**Individuelle Praktische Arbeit**

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

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

**Inhaltsverzeichnis**

[Teil 1: Umfeld und Ablauf 5](#_Toc529881966)

[Aufgabenstellung 5](#_Toc529881967)

[Projektaufbauorganisation 7](#_Toc529881968)

[Vorkenntnisse 8](#_Toc529881969)

[Vorarbeiten 8](#_Toc529881970)

[Benützte Firmenstandards 8](#_Toc529881971)

[Zeitplan 9](#_Toc529881972)

[Arbeitsprotokoll 10](#_Toc529881973)

[Teil 2: Projekt 22](#_Toc529881974)

[Kurzfassung 22](#_Toc529881975)

[Informieren 23](#_Toc529881976)

[Ausgangslage 23](#_Toc529881977)

[Verstandene Aufgabenstellung und Ziel der Arbeit 23](#_Toc529881978)

[Verfeinerung des Auftrages 23](#_Toc529881979)

[Projektumfeld und Systemgrenzen 23](#_Toc529881980)

[Planen 24](#_Toc529881981)

[Verwendete Projektmanagementmethode 24](#_Toc529881982)

[Versionierung und Datensicherheit 24](#_Toc529881983)

[Priorisierung der Tätigkeiten 24](#_Toc529881984)

[Kernfeature ABC 24](#_Toc529881985)

[Use-Case für Abc 24](#_Toc529881986)

[Klassendiagramm 25](#_Toc529881987)

[GUI-Design / Mockup 25](#_Toc529881988)

[Konzeptioneller Aufbau 25](#_Toc529881989)

[Kernfeature ZYZ 25](#_Toc529881990)

[Use-Case für zyz 25](#_Toc529881991)

[Klassendiagramm 26](#_Toc529881992)

[GUI-Design / Mockup 26](#_Toc529881993)

[Konzeptioneller Aufbau 26](#_Toc529881994)

[Geplantes Vorgehen für die Qualitätssicherung 26](#_Toc529881995)

[Anmerkungen zum Zeitplan 26](#_Toc529881996)

[Begründung für Abweichungen zum Zeitplan 27](#_Toc529881997)

[Entscheiden 28](#_Toc529881998)

[Beschreibung Variante abc 28](#_Toc529881999)

[Beschreibung Variante xyz 28](#_Toc529882000)

[Kurzbeschreibung der Entscheidungskriterien 28](#_Toc529882001)

[Variantenvergleich und Entscheidungsmatrix 29](#_Toc529882002)

[Begründung zum Entscheid 29](#_Toc529882003)

[Realisieren 30](#_Toc529882004)

[Abbildung des Gesamtsystems 30](#_Toc529882005)

[Interaktionen zwischen den Teilsystemen 30](#_Toc529882006)

[Vorgehensweise 30](#_Toc529882007)

[Grundaufbau / Architetkur des Projektes 30](#_Toc529882008)

[Anbindung an die Datenbank 30](#_Toc529882009)

[Implementierung des Kernfeatures ABC 31](#_Toc529882010)

[Funktionsweise des Kernfeatures ABC 31](#_Toc529882011)

[Ziel des Kernfeatures ABC 31](#_Toc529882012)

[Effektives GUI / Produkt 31](#_Toc529882013)

[Implementierung des Kernfeatures zyz 32](#_Toc529882014)

[Funktionsweise des Kernfeatures ABC 32](#_Toc529882015)

[Ziel des Kernfeatures ABC 32](#_Toc529882016)

[Effektives GUI / Produkt 32](#_Toc529882017)

[Kontrollieren 33](#_Toc529882018)

[Beschreibung der Randbedingungen / Testanlage (Umfeld) 33](#_Toc529882019)

[Eingesetzte Testmittel und -methoden 33](#_Toc529882020)

[Beschreibung der Testszenarien 33](#_Toc529882021)

[Testprotokoll zum Testszenario abc 34](#_Toc529882022)

[Testprotokoll zum Testszenario xyz 34](#_Toc529882023)

[Bugfixing 35](#_Toc529882024)

[Testprotokoll nach dem Bugfixing 35](#_Toc529882025)

[Auswerten 36](#_Toc529882026)

[Reflexion der Vorgehensweise 36](#_Toc529882027)

[Bewertung des Produktes 36](#_Toc529882028)

