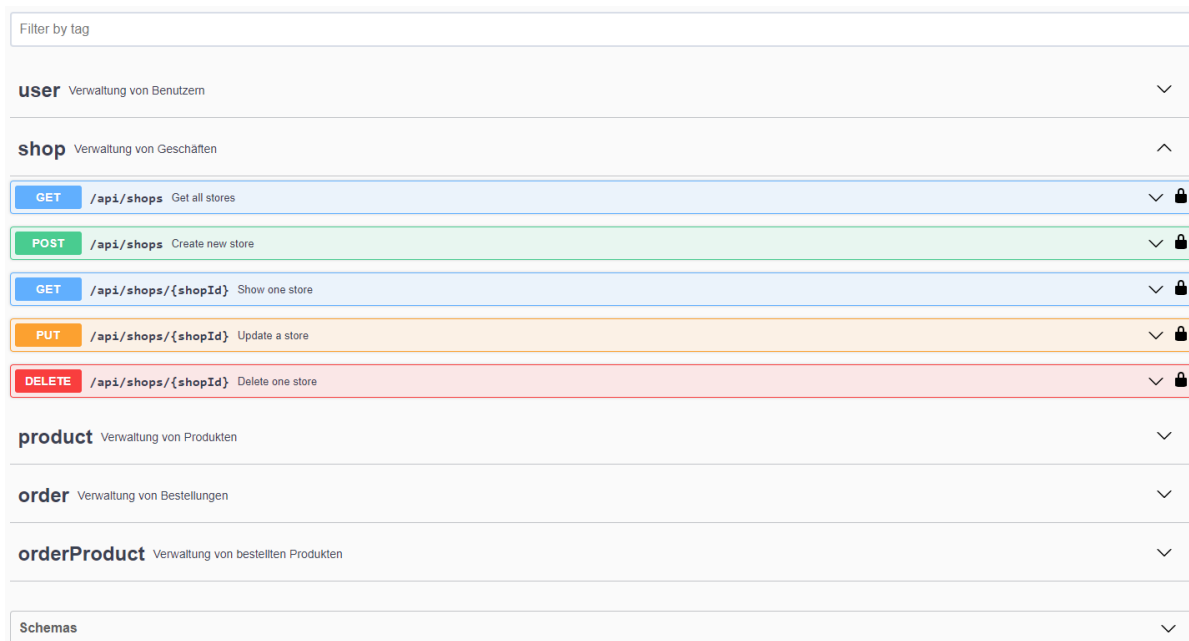


Abbildung 9: Swagger GUI

Kontrollieren

Beschreibung der Randbedingungen / Testanlage (Umfeld)

Durch das PHP-Framework Laravel, werden Endpunkte für Benutzer, Shops, Bestellungen und Produkte bereitgestellt. Über die Implementierung von Swagger, werden diese Endpunkte dokumentiert, können aber auch händisch getestet werden.

Eingesetzte Testmittel und -methoden

Es wird ein Testprotokoll erstellt, das dann über das Swagger-UI händisch ausgeführt und dokumentiert wird.

Beschreibung der Testszenarien

Ein User muss sich registrieren und anmelden können. Dazu müssen Bearer-Tokens erstellt und zurückgegeben werden.

Ein User mit gültigem Bearer-Token muss Zugriff auf alle Endpoints und Daten haben. Das beinhaltet Ansehen, Erstellen, Bearbeiten, Filtern und Löschen.

Generelle Anforderungen

Name:	Validierung
Kategorie:	Funktional
Beschreibung:	Anfragen werden in Request-Klassen validiert und aufbereitet. Fehler in den Eingaben werden dem Nutzer verständlich, auf Deutsch, zurückgeschickt.

Name:	Autorisierung
Kategorie:	Funktional
Beschreibung:	Die Routen und Inhalte sind geschützt, sodass nur eingeloggte User Zugang haben. Die Authentifizierung soll via Bearer-Token funktionieren.

Name:	CRUD-Verben
Kategorie:	Funktional
Beschreibung:	Die Routen werden mit den korrekten Verben angesprochen. (Create = POST, Read = GET, Update = PUT/PATCH, Delete = DELETE)

Name:	CRUD
Kategorie:	Funktional
Beschreibung:	Der eingeloggte Nutzer sollte dann die Möglichkeit haben, Produkte zu erstellen, bearbeiten und zu löschen. Zudem soll der Nutzer die Möglichkeit haben, Produkte einer Bestellung zuzuweisen und mit allen Inhalten zu speichern, bearbeiten und zu löschen. Inhalte sollen mit der gewünschten Sortierung und Filterung abgefragt werden können.

Name:	JSON
Kategorie:	Nicht funktional
Beschreibung:	Die Kommunikation mit der API läuft über JSON.

Die generellen Anforderungen gelten für alle Routen. Sie werden somit für jeden Text angewendet und müssen immer erfüllt sein, um einen Test als bestanden bewerten zu können.

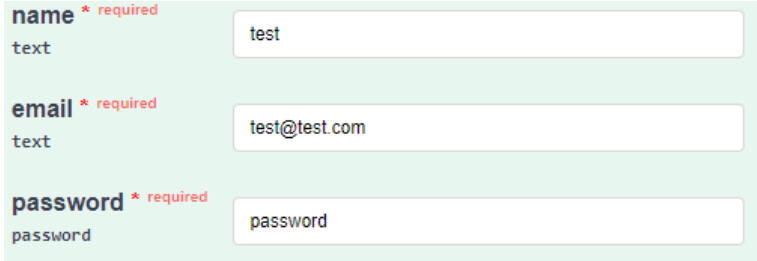
Kurzbeschreibung:	Registrierung als User möglich
Voraussetzung:	-
Eingabe: Post Request an /api/register	
	
Ausgabe: Statuscode 200	
<pre>{ "access_token": "3 Zp7IywNd2nyVrxRufgQYhZLixfORuPDaWrpfPD6t", "token_type": "Bearer" }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 19 - Testprotokoll für User Registrierung

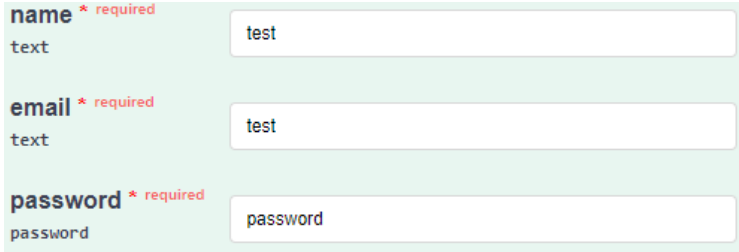
Kurzbeschreibung:	Registrierung als User falsche Eingaben
Voraussetzung:	-
Eingabe: Post Request an /api/register	
	
Ausgabe: Statuscode 422	
<pre>{ "message": "Die E-Mail Adresse ist im falschen Format", "errors": { "email": ["Die E-Mail Adresse ist im falschen Format"] } }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 20: Testprotokoll für User Registrierung Fehler

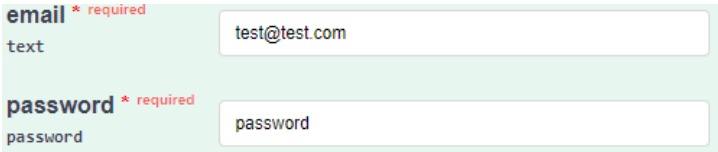
Kurzbeschreibung:	Login als User ist möglich
Voraussetzung:	User Datensatz existiert in der Datenbank
Eingabe: Post Request an /api/login 	
Ausgabe: Statuscode 200 <pre>{ "access_token": "5 54EkesvNtHP14wpEcjGtGpbDmF1jX00Ifqefp8M6", "token_type": "Bearer" }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 21: Testprotokoll für User Login Fehler

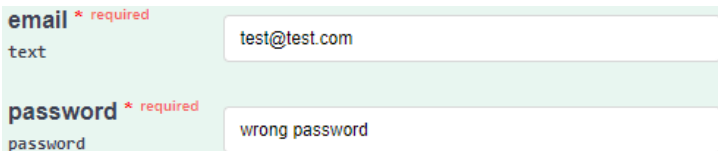
Kurzbeschreibung:	Login als User falsche Eingabe
Voraussetzung:	-
Eingabe: Post Request an /api/login 	
Ausgabe: Statuscode 401 <pre>{ "message": "Invalid login details" }</pre>	
Datum:	24.03.2023
Resultat:	

Tabelle 22: Testprotokoll für User Login Fehler

Kurzbeschreibung:	Anzeigen aller Shops
Voraussetzung:	Ein Shop existiert in der Datenbank
Eingabe: Get Request an /api/shops	
Ausgabe: Statuscode 200	
<pre>[{ "id": 1, "name": "mueller", "link": "https://www.aufderhar.com/modi-molestias-possimus-et-distinctio-nihil-dolor", "products": [{ "id": 2, "shop": "mueller", "name": "sit", "image": null }, { "id": 5, "shop": "mueller", "name": "nam", "image": null }] }, { "id": 2, "name": "keeling", "link": "http://www.auer.com/ullam-non-expedita-consequatur-omnis", "products": [{ "id": 1, "shop": "keeling", "name": "placeat", "image": null }, { "id": 4, "shop": "keeling", "name": "ab", "image": null }] }, { "id": 3, "name": "moore", "link": "https://www.fisher.com/tempore-laudantium-tenetur-sit-mollitia-molestias-magni-molestias-rerum", "products": [{ "id": 3, "shop": "moore", "name": "quam", "image": "1a25c9ab8caaba6f39efce58ab553fa7.png" }] }]</pre>	