[Persönliches Schlusswort und Bilanz 36](#_Toc529882029)

[Glossar 37](#_Toc529882030)

[Quellenverzeichnis 40](#_Toc529882031)

[Anhang 42](#_Toc529882032)

Teil 1: Umfeld und Ablauf

# Aufgabenstellung

Titel der Arbeit

Bestellungs-Übersicht | API-Backend in Laravel (PHP-Framework).[[1]](#footnote-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]](#footnote-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
* 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
* Email
* Passwort

Orders:

* Bestellungen sollen CRUD Möglichkeiten haben
* Bestellungen beinhalten:
* Bestelldatum
* User, der die Bestellung eingetragen hat (siehe "Users" oben)
* Produkte (siehe "Products" unten)
* Erhaltdatum 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 Erhaltdatum eingetragen haben. Bis dahin bleibt dieser "Offen". Dieser Status kann dynamisch aus der Datenbank ausgelesen und muss in keiner Tabelle erfasst werden. Das Erhaltdatum 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]](#footnote-3)

Mitteln und Methoden

XAMPP, Laravel, PHP, Postman/Insomnia.[[4]](#footnote-4)

Neue Lerninhalte

Keine.[[5]](#footnote-5)

Arbeiten in den letzten 6 Monaten

Ausschliesslich Aufgaben und Projekte in Laravel.[[6]](#footnote-6)

# Projektaufbauorganisation

Projektorganisation

|  |  |
| --- | --- |
| Auszubildender | Lehrbetrieb |
| Julien Rädler  Im Dreispitz 235  8050, Zürich | Asperger 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]](#footnote-7)

# Vorarbeiten

Keine.[[8]](#footnote-8)

# Benützte Firmenstandards

Dokumentation:

Vorlage der twofold AG.

Coding-Vorgaben:

Github zur Speicherung des Projektes online. [[9]](#footnote-9)

**Zeitplan**

# 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 soweit 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, unteranderem weil die Autorisierung problemlos war. |

Tabelle 11 - Arbeitsprotokoll Tag 7

Montag, xx. Monat yyyy

|  |  |
| --- | --- |
| Tagesziele gemäss Zeitplan |  |
| Ungeplante Arbeiten /  geleistete Überstunden |  |
| Erreichte Ziele, Erfolgserlebnisse |  |
| Herausforderungen Probleme |  |
| Lösungen |  |
| Durchgeführte Tests |  |
| Wissensbeschaffung |  |
| Beanspruchte Hilfeleistung |  |
| Vergleich mit dem Soll-Zeitplan |  |
| Persönliche Tagesreflexion |  |

Tabelle 12 - Arbeitsprotokoll Tag 8

Montag, xx. Monat yyyy

|  |  |
| --- | --- |
| Tagesziele gemäss Zeitplan |  |
| Ungeplante Arbeiten /  geleistete Überstunden |  |
| Erreichte Ziele, Erfolgserlebnisse |  |
| Herausforderungen Probleme |  |
| Lösungen |  |
| Durchgeführte Tests |  |
| Wissensbeschaffung |  |
| Beanspruchte Hilfeleistung |  |
| Vergleich mit dem Soll-Zeitplan |  |
| Persönliche Tagesreflexion |  |

Tabelle 13 - Arbeitsprotokoll Tag 9

Montag, xx. Monat yyyy

|  |  |
| --- | --- |
| Tagesziele gemäss Zeitplan |  |
| Ungeplante Arbeiten /  geleistete Überstunden |  |
| Erreichte Ziele, Erfolgserlebnisse |  |
| Herausforderungen Probleme |  |
| Lösungen |  |
| Durchgeführte Tests |  |
| Wissensbeschaffung |  |
| Beanspruchte Hilfeleistung |  |
| Vergleich mit dem Soll-Zeitplan |  |
| Persönliche Tagesreflexion |  |

Tabelle 14 - Arbeitsprotokoll Tag 10

Montag, xx. Monat yyyy

|  |  |
| --- | --- |
| Tagesziele gemäss Zeitplan |  |
| Ungeplante Arbeiten /  geleistete Überstunden |  |
| Erreichte Ziele, Erfolgserlebnisse |  |
| Herausforderungen Probleme |  |
| Lösungen |  |
| Durchgeführte Tests |  |
| Wissensbeschaffung |  |
| Beanspruchte Hilfeleistung |  |
| Vergleich mit dem Soll-Zeitplan |  |
| Persönliche Tagesreflexion |  |