<pre> "id": 6, "shop": "moore", "name": "USB cable", "image": null }] }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 23: Testprotokoll für Anzeige aller Shops

Kurzbeschreibung:	Anzeigen eines Shops ist möglich
Voraussetzung:	Ein Shop existiert in der Datenbank
Eingabe: Get Request an /api/shops/1	
<div> ID of shop <div>1</div> </div>	
Ausgabe: Statuscode 200	
<pre> { "id": 1, "name": "mueller", "link": "https://www.aufderhar.com/modi-molestias-possimus-et-distinctio-nihil-dolorem", "products": [{ "id": 2, "shop": "mueller", "name": "sit", "image": null }, { "id": 5, "shop": "mueller", "name": "nam", "image": null }] }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 24: Testprotokoll für Shop Anzeige

Kurzbeschreibung:	Anzeigen eines Shops falsche Eingabe
Voraussetzung:	-
Eingabe: Get Request an /api/shops/100	
<div>ID of shop</div> <div>100</div>	
Ausgabe: Statuscode 404	
<pre>{ "message": "Shop-Datensatz nicht gefunden." }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 25: Testprotokoll für Shop Anzeige Fehler

Kurzbeschreibung:	Erstellen eines Shops ist möglich
Voraussetzung:	-
Eingabe: Post Request /api/shops	
<pre>{ "name": "Digitec", "link": "www.digitec.ch" }</pre>	
Ausgabe: Statuscode 200	
<pre>{ "id": 5, "name": "Digitec", "link": "www.digitec.ch", "products": [] }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 26: Testprotokoll für Erstellung eines Shops

Kurzbeschreibung:	Erstellen eines Shops falsche Eingabe
Voraussetzung:	-
Eingabe: Post Request /api/shops	
<pre>{ "name": "Digitec" }</pre>	
Ausgabe: Statuscode 422	
<pre>{ "message": "Ein Link ist erforderlich", "errors": { "link": ["Ein Link ist erforderlich"] } }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 27: Testprotokoll für Erstellung eines Shops Fehler

Kurzbeschreibung:	Bearbeiten eines Shops ist möglich
Voraussetzung:	Ein Shop existiert in der Datenbank
Eingabe: Put Request /api/shops/5	
<div> ID of shop <div>5</div> </div> <pre>{ "name": "Updated Digitec", "link": "www.updatedigitec.ch" }</pre>	
Ausgabe: Statuscode 200	
<pre>{ "id": 5, "name": "Updated Digitec", "link": "www.updatedigitec.ch", "products": [] }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 28: Testprotokoll für Bearbeitung eines Shops

Kurzbeschreibung:	Bearbeiten eines Shops falsche Eingabe
Voraussetzung:	Ein Shop existiert in der Datenbank
Eingabe: Put Request /api/shops/50 <div> <div>ID of shop</div> <div>50</div> </div> <pre>{ "name": "Updated Digitec", "link": "www.updatedigitec.ch" }</pre>	
Ausgabe: Statuscode 404 <pre>{ "message": "Shop-Datensatz nicht gefunden." }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 29: Testprotokoll für Bearbeitung eines Shops Fehler

Kurzbeschreibung:	Löschen eines Shops ist möglich
Voraussetzung:	Ein Shop existiert in der Datenbank
Eingabe: Delete Request /api/shops/5 <div> <div>ID of shop</div> <div>5</div> </div>	
Ausgabe: Statuscode 204	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 30: Testprotokoll für Löschung eines Shops

Kurzbeschreibung:	Löschen eines Shops falsche Eingabe
Voraussetzung:	-
Eingabe: Delete Request /api/shops/50	
<div> ID of shop <div>50</div> </div>	
Ausgabe: Statuscode 404	
<pre>{ "message": "Shop-Datensatz nicht gefunden." }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 31: Testprotokoll für Löschung eines Shops Fehler

Kurzbeschreibung:	Anzeigen aller Produkte
Voraussetzung:	Ein Produkt existiert in der Datenbank
Eingabe: Get Request /api/products	
Ausgabe: Statuscode 200	
<pre>[{ "id": 1, "shop": "keeling", "name": "placeat", "image": null }, { "id": 2, "shop": "mueller", "name": "sit", "image": null }, { "id": 3, "shop": "moore", "name": "quam", "image": "1a25c9ab8caaba6f39efce58ab553fa7.png" }, { "id": 4, "shop": "keeling", "name": "ab", "image": null }, { "id": 5, "shop": "mueller", "name": "nam", </pre>	

<pre> "image": null }, { "id": 6, "shop": "moore", "name": "USB cable", "image": null }] </pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 32: Testprotokoll für Anzeige aller Produkt

Kurzbeschreibung:	Anzeigen eines Produktes ist möglich
Voraussetzung:	Ein Produkt existiert in der Datenbank
Eingabe: Get Request /api/products/1 <div> ID of product <input type="text" value="1"/> </div>	
Ausgabe: Statuscode 200 <pre> { "id": 1, "shop": "keeling", "name": "placeat", "image": null } </pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 33: Testprotokoll für Anzeige eines Produktes

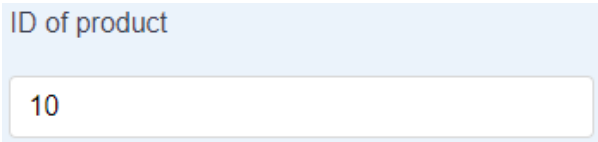
Kurzbeschreibung:	Anzeigen eines Produktes falsche Eingabe
Voraussetzung:	-
Eingabe: Get Request /api/products/10	
	
Ausgabe: Statuscode 404	
<pre>{ "message": "Produktdatensatz nicht gefunden." }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 34: Testprotokoll für Anzeige eines Produktes mit Fehler

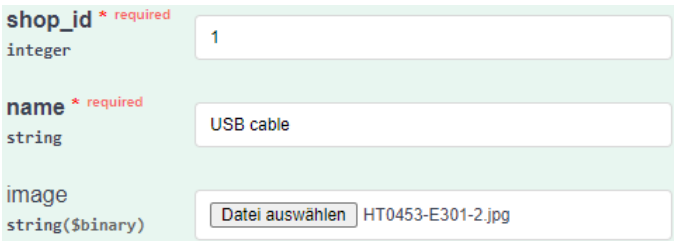
Kurzbeschreibung:	Erstellung eines Produktes ist möglich
Voraussetzung:	Ein Shop existiert in der Datenbank
Eingabe: Post Request /api/products	
	
Ausgabe: Statuscode 200	
<pre>{ "id": 8, "shop": "mueller", "name": "USB cable", "image": "Q8K63fwbsgZQmiVGAtaT0kAkiCNzaZ45CaHk7mLt.jpg" }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 35: Testprotokoll für Erstellung eines Produktes

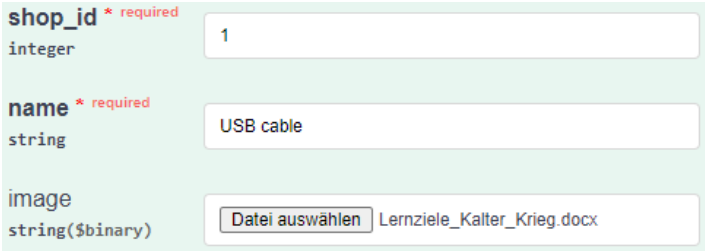
Kurzbeschreibung:	Erstellung eines Produktes falsche Eingabe
Voraussetzung:	Ein Shop existiert in der Datenbank
Eingabe: Post Request /api/products	
	
Ausgabe: Statuscode 422	
<pre>{ "message": "Muss eine Bilddatei sein", "errors": { "image": ["Muss eine Bilddatei sein"] } }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 36: Testprotokoll für Erstellung eines Produktes Fehler

Kurzbeschreibung:	Bearbeitung eines Produktes ist möglich
Voraussetzung:	Ein Shop existiert in der Datenbank
<p>Eingabe: Post Request /api/products/8</p> <div> <p>productid * required integer(\$int64) (path)</p> <p>ID of product</p> <input type="text" value="8"/> </div> <p>Request body required</p> <div> <p>Enter product Data</p> <p>shop_id integer</p> <input type="text" value="2"/> <input type="checkbox"/> Send empty value <p>name string</p> <input type="text" value="updated USB cable"/> <input type="checkbox"/> Send empty value <p>image string(\$binary)</p> <div> <input type="button" value="Datei auswählen"/> <input type="button" value="Keine ausgewählt"/> </div> <input checked="" type="checkbox"/> Send empty value <p>_method string</p> <input type="text" value="PUT"/> <input type="checkbox"/> Send empty value </div>	
<p>Ausgabe: Statuscode 200</p> <pre>{ "id": 8, "shop": "keeling", "name": "updated USB cable", "image": null }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 37: Testprotokoll für Bearbeitung eines Produktes

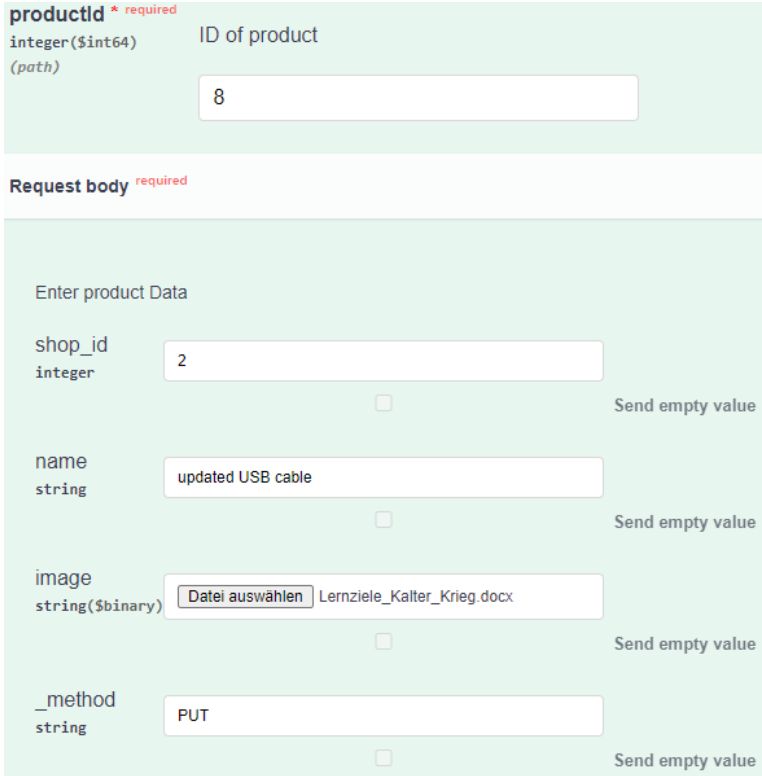
Kurzbeschreibung:	Bearbeitung eines Produktes falsche Eingabe
Voraussetzung:	Ein Shop existiert in der Datenbank
Eingabe: Post Request /api/products/8	
	
Ausgabe: Statuscode 422	
<pre>{ "message": "Muss eine Bilddatei sein", "errors": { "image": ["Muss eine Bilddatei sein"] } }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 38: Testprotokoll für Bearbeitung eines Produktes Fehler

Kurzbeschreibung:	Löschung eines Produktes ist möglich
Voraussetzung:	Ein Produkt existiert in der Datenbank
Eingabe: Delete Request /api/products/8	
<div>ID of product</div> <div>8</div>	
Ausgabe: Statuscode 204	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 39: Testprotokoll für Löschung eines Produktes

Kurzbeschreibung:	Löschung eines Produktes falsche Eingabe
Voraussetzung:	-
Eingabe: Delete Request /api/products/80	
<div>ID of product</div> <div>8</div>	
Ausgabe: Statuscode 404	
<pre>{ "message": "Produktdatensatz nicht gefunden." }</pre>	
Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 40: Testprotokoll für Löschung eines Produktes Fehler

Kurzbeschreibung:	Anzeigen aller Produkte nach Namen
Voraussetzung:	Ein Produkt existiert in der Datenbank
Eingabe: Get Request /api/products/sort/name	
Ausgabe: Statuscode 200	
<pre>[{ "id": 2, "shop": "lehner", "name": "ab", "image": "0739acac12228cc46c3777dbfc1223f8.png" }, { "id": 6,</pre>	

```
[
  {
    "shop": "hilpert",
    "name": "at",
    "image": null
  },
  {
    "id": 4,
    "shop": "hamill",
    "name": "et",
    "image": null
  },
  {
    "id": 3,
    "shop": "hilpert",
    "name": "similique",
    "image": "7d35cad275d67bbbed39c1e69aca53ac7.png"
  },
  {
    "id": 1,
    "shop": "hamill",
    "name": "sit",
    "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png"
  },
  {
    "id": 5,
    "shop": "hilpert",
    "name": "ut",
    "image": null
  }
]
```

Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 41: Testprotokoll für Anzeige aller Produkte nach Namen

Kurzbeschreibung:	Anzeigen aller Produkte sortiert nach Shop
Voraussetzung:	Ein Shop existiert in der Datenbank
Eingabe: Get Request /api/products/sort/shop	
Ausgabe: Statuscode 200	
<pre>[{ "id": 1, "shop": "hamill", "name": "sit", "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png" }, { "id": 4, "shop": "hamill", "name": "et", "image": null }, { "id": 3, "shop": "hilpert", "name": "similique", </pre>	

<pre> "image": "7d35cad275d67bbcd39c1e69aca53ac7.png" }, { "id": 5, "shop": "hilpert", "name": "ut", "image": null }, { "id": 6, "shop": "hilpert", "name": "at", "image": null }, { "id": 2, "shop": "lehner", "name": "ab", "image": "0739acac12228cc46c3777dbfc1223f8.png" }] </pre>	
Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 42: Testprotokoll für Anzeige aller Produkte sortiert nach Shop

Kurzbeschreibung:	Anzeigen aller Produkte gefiltert nach Shop
Voraussetzung:	Ein Shop existiert in der Datenbank
Eingabe: Post Request /api/products/filter/shop <pre> { "shop": "lener" } </pre>	
Ausgabe: Statuscode 200 <pre> [{ "id": 2, "shop": "lehner", "name": "ab", "image": "0739acac12228cc46c3777dbfc1223f8.png" }] </pre>	
Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 43: Testprotokoll für Anzeige aller Produkte gefiltert nach Shop

Kurzbeschreibung:	Anzeigen aller Produkte gefiltert nach Shop falsche Eingabe
Voraussetzung:	-
Eingabe: Post Request /api/products/filter/shop	
<pre>{ "wrong": "not a shop" }</pre>	
Ausgabe: Statuscode 422	
<pre>{ "message": "Ein Shop-Name ist erforderlich", "errors": { "shop": ["Ein Shop-Name ist erforderlich"] } }</pre>	
Datum:	
Resultat:	

Tabelle 44: Testprotokoll für Anzeigen aller Produkte gefiltert nach Shop Fehler

Kurzbeschreibung:	Anzeigen aller Bestellungen ist möglich
Voraussetzung:	Eine Bestellung existiert in der Datenbank
Eingabe: Get /api/orders	
Ausgabe: Statuscode 200	
<pre>[{ "ordered_at": "2015-03-03", "user": "Test User", "completed": false, "ordered_products": [{ "product": { "id": 1, "shop": "hamill", "name": "sit", "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png" }, "recieved_at": null, "volume": 3, "price": 6577, "price_total": 19731 }, { "product": { "id": 3, "shop": "hilpert", "name": "similique", </pre>	

```

    "image": "7d35cad275d67bbbed39c1e69aca53ac7.png"
  },
  "recieved_at": "2019-08-31",
  "volume": 5,
  "price": 3837,
  "price_total": 19185
},
{
  "product": {
    "id": 2,
    "shop": "lehner",
    "name": "ab",
    "image": "0739acac12228cc46c3777dbfc1223f8.png"
  },
  "recieved_at": null,
  "volume": 5,
  "price": 3193,
  "price_total": 15965
}
],
"order_price": 54881,
"note": "Quaerat fugit et iusto provident maiores reprehenderit aut harum."
},
{
  "ordered_at": "2022-04-03",
  "user": "Test User",
  "completed": false,
  "ordered_products": [
    {
      "product": {
        "id": 4,
        "shop": "hamill",
        "name": "et",
        "image": null
      },
      "recieved_at": "1973-08-23",
      "volume": 4,
      "price": 5671,
      "price_total": 22684
    },
    {
      "product": {
        "id": 3,
        "shop": "hilpert",
        "name": "similique",
        "image": "7d35cad275d67bbbed39c1e69aca53ac7.png"
      },
      "recieved_at": null,
      "volume": 1,
      "price": 1434,
      "price_total": 1434
    }
  ],
  "order_price": 24118,
  "note": "Aut eos adipisci reprehenderit amet. Rerum assumenda sit dolores vel laboru
m est repudiandae."
},
{
  "ordered_at": "1994-12-04",
  "user": "Test User",

```

```
[
  {
    "completed": false,
    "ordered_products": [
      {
        "product": {
          "id": 4,
          "shop": "hamill",
          "name": "et",
          "image": null
        },
        "recieved_at": null,
        "volume": 3,
        "price": 8978,
        "price_total": 26934
      },
      {
        "product": {
          "id": 5,
          "shop": "hilpert",
          "name": "ut",
          "image": null
        },
        "recieved_at": null,
        "volume": 5,
        "price": 4167,
        "price_total": 20835
      },
      {
        "product": {
          "id": 2,
          "shop": "lehner",
          "name": "ab",
          "image": "0739acac12228cc46c3777dbfc1223f8.png"
        },
        "recieved_at": null,
        "volume": 10,
        "price": 6069,
        "price_total": 60690
      },
      {
        "product": {
          "id": 1,
          "shop": "hamill",
          "name": "sit",
          "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png"
        },
        "recieved_at": "1992-12-27",
        "volume": 9,
        "price": 7440,
        "price_total": 66960
      }
    ],
    "order_price": 175419,
    "note": null
  }
]
```

Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 45: Testprotokoll für Anzeige aller Bestellungen