Tabelle 15 - Arbeitsprotokoll Tag 11

Montag, xx. Monat yyyy

|  |  |
| --- | --- |
| Tagesziele gemäss Zeitplan |  |
| Ungeplante Arbeiten /  geleistete Überstunden |  |
| Erreichte Ziele, Erfolgserlebnisse |  |
| Herausforderungen Probleme |  |
| Lösungen |  |
| Durchgeführte Tests |  |
| Wissensbeschaffung |  |
| Beanspruchte Hilfeleistung |  |
| Vergleich mit dem Soll-Zeitplan |  |
| Persönliche Tagesreflexion |  |

Tabelle 16 - Arbeitsprotokoll Tag 12

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-Mehtode 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 wird anhand der Testspezifikation getestet.

Die Auswertung beendet das Projekt.

Ergebnis

Es wurde eine API mt dem PHP-Framework Laravel erstellt, auf die über Swagger zugegriffen werden kann. Diese API erlaubt es sich als Nutzer zu registrieren und anzumalden. 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

Authorizierung 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. Routes für die Funktionen der API und die dazugehörende Controller-Logik. 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 würde könnte man argumentieren, dass Redundanzen in der Datenbank die Logik vereinfach 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 Zeitpukt ist keine Veröffentlichung geplant.

# Planen

## Verwendete Projektmanagementmethode

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 |
| Usecases beschreiben | Niedrig |
| Authentifizierung sicherstellen | Hoch |
| Swagger implementieren | Hoch |
| Tests durchführen | Hoch |
| Dokumentation erstellen | Normal |

Tabelle 18 - Tätigkeitenliste

## Kernfeature Laravel API

User sollen sich Registrieren und Anmelden können. Daten sollen gespeichert und erhallten werden können. Die Daten sollen im JSON-Format ausgegeben werden. Authentifizierung und Datensicherheit sind gewährleistet. Es sollen verständliche Fehlermeldungen ausgegeben warden.

### Aktivitätsdiagramm Laravel API

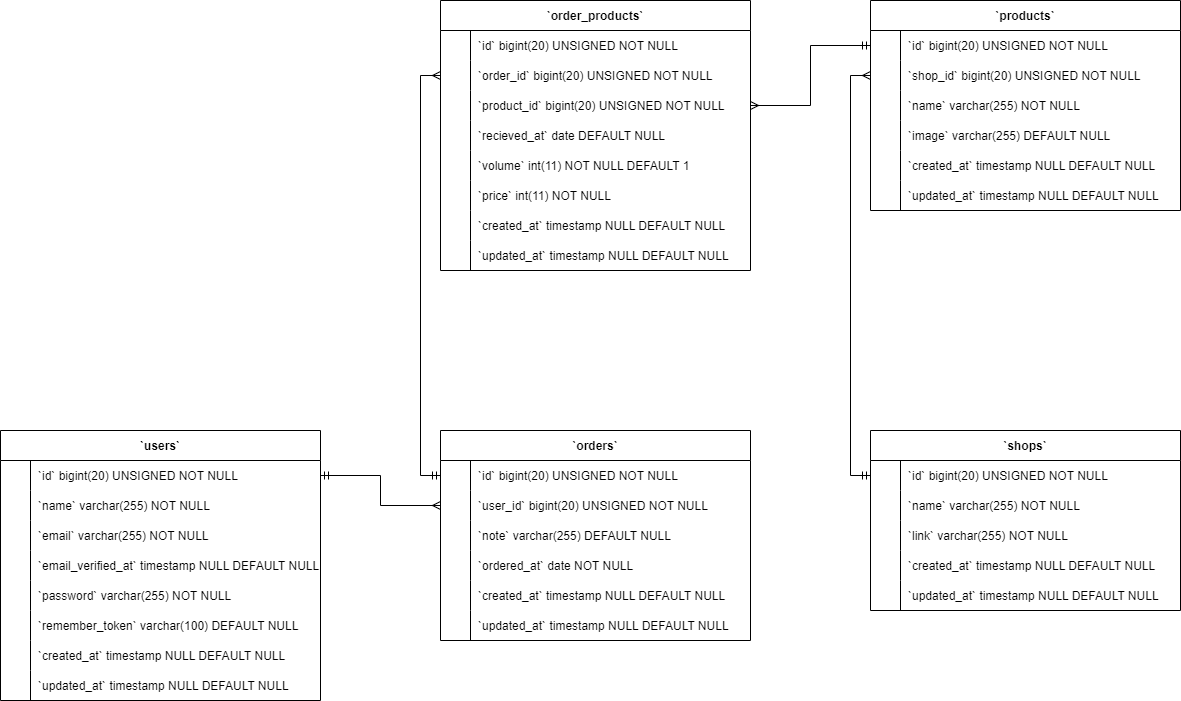
Bei dieser API wird jede Aktivität stets von einem Benutzer angestossen werden. Jeder Endpoint representiert dabei eine Aktivität. Wie zum Beispiel das Hinzufügen von Produkten:

Ein Bild, das Diagramm enthält.

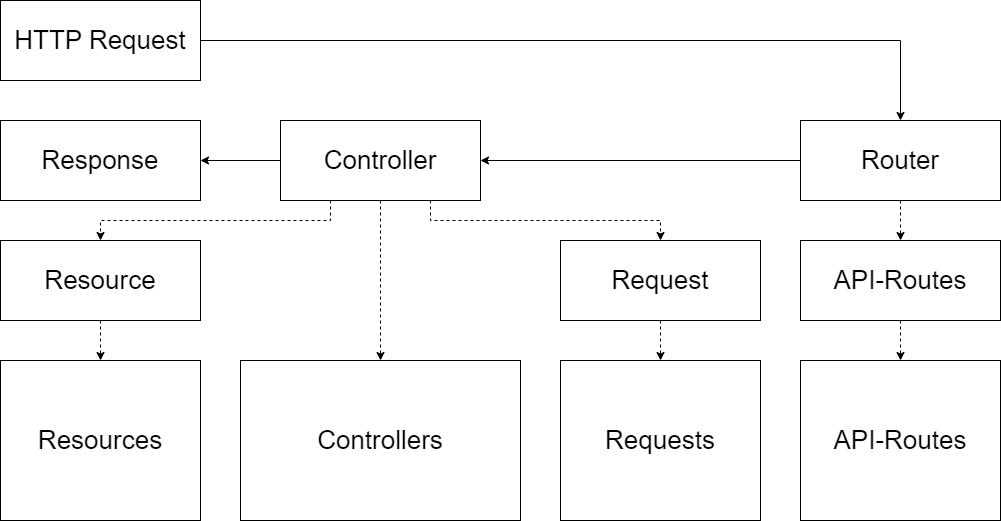
Automatisch generierte Beschreibung

### Klassendiagramm

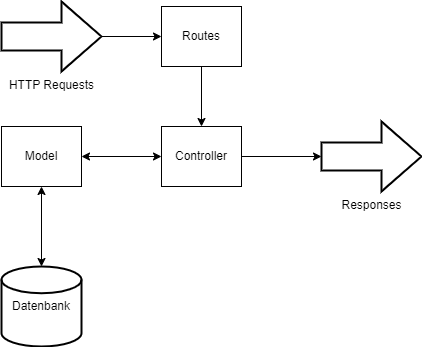
Das Klassendiagramm für Datenbank. Die Tabellen für Migrationen, Tokens und Jobs sind hier nicht abgebildet, um die Übersichtlichkeit zu erhöhen und weil sie hier nicht sehr relevant sind.



### Konzeptioneller Aufbau



Hier abgebildet ist der Vereinfachte Lifecycle eines Requests. Die Punkte werden nach diesem Konzept in einer nach dem anderen durchlaufen.

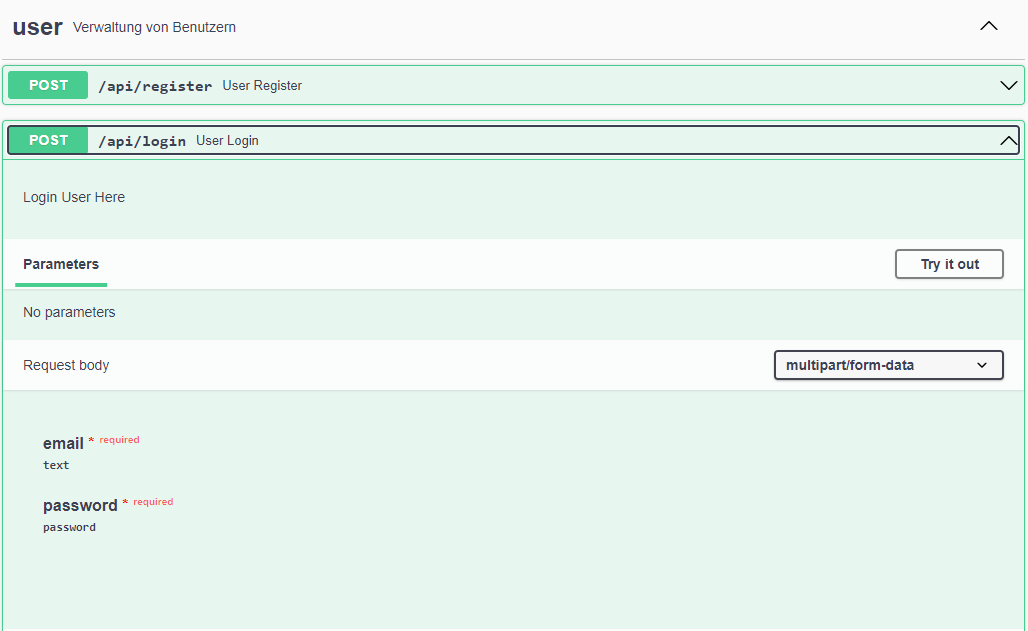


Laravel ist ein Objektorientiertes Framework. Es benuzt Model-Klassen um auf die Datenbank zuzugreifen. Modele können auch Atribute haben, welche Beziehungen zwischen Modelen representieren können.

## Kernfeature Swagger GUI

Swagger ist ein Format zur Beschreibung einer REST-API. Es ist Open Source und benuzt OpenAPI specification. Swagger kann für verschiedene Programmiersprachen und Frameworks verwendet werden. Da diese Anwendung auf dem Laravel-Faramework aufgebaut ist, wurde das eine Version von Swagger spezifisch für Laravel verwendet. Das GitHub Repository von DarkaOnLine, mit dem Namen L5 Swagger[[10]](#footnote-10). 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.

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.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

### 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.

Ein Bild, das Diagramm enthält.

Automatisch generierte Beschreibung

## Geplantes Vorgehen für die Qualitätssicherung

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

Weiter wird durch das händisch erstellte Test konzept 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 besonders eine gute Entscheidung eine Pufferzeit einzuplanen.

## Begründung für Abweichungen zum Zeitplan

Ich habe aber den Dokumentationsaufwand in der Planungsphase ein wenig unterschätzt, wodurch ein kleiner 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.

# Entscheiden

## Variante Bilder als Blob speichern

Eine Bilddatei kann vollständig als sein sogenanter Blob in der Datenbank gespeichert werden. Der Vorteil dieser Methode ist, dass es nicht zwie unterschiedliche Reverenzen 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 seperaten Datei übereinstimmt. Die Datenbank entspricht dem ACID Standard. Backups und Transfers der Datenbank sind somit problemlos.

## 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-Syle | 50% | 4 | 200 | 10 | 500 |
| Komplikation | 30% | 6 | 180 | 10 | 300 |
| Integrität und Datensicherheit | 10% | 10 | 100 | 4 | 40 |
|  |  |  |  |  |  |
| Total |  |  | 500 |  | 920 |

## 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 zurferfügung.

## 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 |

# Realisieren

## Abbildung des Gesamtsystems

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

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 Komunikation 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 komunizieren können. Die Datenbank selbst wird über Migrationen erstellt und mit einem Seeder mit Testdaten populiert.

## 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 Funkionalität erreicht warden 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 Modele 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 mehr 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 (NotFoundHttpException $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 NotFoundHttpException abfange und durch meine eigene Fehlermeldung ersetze. Diese Lösung habe ich auf der Seite Laraveldaily[[11]](#footnote-11) 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 erhallten.

### Funktionsweise des Kernfeatures ABC

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 ABC

Benutzerfreundliche und logisch aufgebauter Zugang zur Verwaltung von Bestellungen, Produkten und Shops gewährleisten.