Kurzbeschreibung:	Anzeigen einer Bestellung ist möglich
Voraussetzung:	Eine Bestellung existiert in der Datenbank
Eingabe: Get /api/orders/1	
<div> ID of order <div>1</div> </div>	
Ausgabe: Statuscode 200	
<pre>{ "ordered_at": "2015-03-03", "user": "Test User", "completed": false, "ordered_products": [{ "product": { "id": 1, "shop": "hamill", "name": "sit", "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png" }, "recieved_at": null, "volume": 3, "price": 6577, "price_total": 19731 }, { "product": { "id": 3, "shop": "hilpert", "name": "similique", "image": "7d35cad275d67bbed39c1e69aca53ac7.png" }, "recieved_at": "2019-08-31", "volume": 5, "price": 3837, "price_total": 19185 }, { "product": { "id": 2, "shop": "lehner", "name": "ab", "image": "0739acac12228cc46c3777dbfc1223f8.png" }, "recieved_at": null, "volume": 5, "price": 3193, "price_total": 15965 }], "order_price": 54881, "note": "Quaerat fugit et iusto provident maiores reprehenderit aut harum." }</pre>	

Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 46: Testprotokoll für Anzeige einer Bestellung

Kurzbeschreibung:	Anzeigen einer Bestellung falsche Eingabe
Voraussetzung:	-
Eingabe: Get Request /api/orders/10 <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p>ID of order</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 5px;">10</div> </div>	
Ausgabe: Statuscode 404 <pre>{ "message": "Bestelldatensatz nicht gefunden." }</pre>	
Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 47: Testprotokoll für Anzeige einer Bestellung Fehler

Kurzbeschreibung:	Erstellen einer Bestellung ist möglich
Voraussetzung:	Produkt existiert in der Datenbank
Eingabe: Post Request /api/orders <pre>{ "note": "Is super important!", "ordered_at": "2000-02-02", "orderProducts": [{ "product_id": 1, "volume": 10, "price": 100 }] }</pre>	
Ausgabe: Statuscode 200 <pre>{ "ordered_at": "2000-02-02", "user": "Test User", "completed": false, "ordered_products": [{ "product": { "id": 1, </pre>	

<pre> "shop": "hamill", "name": "sit", "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png" }, "recieved_at": null, "volume": 10, "price": 100, "price_total": 1000 }], "order_price": 1000, "note": "Is super important!" } </pre>	
Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 48: Testprotokoll für Erstellung einer Bestellung

Kurzbeschreibung:	Erstellen einer Bestellung falsche Eingabe
Voraussetzung:	-
<p>Eingabe: Post Request /api/orders</p> <pre> { "note": "Is super important!", "ordered_at": "0", "orderProducts": [{ "product_id": 1, "volume": 0, "price": 0 }] } </pre>	
<p>Ausgabe: Statuscode 422</p> <pre> { "message": "Muss ein Datum im Format YYYY-MM-DD sein (and 2 more errors)", "errors": { "ordered_at": ["Muss ein Datum im Format YYYY-MM-DD sein"], "orderProducts.0.volume": ["Es ist ein Volumen von mindestens 1 erforderlich"], "orderProducts.0.price": ["Es ist ein Preis von mindestens 5 Rappen erforderlich"]] } </pre>	
Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 49: Testprotokoll für Erstellung einer Bestellung Fehler

Kurzbeschreibung:	Bearbeiten einer Bestellung ist möglich
Voraussetzung:	Eine Bestellung existiert in der Datenbank
Eingabe: Put Request /api/orders/4 <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>ID of order</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 5px;">4</div> <pre>{ "note": "Important note about order!" }</pre> </div>	
Ausgabe: Statuscode 200 <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px; background-color: #2e3436; color: #eeeeec;"> <pre>{ "ordered_at": "2000-02-02", "user": "Test User", "completed": false, "ordered_products": [{ "product": { "id": 1, "shop": "hamill", "name": "sit", "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png" }, "recieved_at": null, "volume": 10, "price": 100, "price_total": 1000 }], "order_price": 1000, "note": "Important note about order!" }</pre> </div>	
Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 50: Testprotokoll für Bearbeitung einer Bestellung

Kurzbeschreibung:	Bearbeiten einer Bestellung falsche Eingabe
Voraussetzung:	-
Eingabe: Put Request /api/orders/40 <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>ID of order</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 5px;">40</div> <pre>{ "note": "Important note about order!" }</pre> </div>	

}	
Ausgabe: Statuscode 404	
<pre>{ "message": "Bestelldatensatz nicht gefunden." }</pre>	
Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 51: Testprotokoll für Bearbeitung einer Bestellung Fehler

Kurzbeschreibung:	Löschen einer Bestellung ist möglich
Voraussetzung:	Eine Bestellung existiert in der Datenbank
Eingabe: Delete Request /api/orders/4	
<p>ID of order</p> <div>4</div>	
Ausgabe: Statuscode 204	
Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 52: Testprotokoll für Löschung einer Bestellung

Kurzbeschreibung:	Löschen einer Bestellung falsche Eingabe
Voraussetzung:	-
Eingabe: Delete Request /api/orders/40	
<p>ID of order</p> <div>40</div>	
Ausgabe: Statuscode 404	
<pre>{ "message": "Bestelldatensatz nicht gefunden." }</pre>	
Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 53: Testprotokoll für Löschung einer Bestellung Fehler

Kurzbeschreibung:	Anzeigen aller Bestellung sortiert nach Datum
Voraussetzung:	Eine Bestellung existiert in der Datenbank
Eingabe: Get Request /api/orders/sort/date/old	
Ausgabe: Statuscode 200	
<pre>[{ "ordered_at": "1994-12-04", "user": "Test User", "completed": false, "ordered_products": [{ "product": { "id": 4, "shop": "hamill", "name": "et", "image": null }, "recieved_at": null, "volume": 3, "price": 8978, "price_total": 26934 }, { "product": { "id": 5, "shop": "hilpert", "name": "ut", "image": null }, "recieved_at": null, "volume": 5, "price": 4167, "price_total": 20835 }, { "product": { "id": 2, "shop": "lehner", "name": "ab", "image": "0739acac12228cc46c3777dbfc1223f8.png" }, "recieved_at": null, "volume": 10, "price": 6069, "price_total": 60690 }, { "product": { "id": 1, "shop": "hamill", "name": "sit", "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png" }, "recieved_at": "1992-12-27", "volume": 9, "price": 7440,</pre>	

```
    "price_total": 66960
  },
  ],
  "order_price": 175419,
  "note": null
},
{
  "ordered_at": "2015-03-03",
  "user": "Test User",
  "completed": false,
  "ordered_products": [
    {
      "product": {
        "id": 1,
        "shop": "hamill",
        "name": "sit",
        "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png"
      },
      "recieved_at": null,
      "volume": 3,
      "price": 6577,
      "price_total": 19731
    },
    {
      "product": {
        "id": 3,
        "shop": "hilpert",
        "name": "similique",
        "image": "7d35cad275d67bbed39c1e69aca53ac7.png"
      },
      "recieved_at": "2019-08-31",
      "volume": 5,
      "price": 3837,
      "price_total": 19185
    },
    {
      "product": {
        "id": 2,
        "shop": "lehner",
        "name": "ab",
        "image": "0739acac12228cc46c3777dbfc1223f8.png"
      },
      "recieved_at": null,
      "volume": 5,
      "price": 3193,
      "price_total": 15965
    }
  ],
  "order_price": 54881,
  "note": "Quaerat fugit et iusto provident maiores reprehenderit aut harum."
},
{
  "ordered_at": "2022-04-03",
  "user": "Test User",
  "completed": false,
  "ordered_products": [
    {
      "product": {
        "id": 4,
        "shop": "hamill",
```

```
[
  {
    "name": "et",
    "image": null
  },
  "recieved_at": "1973-08-23",
  "volume": 4,
  "price": 5671,
  "price_total": 22684
},
{
  "product": {
    "id": 3,
    "shop": "hilpert",
    "name": "similique",
    "image": "7d35cad275d67bbed39c1e69aca53ac7.png"
  },
  "recieved_at": null,
  "volume": 1,
  "price": 1434,
  "price_total": 1434
}
],
"order_price": 24118,
"note": "Aut eos adipisci reprehenderit amet. Rerum assumenda sit dolores vel laboru
m est repudiandae."
}
]
```

Datum:	24.03.2023
Resultat:	Bestanden

Tabelle 54: Testprotokoll für Anzeige aller Bestellungen sortiert nach Datum

Kurzbeschreibung:	Anzeigen aller Bestellung sortiert nach Status
Voraussetzung:	Eine Bestellung existiert in der Datenbank
Eingabe: Get Request /api/orders/sort/new	
Ausgabe: Stautscode 200	
<pre>[{ "ordered_at": "2022-04-03", "user": "Test User", "completed": false, "ordered_products": [{ "product": { "id": 4, "shop": "hamill", "name": "et", "image": null }, "recieved_at": "1973-08-23", "volume": 4, "price": 5671, "price_total": 22684 }, { "product": {</pre>	

```

        "id": 3,
        "shop": "hilpert",
        "name": "similique",
        "image": "7d35cad275d67bbbed39c1e69aca53ac7.png"
    },
    "recieved_at": null,
    "volume": 1,
    "price": 1434,
    "price_total": 1434
  }
],
"order_price": 24118,
"note": "Aut eos adipisci reprehenderit amet. Rerum assumenda sit dolores vel laboru
m est repudiandae."
},
{
  "ordered_at": "2015-03-03",
  "user": "Test User",
  "completed": false,
  "ordered_products": [
    {
      "product": {
        "id": 1,
        "shop": "hamill",
        "name": "sit",
        "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png"
      },
      "recieved_at": null,
      "volume": 3,
      "price": 6577,
      "price_total": 19731
    },
    {
      "product": {
        "id": 3,
        "shop": "hilpert",
        "name": "similique",
        "image": "7d35cad275d67bbbed39c1e69aca53ac7.png"
      },
      "recieved_at": "2019-08-31",
      "volume": 5,
      "price": 3837,
      "price_total": 19185
    },
    {
      "product": {
        "id": 2,
        "shop": "lehner",
        "name": "ab",
        "image": "0739acac12228cc46c3777dbfc1223f8.png"
      },
      "recieved_at": null,
      "volume": 5,
      "price": 3193,
      "price_total": 15965
    }
  ],
  "order_price": 54881,
  "note": "Quaerat fugit et iusto provident maiores reprehenderit aut harum."
},