## Implementierung des Kernfeatures Swagger

Kompromisse:

Die Laravel-Ressource für Routen sieht vor, dass die Update-Methode im Controller mit einem 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 eine weitere Route zu erstellen, die die Update-Methode über eine POST-Route anspricht.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

### Funktionsweise des Kernfeatures ABC

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

### Ziel des Kernfeatures ABC

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

### Effektives GUI / Produkt

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

## Vorbereitungen für den Upload / Live-Schaltung

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

# 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 beinhalltet 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: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 19 - Testprotokoll für User Registrierung

|  |  |
| --- | --- |
| Kurzbeschreibung: | Registrierung als User falsche Eingaben |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 20 - Testprotokoll für User Registrierung Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Login als User falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 21 - Testprotokoll für User Login Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Shops |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 22 - Testprotokoll für Anzeige aller Shops

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Shops falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 23 - Testprotokoll für Anzeige aller Shops Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen eines Shops ist möglich |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 24 - Testprotokoll für Shop Anzeige

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen eines Shops falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 25 - Testprotokoll für Shop Anzeige Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Erstellen eines Shops ist möglich |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 26 - Testprotokoll für Erstellung eines Shops

|  |  |
| --- | --- |
| Kurzbeschreibung: | Erstellen eines Shops falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 27 - Testprotokoll für Erstellung eines Shops Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Bearbeiten eines Shops ist möglich |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 28 - Testprotokoll für Bearbeitung eines Shops

|  |  |
| --- | --- |
| Kurzbeschreibung: | Bearbeiten eines Shops falsche Eingabe |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 29 - Testprotokoll für Bearbeitung eines Shops Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Löschen eines Shops ist möglich |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 30 - Testprotokoll für Löschung eines Shops

|  |  |
| --- | --- |
| Kurzbeschreibung: | Löschen eines Shops falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 31 - Testprotokoll für Löschung eines Shops Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Produkte |
| Voraussetzung: | Ein Produkt existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 32 - Testprotokoll für Anzeige aller Produkt

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Produkte falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 33 - Testprotokoll für Anzeige aller Produkte Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen eines Produktes ist möglich |
| Voraussetzung: | Ein Produkt existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 34 - Testprotokoll für Anzeige eines Produktes

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen eines Produktes falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 35 - Testprotokoll für Anzeige eines Produktes Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Erstellung eines Produktes ist möglich |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 36 - Testprotokoll für Erstellung eines Produktes

|  |  |
| --- | --- |
| Kurzbeschreibung: | Erstellung eines Produktes falsche Eingabe |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 37 - Testprotokoll für Erstellung eines Produktes Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Bearbeitung eines Produktes ist möglich |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 38 - Testprotokoll für Bearbeitung eines Produktes

|  |  |
| --- | --- |
| Kurzbeschreibung: | Bearbeitung eines Produktes falsche Eingabe |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 39 - Testprotokoll für Bearbeitung eines Produktes Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Löschung eines Produktes ist möglich |
| Voraussetzung: | Ein Produkt existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 40 - Testprotokoll für Löschung eines Produktes

|  |  |
| --- | --- |
| Kurzbeschreibung: | Löschung eines Produktes falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 41 - Testprotokoll für Löschung eines Produktes Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Produkte nach Namen |
| Voraussetzung: | Ein Produkt existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 42 - Testprotokoll für Anzeige aller Produkte nach Namen

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Produkte nach Namen falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 43 - Testprotokoll für Anzeige aller Produkte nach Namen Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Produkte sortiert nach Shop |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 44 - Testprotokoll für Anzeige aller Produkte sortiert nach Shop

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Produkte sortiert nach Shop falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 45 - Testprotokoll für Anzeige aller Produkte sortiert nach Shop Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Produkte gefiltert nach Shop |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 46 - Testprotokoll für Anzeige aller Produkte gefiltert nach Shop

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Produkte gefiltert nach Shop falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 47 - Testprotokoll für Anzeige aller Produkte gefiltert nach Shop Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Bestellung ist möglich |
| Voraussetzung: | Eine Bestellung existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 48 - Testprotokoll für Anzeige aller Bestellungen