```

```
{
  "ordered_at": "1994-12-04",
  "user": "Test User",
  "completed": false,
  "ordered_products": [
    {
      "product": {
        "id": 4,
        "shop": "hamill",
        "name": "et",
        "image": null
      },
      "recieved_at": null,
      "volume": 3,
      "price": 8978,
      "price_total": 26934
    },
    {
      "product": {
        "id": 5,
        "shop": "hilpert",
        "name": "ut",
        "image": null
      },
      "recieved_at": null,
      "volume": 5,
      "price": 4167,
      "price_total": 20835
    },
    {
      "product": {
        "id": 2,
        "shop": "lehner",
        "name": "ab",
        "image": "0739acac12228cc46c3777dbfc1223f8.png"
      },
      "recieved_at": null,
      "volume": 10,
      "price": 6069,
      "price_total": 60690
    },
    {
      "product": {
        "id": 1,
        "shop": "hamill",
        "name": "sit",
        "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png"
      },
      "recieved_at": "1992-12-27",
      "volume": 9,
      "price": 7440,
      "price_total": 66960
    }
  ],
  "order_price": 175419,
  "note": null
}
```

1

Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 55: Testprotokoll für Anzeige aller Bestellungen sortiert nach Status

Kurzbeschreibung:	Anzeigen aller Bestellung gefiltert nach Status
Voraussetzung:	Eine Bestellung existiert in der Datenbank
Eingabe: Get Request /api/orders/sort/status	
Ausgabe: Statuscode 200	
<pre>[{ "ordered_at": "2015-03-03", "user": "Test User", "completed": false, "ordered_products": [{ "product": { "id": 1, "shop": "hamill", "name": "sit", "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png" }, "recieved_at": null, "volume": 3, "price": 6577, "price_total": 19731 }, { "product": { "id": 3, "shop": "hilpert", "name": "similique", "image": "7d35cad275d67bbed39c1e69aca53ac7.png" }, "recieved_at": "2019-08-31", "volume": 5, "price": 3837, "price_total": 19185 }, { "product": { "id": 2, "shop": "lehner", "name": "ab", "image": "0739acac12228cc46c3777dbfc1223f8.png" }, "recieved_at": null, "volume": 5, "price": 3193, "price_total": 15965 }], "order_price": 54881, "note": "Quaerat fugit et iusto provident maiores reprehenderit aut harum." }, { </pre>	

```
"ordered_at": "2022-04-03",
"user": "Test User",
"completed": false,
"ordered_products": [
  {
    "product": {
      "id": 4,
      "shop": "hamill",
      "name": "et",
      "image": null
    },
    "recieved_at": "1973-08-23",
    "volume": 4,
    "price": 5671,
    "price_total": 22684
  },
  {
    "product": {
      "id": 3,
      "shop": "hilpert",
      "name": "similique",
      "image": "7d35cad275d67bbbed39c1e69aca53ac7.png"
    },
    "recieved_at": null,
    "volume": 1,
    "price": 1434,
    "price_total": 1434
  }
],
"order_price": 24118,
"note": "Aut eos adipisci reprehenderit amet. Rerum assumenda sit dolores vel laboru
m est repudiandae."
},
{
  "ordered_at": "1994-12-04",
  "user": "Test User",
  "completed": false,
  "ordered_products": [
    {
      "product": {
        "id": 4,
        "shop": "hamill",
        "name": "et",
        "image": null
      },
      "recieved_at": null,
      "volume": 3,
      "price": 8978,
      "price_total": 26934
    },
    {
      "product": {
        "id": 5,
        "shop": "hilpert",
        "name": "ut",
        "image": null
      },
      "recieved_at": null,
      "volume": 5,
      "price": 4167,
```

```
[
  {
    "price_total": 20835,
    {
      "product": {
        "id": 2,
        "shop": "lehner",
        "name": "ab",
        "image": "0739acac12228cc46c3777dbfc1223f8.png"
      },
      "recieved_at": null,
      "volume": 10,
      "price": 6069,
      "price_total": 60690
    },
    {
      "product": {
        "id": 1,
        "shop": "hamill",
        "name": "sit",
        "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png"
      },
      "recieved_at": "1992-12-27",
      "volume": 9,
      "price": 7440,
      "price_total": 66960
    }
  },
  "order_price": 175419,
  "note": null
]
```

Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 56: Testprotokoll für Anzeige aller Bestellungen gefiltert nach Status

Kurzbeschreibung:	Anzeigen aller Bestellungsprodukte
Voraussetzung:	Ein bestelltes Produkt existiert in der Datenbank
Eingabe: Get Request /api/orderProduct	
Ausgabe: Statuscode 200	
<pre>[{ "order_id": 1, "order_note": "Quaerat fugit et iusto provident maiores reprehenderit aut harum.", "product": { "id": 1, "shop": "hamill", "name": "sit", "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png" }, "recieved_at": null, "volume": 3, "price": 6577, "price_total": 19731 },]</pre>	

```
{
  "order_id": 3,
  "order_note": null,
  "product": {
    "id": 4,
    "shop": "hamill",
    "name": "et",
    "image": null
  },
  "recieved_at": null,
  "volume": 3,
  "price": 8978,
  "price_total": 26934
},
{
  "order_id": 3,
  "order_note": null,
  "product": {
    "id": 5,
    "shop": "hilpert",
    "name": "ut",
    "image": null
  },
  "recieved_at": null,
  "volume": 5,
  "price": 4167,
  "price_total": 20835
},
{
  "order_id": 2,
  "order_note": "Aut eos adipisci reprehenderit amet. Rerum assumenda sit dolores vel laborum est repudiandae.",
  "product": {
    "id": 4,
    "shop": "hamill",
    "name": "et",
    "image": null
  },
  "recieved_at": "1973-08-23",
  "volume": 4,
  "price": 5671,
  "price_total": 22684
},
{
  "order_id": 1,
  "order_note": "Quaerat fugit et iusto provident maiores reprehenderit aut harum.",
  "product": {
    "id": 3,
    "shop": "hilpert",
    "name": "similique",
    "image": "7d35cad275d67bbcd39c1e69aca53ac7.png"
  },
  "recieved_at": "2019-08-31",
  "volume": 5,
  "price": 3837,
  "price_total": 19185
},
{
  "order_id": 1,
  "order_note": "Quaerat fugit et iusto provident maiores reprehenderit aut harum.",
```

```
[
  {
    "product": {
      "id": 2,
      "shop": "lehner",
      "name": "ab",
      "image": "0739acac12228cc46c3777dbfc1223f8.png"
    },
    "recieved_at": null,
    "volume": 5,
    "price": 3193,
    "price_total": 15965
  },
  {
    "order_id": 3,
    "order_note": null,
    "product": {
      "id": 2,
      "shop": "lehner",
      "name": "ab",
      "image": "0739acac12228cc46c3777dbfc1223f8.png"
    },
    "recieved_at": null,
    "volume": 10,
    "price": 6069,
    "price_total": 60690
  },
  {
    "order_id": 2,
    "order_note": "Aut eos adipisci reprehenderit amet. Rerum assumenda sit dolores vel laborum est repudiandae.",
    "product": {
      "id": 3,
      "shop": "hilpert",
      "name": "similique",
      "image": "7d35cad275d67bbcd39c1e69aca53ac7.png"
    },
    "recieved_at": null,
    "volume": 1,
    "price": 1434,
    "price_total": 1434
  },
  {
    "order_id": 3,
    "order_note": null,
    "product": {
      "id": 1,
      "shop": "hamill",
      "name": "sit",
      "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png"
    },
    "recieved_at": "1992-12-27",
    "volume": 9,
    "price": 7440,
    "price_total": 66960
  }
]
```

Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 57: Testprotokoll für Anzeige aller Bestellungsprodukte

Kurzbeschreibung:	Anzeigen eines Bestellungsproduktes ist möglich
Voraussetzung:	Ein bestelltes Produkt existiert in der Datenbank
Eingabe: Get /api/orderProduct/1	
<div>ID of ordered product</div> <div>1</div>	
Ausgabe: Statuscode 200	
<pre>{ "order_id": 1, "order_note": "Quaerat fugit et iusto provident maiores reprehenderit aut harum.", "product": { "id": 1, "shop": "hamill", "name": "sit", "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png" }, "recieved_at": null, "volume": 3, "price": 6577, "price_total": 19731 }</pre>	
Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 58: Testprotokoll für Anzeige eines Bestellungsproduktes

Kurzbeschreibung:	Anzeigen eines Bestellungsproduktes falsche Eingabe
Voraussetzung:	-
Eingabe: Get /api/orderProducts/100	
<div>ID of ordered product</div> <div>100</div>	
Ausgabe: Statuscode 404	
<pre>{ "message": "Bestellter Produktdatensatz nicht gefunden." }</pre>	
Datum:	25.03.2023
Resultat:	Bestanden

Tabelle 59: Testprotokoll für Anzeige eines Bestellungsproduktes Fehler

Kurzbeschreibung:	Erstellung eines Bestellungsproduktes ist möglich
Voraussetzung:	Eine Bestellung existiert in der Datenbank. Ein Produkt existiert in der Datenbank.
Eingabe: Post /api/orderProducts <pre>{ "order_id": 1, "product_id": 1, "volume": 5, "price": 20 }</pre>	
Ausgabe: Statuscode 200 <pre>{ "order_id": 1, "order_note": "Quaerat fugit et iusto provident maiores reprehenderit aut harum.", "product": { "id": 1, "shop": "hamill", "name": "sit", "image": "7b6f588fe13bf2df00ed9f28fbc2f0e0.png" }, "recieved_at": null, "volume": 5, "price": 20, "price_total": 100 }</pre>	
Datum:	26.03.2023
Resultat:	Bestanden

Tabelle 60: Testprotokoll für Erstellung eines Bestellungsproduktes

Kurzbeschreibung:	Erstellung eines Bestellungsproduktes falsche Eingabe
Voraussetzung:	Eine Bestellung existiert in der Datenbank. Ein Produkt existiert in der Datenbank.
Eingabe: Post /api/orderProducts <pre>{ "order_id": 1, "product_id": 1, "volume": 0, "price": 0 }</pre>	
Ausgabe: Statuscode 422 <pre>{ "message": "Es ist ein Volumen von mindestens 1 erforderlich (and 1 more error)", "errors": { "volume": ["Es ist ein Volumen von mindestens 1 erforderlich"] } }</pre>	

<pre> "price": ["Es ist ein Preis von mindestens 5 Rappen erforderlich"] } } </pre>	
Datum:	26.03.2023
Resultat:	Bestanden

Tabelle 61: Testprotokoll für Erstellung eines Bestellungsproduktes Fehler

Kurzbeschreibung:	Bearbeitung eines Bestellungsproduktes ist möglich
Voraussetzung:	Ein bestelltes Produkt existiert in der Datenbank. Eine Bestellung existiert in der Datenbank. Ein Produkt existiert in der Datenbank.
<p>Eingabe: Put Request /api/orderProducts/11</p> <p>ID of ordered product</p> <p>11</p> <pre> { "order_id": 2, "product_id": 2, "volume": 10, "price": 50, "recieved_at": "2012-12-21" } </pre>	
<p>Ausgabe: Statuscode 200</p> <pre> { "order_id": 2, "order_note": "Aut eos adipisci reprehenderit amet. Rerum assumenda sit dolores vel la borum est repudiandae.", "product": { "id": 2, "shop": "lehner", "name": "ab", "image": "0739acac12228cc46c3777dbfc1223f8.png" }, "recieved_at": "2012-12-21", "volume": 10, "price": 50, "price_total": 500 } </pre>	
Datum:	26.03.2023
Resultat:	Bestanden

Tabelle 62: Testprotokoll für Bearbeitung eines Bestellungsproduktes

Kurzbeschreibung:	Bearbeitung eines Bestellungsproduktes falsche Eingabe
Voraussetzung:	Ein bestelltes Produkt existiert in der Datenbank. Eine Bestellung existiert in der Datenbank. Ein Produkt existiert in der Datenbank.
Eingabe: Put /api/orderProducts/11 <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p>ID of ordered product</p> <div style="border: 1px solid #ccc; padding: 5px; margin: 5px 0;">11</div> <pre>{ "order_id": 2, "product_id": 20, "volume": 10, "price": 50, "recieved_at": "2012-12-21" }</pre> </div>	
Ausgabe: Statuscode 422 <div style="background-color: #2e3436; color: #eeeeec; padding: 10px; margin: 10px 0;"> <pre>{ "message": "Produktdatensatz nicht gefunden", "errors": { "product_id": ["Produktdatensatz nicht gefunden"] } }</pre> </div>	
Datum:	26.03.2023
Resultat:	Bestanden

Tabelle 63: Testprotokoll für Bearbeitung eines Bestellungsproduktes Fehler

Kurzbeschreibung:	Löschung eines Bestellungsproduktes ist möglich
Voraussetzung:	Ein bestelltes Produkt existiert in der Datenbank
Eingabe: Delete Request /api/orderProducts/11 <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p>ID of ordered product</p> <div style="border: 1px solid #ccc; padding: 5px; margin: 5px 0;">11</div> </div>	
Ausgabe: Statuscode 204	
Datum:	26.03.2023
Resultat:	Bestanden

Tabelle 64: Testprotokoll für Löschung eines Bestellungsproduktes

Kurzbeschreibung:	Löschung eines Bestellungsproduktes falsche Eingabe
Voraussetzung:	-
Eingabe: Delete Request /api/orderProducts/111	
<div style="background-color: #f8d7da; padding: 10px; border: 1px solid #f5c6cb;"> ID of ordered product </div> <div style="border: 1px solid #f5c6cb; padding: 5px; margin-top: 5px;"> 111 </div>	
Ausgabe: Statuscode 404	
<pre>{ "message": "Bestellter Produktdatensatz nicht gefunden." }</pre>	
Datum:	26.03.2023
Resultat:	Bestanden

Tabelle 65: Testprotokoll für Löschung eines Bestellungsproduktes Fehler

Eingesetzte Testmittel

Das Testen findet mit der Swagger-Dokumentation statt. Es wird geprüft, ob die vorher bestimmten Requests die korrekten Responses generieren.

Rahmenbedingungen

XAMPP-Server muss eingeschaltet sein.

Mysql Datenbank muss existieren und korrekt mit Laravel verbunden sein.

Datenbank muss Daten enthalten (Migration und Seeding vollziehen).

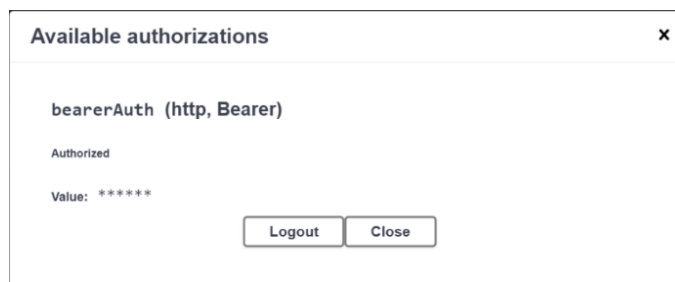
```
php artisan migrate:fresh
```

```
php artisan db:seed
```

Laravel muss Server muss aktiviert sein.

```
php artisan serve
```

Es muss ein User registriert werden und das zurückgegebene Bearer-Token muss in die Swagger-Autorisierung eingefügt werden.



Auswerten

Reflexion der Vorgehensweise

Informieren:

Dank meiner Erfahrung in der Arbeit mit Laravel konnte ich mir eine sehr gute Vorstellung vom Auftrag aus den Vorgaben bilden. Es war mir möglich viele nötige Informationen und Antworten auf Fragen, die anfallen würden, abzusehen und im Vorn herein einzuholen. Ich fühlte mich dadurch gut vorbereitet nach dem Informations-Schritt.

Planen:

Bei der Planung sind mir ganz klar einige Fehler unterlaufen. Obwohl ich den Aufwand für die meisten Schritte richtig antizipiert habe, stechen zwei grosse Misskalkulationen heraus. Die Fehldinschätzung für das Erstellen der Mockups ist etwas überraschend, da ich mir eigentlich bewusst bin, dass diese viel Zeit in Anspruch nehmen. Beim Testkonzept hatte ich Probleme, weil es sehr offen war, was genau getestet werden musste. Ich hätte von Anfang an besser abklären sollen, was genau erwartet wird.

Entscheiden:

Hier kam mir zugute, dass ich mich an den Laravel-Konventionen orientieren konnte, was mir viele Entscheidungen in der Umsetzung abnahm. Für die Entscheidungen die ausserhalb von Coding-Style lagen, konnte ich mit Hilfe meines Entscheidungsprozesses, zum Beispiel mit Entscheidungstabellen, gute Entscheidungen treffen.

Realisieren:

Meine Erfahrung mit Laravel erlaubte es mir hier schnell und gezielt vorzugehen und verlorene Zeit aufzuholen. Jedoch zeigte sich hier meine zweite grössere Fehl Kalkulation. Obwohl ich wusste, dass Swagger-Annotationen immer viel Zeit verschlingen, habe ich viel zu wenig Zeit für diese eingeplant.

Kontrollieren:

Die Kontrolle verlief erwartungsgemäss ereignislos. Bei der Erstellung der API arbeitete ich eng mit Insomnia, um die Funktionalität sicherzustellen. Die Tests waren alle beim ersten Versuch bestanden.

Bewertung des Produktes

Die Laravel-API und die Swagger-Implementation funktionieren beide korrekt. Insbesondere bin ich zufrieden mit der Laravel-API. Der Code ist sehr sauber nach OOP aufgebaut und nach Laravel-Konventionen umgesetzt.

Einige Punkte, die noch Verbesserungspotential haben, sind herauszustreichen.

In Bezug auf Swagger könnte immer noch mehr gemacht werden. Zum Beispiel Frontend-Fehlermeldungen, auf die ich aus Zeitgründen hier verzichtet habe. In dieser Anwendung ist das jetzige Level aber zufriedenstellend.

Ich habe Zweifel, ob der zusätzliche Programmieraufwand, um die Normalisierung der Datenbank zu erreichen, gerechtfertigt ist. Da es aber eine Vorgabe war, habe ich es so umgesetzt.

Falls die Applikation ausgebaut oder multilingual gemacht werden soll, würde es Sinn machen, die Fehlermeldungen in Sprach-Files zu organisieren.

Bei einer Vergrößerung der Applikation würde auch eine Textsuche mit Scout Sinn machen.

Persönliches Schlusswort und Bilanz

Was ich sicher anders machen würde, ist das Erstellen des Zeitplans. Ich würde eine intuitivere Methode benutzen, als zu versuchen, mich an einer Excel-Tabelle zu orientieren. Was besonders heraus sticht, ist das Testen und die Swagger-Implementation nicht mehr zu unterschätzen.

Sehr zufrieden bin ich hingegen mit der Umsetzung der Laravel-API und dem Arbeiten mit Laravel-Konventionen.

Glossar

A

ACID · *Atomicity, Consistency, Isolation, Durability*

API · *Programmierschnittstelle*

B

Bearer-Token · *HTTP Authentifizierungs-Schema*

Blob · *Binary large object*

C

Composer · *Composer ist ein anwendungsorientierter Paketmanager für die Skriptsprache PHP.*

CRUD · *Akronym CREATE, READ, UPDATE, DELETE*

I

Insomnia · *API-Plattform*

IPERKA-Methode · *Projektmethode*

J

JSON · *JavaScript Object Notation*

L

Laravel · *PHP-Framework*

M

Multipart-Formdata · *Request-Typ zur Sendung von Daten*

O

ORM · *Object Relational Mapper*

P

Pkorg · *Prüfungskommission für die IPA*

Postman · *API-Plattform*

S

Scout · *Voll-Textsuche für Laravel-Modelle*

similar_text · *PHP-Methode für Textvergleiche*

Swagger · *Regeln und Programme zur Beschreibung einer API*

X

XAMPP · *Open-Source Web-Server*

Quellenverzeichnis

Literaturverzeichnis

- DarkaOnLine. (17. 03 2023). *DarkaOnLine/L5-Swagger: OpenApi or Swagger integration to Laravel*. Von DarkaOnLine/L5-Swagger: OpenApi or Swagger integration to Laravel:
<https://github.com/DarkaOnLine/L5-Swagger> abgerufen
- Laraveldaily. (21. 03 2023). *laraveldaily.com*. Von laraveldaily.com: <https://laraveldaily.com/post/laravel-api-override-404-error-message-route-model-binding> abgerufen

Abbildungsverzeichnis

Abbildung 1: Aktivitätsdiagramm Produkt erstellen	27
Abbildung 2: Klassendiagramm	28
Abbildung 3: Aufbaudiagramm Laravel Konzeptionell	29
Abbildung 4: Lifecycle-Diagramm Request.....	29
Abbildung 5: Mock-Up Swagger	30
Abbildung 6: Mock-Up Swagger Schemas	31
Abbildung 7: Aufbaudiagramm Swagger konzeptionell	32
Abbildung 8: Aufbaudiagramm Laravel	35
Abbildung 9: Swagger GUI	41

Tabellenverzeichnis

Tabelle 1 - Adressinformation Auszubildender	8
Tabelle 2 – Ausführungszeitraum	8
Tabelle 3 – Termine	8
Tabelle 4 - Involvierte Personen	8
Tabelle 5 - Arbeitsprotokoll Tag 1	14
Tabelle 6 - Arbeitsprotokoll Tag 2	15
Tabelle 7 - Arbeitsprotokoll Tag 3	16
Tabelle 8 - Arbeitsprotokoll Tag 4	17
Tabelle 9 - Arbeitsprotokoll Tag 5	18
Tabelle 10 - Arbeitsprotokoll Tag 6	19
Tabelle 11 - Arbeitsprotokoll Tag 7	20
Tabelle 12 - Arbeitsprotokoll Tag 8	21
Tabelle 13 - Arbeitsprotokoll Tag 9	22
Tabelle 14 - Arbeitsprotokoll Tag 10	23
Tabelle 15 - Tätigkeiten Liste	26
Tabelle 16: Entscheidungsmatrix Bild.....	33

Tabelle 17: Entscheidungsmatrix Textsuche	34
Tabelle 18: Laravel Ressource	36
Tabelle 19 - Testprotokoll für User Registrierung	44
Tabelle 20: Testprotokoll für User Registrierung Fehler	44
Tabelle 21: Testprotokoll für User Login Fehler	45
Tabelle 22: Testprotokoll für User Login Fehler	45
Tabelle 23: Testprotokoll für Anzeige aller Shops	47
Tabelle 24: Testprotokoll für Shop Anzeige.....	47
Tabelle 25: Testprotokoll für Shop Anzeige Fehler	48
Tabelle 26: Testprotokoll für Erstellung eines Shops	48
Tabelle 27: Testprotokoll für Erstellung eines Shops Fehler	49
Tabelle 28: Testprotokoll für Bearbeitung eines Shops.....	49
Tabelle 29: Testprotokoll für Bearbeitung eines Shops Fehler	50
Tabelle 30: Testprotokoll für Löschung eines Shops	50
Tabelle 31: Testprotokoll für Löschung eines Shops Fehler	51
Tabelle 32: Testprotokoll für Anzeige aller Produkt	52
Tabelle 33: Testprotokoll für Anzeige eines Produktes	52
Tabelle 34: Testprotokoll für Anzeige eines Produktes mit Fehler.....	53
Tabelle 35: Testprotokoll für Erstellung eines Produktes.....	53
Tabelle 36: Testprotokoll für Erstellung eines Produktes Fehler	54
Tabelle 37: Testprotokoll für Bearbeitung eines Produktes	55
Tabelle 38: Testprotokoll für Bearbeitung eines Produktes Fehler.....	56
Tabelle 39: Testprotokoll für Löschung eines Produktes	57
Tabelle 40: Testprotokoll für Löschung eines Produktes Fehler	57
Tabelle 41: Testprotokoll für Anzeige aller Produkte nach Namen.....	58
Tabelle 42: Testprotokoll für Anzeige aller Produkte sortiert nach Shop	59
Tabelle 43: Testprotokoll für Anzeige aller Produkte gefiltert nach Shop	59
Tabelle 44: Testprotokoll für Anzeige aller Produkte gefiltert nach Shop Fehler	60
Tabelle 45: Testprotokoll für Anzeige aller Bestellungen	62
Tabelle 46: Testprotokoll für Anzeige einer Bestellung.....	64
Tabelle 47: Testprotokoll für Anzeige einer Bestellung Fehler.....	64
Tabelle 48: Testprotokoll für Erstellung einer Bestellung	65
Tabelle 49: Testprotokoll für Erstellung einer Bestellung Fehler	65
Tabelle 50: Testprotokoll für Bearbeitung einer Bestellung	66
Tabelle 51: Testprotokoll für Bearbeitung einer Bestellung Fehler	67
Tabelle 52: Testprotokoll für Löschung einer Bestellung	67
Tabelle 53: Testprotokoll für Löschung einer Bestellung Fehler	67
Tabelle 54: Testprotokoll für Anzeige aller Bestellungen sortiert nach Datum	70
Tabelle 55: Testprotokoll für Anzeige aller Bestellungen sortiert nach Status	73
Tabelle 56: Testprotokoll für Anzeige aller Bestellungen gefiltert nach Status	75
Tabelle 57: Testprotokoll für Anzeige aller Bestellungsprodukte	77
Tabelle 58: Testprotokoll für Anzeige eines Bestellungsproduktes	78

Tabelle 59: Testprotokoll für Anzeige eines Bestellungsproduktes Fehler	78
Tabelle 60: Testprotokoll für Erstellung eines Bestellungsproduktes	79
Tabelle 61: Testprotokoll für Erstellung eines Bestellungsproduktes Fehler	80
Tabelle 62: Testprotokoll für Bearbeitung eines Bestellungsproduktes	80
Tabelle 63: Testprotokoll für Bearbeitung eines Bestellungsproduktes Fehler	81
Tabelle 64: Testprotokoll für Löschung eines Bestellungsproduktes	81
Tabelle 65: Testprotokoll für Löschung eines Bestellungsproduktes Fehler	82

Anhang

PROJEKTJOURNAL	90
GESPRÄCHSPROTOKOLL VOM 20. MÄRZ 2023	90
GESPRÄCHSPROTOKOLL VOM 27. MÄRZ 2023	90
ERSTER EXPERTENBESUCH	91
ZWEITER EXPERTENBESUCH	92
CODE INFO	94
FILENAME	94
ZUSÄTZLICHE MANUALS, SKRIPTS UND WEITERES	95
HANDBUCH LARAVEL.....	95
HANDBUCH SWAGGER FÜR LARAVEL	95

Projektjournal

Gesprächsprotokoll vom 20. März 2023

Teilnehmer: Julien Rädler, Domenik Hofer

Datum, Zeit, Ort: 20.03.2023, 11:00, twofold academy AG, Thurgauerstrasse 54, 8050 Zürich

Besprechungsnotizen:

Julien: Swagger kann keine Bilder in einer PUT Methode senden, ist es ok Post zu benutzen?

Domenik: Nein.

Julien: Wie kann ich es mit PUT senden?

Domenik: Put als Request-Body Methode senden.

Gesprächsprotokoll vom 27. März 2023

Teilnehmer: Julien Rädler, Domenik Hofer

Datum, Zeit, Ort: 27.03.2023, 11:00, twofold academy AG, Thurgauerstrasse 54, 8050 Zürich

Besprechungsnotizen:

Julien: Einige Routen sind nicht wirklich auf Fehler testbar (Index Routen), wie soll ich dort vorgehen?

Domenik: Nur positive Variante testen.

Julien: Wie muss ich die Überzeit vom Wochenende deklarieren?

Domenik: Im Tagesprotokoll erwähnen und als Überzeit angeben.

Erster Expertenbesuch

Teilnehmer: Julien Rädler (Kandidat), Hansruedi Menzi (HEX), Sascha Gysel (NEX), Domenik Hofer (VF)

Datum, Zeit, Ort: 15.03.2023, 16:00, twofold academy AG, Thurgauerstrasse 54, 8050 Zürich

Besprechungsnotizen:

HEX:

- Wollte, dass alle Anwesenden sich kurz vorstellen
- Wollte Arbeitsplatz sehen
- Fragte, ob Aufgabenstellung klar sei oder es noch Fragen gäbe
- Empfehlungen HEX:
 - o Erwähnte, dass der Quellenachweis in der Doku nicht vergessen gehen sollte. Es muss klar gezeigt werden, woher die Infos kommen
 - o Nicht nur generelle Links ins Quellenverzeichnis, sondern genau beschreiben, wo Infos gefunden wurden
 - o Inhaltsverzeichnis nicht vergessen
 - o Kopf und Fusszeile mit Druckdatum und Name nicht vergessen
 - o Glossar soll alphabetisch sortiert sein, mit korrekten Erläuterungen von Fachbegriffen
 - o Alles ins Glossar, was für eine nicht-Fachperson unverständlich wäre
 - o Rechtschreibung und Grammatik beachten
 - o Wichtigste Dokumente für die IPA: Vor allem QV-Leitfaden, insbesondere Kapitel 3.3; keine Plagiate.
 - o Fremdarbeiten (Libraries o.Ä.) können eingebaut werden, müssen aber im Quellenverzeichnis erwähnt sein
 - o Abgemachte Termine beachten
 - o Bei Problemen HEX anrufen: 079 242 08 37
 - o Zeitplan soll Kandidat ins pkOrg in die History hochladen
 - o Wird beim 2. Expertenbesuch besprochen
 - o Zeitplan muss auf A4 ausgedruckt gut lesbar sein → überprüfen
 - o 30-40% sollen für die Doku eingeplant werden und sollte nicht unterschätzt werden
 - o Arbeiten und Dokumentieren Parallel
 - o Pufferzeit einplanen
 - o Falls Termin der Präsi nicht machbar melden → Wird dann aber erst nach Ostern möglich sein.

Kandidat:

- Was, wenn Grossteil des Geschriebenen Eigenleistung ist. Was muss ins Quellenverzeichnis?
 - o Antwort HEX:
 - o Wenn z.B. die 3. Normalform oder IPERKA erklärt wird, soll die Quelle zu diesen Informationen ins Verzeichnis. Sonst sicher generell die Laravel-Doku.

Zweiter Expertenbesuch

Teilnehmer: Julien Rädler (Kandidat), Hansruedi Menzi (HEX), Sascha Gysel (NEX), Domenik Hofer (VF)

Datum, Zeit, Ort: 22.03.2023, 13:30, twofold academy AG, Thurgauerstrasse 54, 8050 Zürich

Besprechungsnotizen:

- Wie gehts mental/körperlich?
 - o Gut
- Wie läuft es mit der Arbeit?
 - o Aktuell Verzug von ca. 8 Stunden wegen Testkonzept, sollte aber aufholbar sein, da andere Tasks weniger Zeit brauchen sollten/könnten
- Zeitplan zeigen
- Hinweis: Zeitplan ausgedruckt hat noch einige Fehler im Raster
- Schreibfehler in Zeitplan
- Zeitplan: Aufbaudiagramm → was ist das? Zeigen in Doku
 - o Ist ein Lifecycle Diagramm aber nicht komplett, deshalb umbenannt
 - o Wollte nur Eigenleistung an Laravel zeigen damit und nicht ganzes System
- Journal von 4. Tag zeigen. Sei in Ordnung, passt HEX und NEX. Sei schon reingeschrieben → gut.
- NEX: unbedingt Rechtschreibung korrigieren + durchlesen lassen
- In eigenen Worten beschreiben, was Projekt ist?
 - o API in Laravel die Verwaltung von online Shops und zugehörigen Produkten zu Bestellungen zusammenführt.
- Onlineshops von uns / oder welche /wie viele?
 - o Onlineshop kann alles sein, was Namen hat
 - o interagiert aber nicht mit API oder sonstigem
- Welche Tools?
 - o Laravel und Swagger
 - o MySQL Datenbank auf XAMPP Server
- Code zeigen
- Routen gezeigt, API-Ressource-Route erklärt
- Shop Controller gezeigt mit Swagger Annotation. "index"-Funktion gezeigt
- Model und Ressource erwähnt
- "store"-Funktion gezeigt
- StoreShopRequest gezeigt inkl. Validierung + Error Messages
- ShopRessource gezeigt, und erklärt was für sinnvoll erachtet, was zurückgegeben werden soll
- Shop Model gezeigt mit Relations → praktisch, um Controller klein zu behalten
- Authentifizierung über Sanctum → gezeigt bei Routen
- hast du auch Bearer Token verwendet?
 - o Ja
- Hast du auch View?
 - o Nein, gebe nur JSON zurück
- Wo testest du das?
 - o Swagger Annotationen funktionieren → gezeigt
- Wieso 204 statt 200 bei Löschen?
 - o Bei Löschen Standard
- Was sonst verwendet zum Testen?
 - o Insomnia
- Wie viel Zeit für Fehlerfälle und wie viel für gut laufenden Code?
 - o 5-10% funktioniert, Rest für Bugfixing

- Grund das du so programmierst?
 - o Liegt in der Natur, Knowns, Unknowns und unknown Unknowns
- Wem nützt das, dass du alle Ausnahmefälle abfängst
 - o Kommt auf verlangtes Endergebnis an. Wenn komplettes Endergebnis, welches erweiterbar sein soll, ist auch befriedigender
- Möchte Code sehen, wie du auf DB kommst, auch wenn vorprogrammiert:
 - o Zeigt Migrationen
 - o Zeigt Factories, um Testdaten zu erstellen
- Wie testest du?
 - o Test ist einfach Testkonzept, nichts automatisiert, Tabelle
- Testkonzept zeigen?
 - o Generelle Anforderungen für alle Funktionen geschrieben zuerst
 - o Jeder End Point ein Testfall
 - o Wären 400 Testfälle, deshalb 5 Kriterien, die in jedem Test erfüllt sein müssen
- Zeigen, wo Doku und Code versioniert mit restore-Möglichkeit
 - o Alles im gleichen Git-Repo inkl. Doku
- Stand Code von gestern zeigen
 - o Konnte Code von gestern zeigen
- Sieht man Datum von Commit irgendwo?
 - o Konnte es in History-Übersicht zeigen
- Doku auch versioniert?
 - o Auch in Git
- Du hattest beim letzten Expertenbesuch Unsicherheiten zu Quellen, konntest du diese lösen?
 - o Habe nicht so viele Quellen
- Beispiel?
 - o Fussnote auf Seite: 37
- Zeige diese Fussnote im Quellenverzeichnis
 - o Scheint noch nicht generiert zu sein
 - o Muss noch gemacht werden vor Abgabe!
- Zeige Diagramme
 - o s. 30 Swagger-Diagramm
 - o s.38 Lifecycle-Diagramm Laravel
 - o s.37 Klassendiagramm + Aufbaudiagramm
 - o s.36 Activity Diagramm
- Umfang Doku 30-40 %
- Erinnerung: hochladen nicht nur Doku auch alle Programme und Scripts!
- Präsi Termin bleibt so wie ist
- Vortrag 15-20 min, nicht mehr, nicht weniger, keine Sekunde
- Vortrag hochdeutsch
- Demo auf CH-DE
- Fachgespräch 30-60min, pro Thema max.10min
- Abgabetermin, 18:00:00:00, lieber vorher schon, sonst Notenabzug
- Bewertung Arbeit VF bis 31.03.
- Fachfragen VF bis 31.03.
 - o Diagramme noch separat als Bilddateien abgegeben werden sollen:
- Programmcode alles im Anhang oder separate Dateien möglich
- Laravel-Projekt am besten als ZIP hochladen
- Diagramme nicht separat in Dateien

Code Info

Filename

Der code des gesamten Projektes wir separat als ZIP-Datei mit gegeben -> ipa.zip

Zusätzliche Manuals, Skripts und weiteres

Handbuch Laravel

<https://laravel.com/docs/10.x>

Handbuch Swagger für Laravel

<https://github.com/DarkaOnLine/L5-Swagger>