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Bestellung falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 49 - Testprotokoll für Anzeige aller Bestellungen Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen einer Bestellung ist möglich |
| Voraussetzung: | Eine Bestellung existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 50 - Testprotokoll für Anzeige einer Bestellung

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen einer Bestellung falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 51 - Testprotokoll für Anzeige einer Bestellung Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Erstellen einer Bestellung ist möglich |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 52 - Testprotokoll für Erstellung einer Bestellung

|  |  |
| --- | --- |
| Kurzbeschreibung: | Erstellen einer Bestellung falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 53 - Testprotokoll für Erstellung einer Bestellung Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Bearbeiten einer Bestellung ist möglich |
| Voraussetzung: | Eine Bestellung existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 54 - Testprotokoll für Bearbeitung einer Bestellung

|  |  |
| --- | --- |
| Kurzbeschreibung: | Bearbeiten einer Bestellung falsche Eingabe |
| Voraussetzung: | Eine Bestellung existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 55 - Testprotokoll für Bearbeitung einer Bestellung Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Löschen einer Bestellung ist möglich |
| Voraussetzung: | Eine Bestellung existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 56 - Testprotokoll für Löschung einer Bestellung

|  |  |
| --- | --- |
| Kurzbeschreibung: | Löschen einer Bestellung falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 57 - Testprotokoll für Löschung einer Bestellung Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Bestellung sortiert nach Datum |
| Voraussetzung: | Eine Bestellung existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 58 - Testprotokoll für Anzeige aller Bestellungen sortiert nach Datum

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Bestellung sortiert nach Datum falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 59 - Testprotokoll für Anzeige aller Bestellungen sortiert nach Datum Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Bestellung sortiert nach Status |
| Voraussetzung: | Eine Bestellung existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 60 - Testprotokoll für Anzeige aller Bestellungen sortiert nach Status

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Bestellung sortiert nach Status falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 61 - Testprotokoll für Anzeige aller Bestellungen sortiert nach Status Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Bestellung gefiltert nach Status |
| Voraussetzung: | Eine Bestellung existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 62 - Testprotokoll für Anzeige aller Bestellungen gefiltert nach Status

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Bestellung gefiltert nach Status falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 63 - Testprotokoll für Anzeige aller Bestellungen gefiltert nach Status Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Bestellungsprodukte |
| Voraussetzung: | Ein Produkt existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 64 - Testprotokoll für Anzeige aller Bestellungsprodukte

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen aller Bestellungsprodukte falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 65 - Testprotokoll für Anzeige aller Bestellungsprodukte Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen eines Bestellungsproduktes ist möglich |
| Voraussetzung: | Ein Produkt existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 66 - Testprotokoll für Anzeige eines Bestellungsproduktes

|  |  |
| --- | --- |
| Kurzbeschreibung: | Anzeigen eines Bestellungsproduktes falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 67 - Testprotokoll für Anzeige eines Bestellungsproduktes Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Erstellung eines Bestellungsproduktes ist möglich |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 68 - Testprotokoll für Erstellung eines Bestellungsproduktes

|  |  |
| --- | --- |
| Kurzbeschreibung: | Erstellung eines Bestellungsproduktes falsche Eingabe |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 69 - Testprotokoll für Erstellung eines Bestellungsproduktes Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Bearbeitung eines Bestellungsproduktes ist möglich |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 70 - Testprotokoll für Bearbeitung eines Bestellungsproduktes

|  |  |
| --- | --- |
| Kurzbeschreibung: | Bearbeitung eines Bestellungsproduktes falsche Eingabe |
| Voraussetzung: | Ein Shop existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 71 - Testprotokoll für Bearbeitung eines Bestellungsproduktes Fehler

|  |  |
| --- | --- |
| Kurzbeschreibung: | Löschung eines Bestellungsproduktes ist möglich |
| Voraussetzung: | Ein Produkt existiert in der Datenbank |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 72 - Testprotokoll für Löschung eines Bestellungsproduktes

|  |  |
| --- | --- |
| Kurzbeschreibung: | Löschung eines Bestellungsproduktes falsche Eingabe |
| Voraussetzung: | - |
| Eingabe: | |
| Ausgabe: | |
| Datum: |  |
| Resultat: |  |

Tabelle 73 - Testprotokoll für Löschung eines Bestellungsproduktes Fehler

## Bugfixing

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

## Testprotokoll nach dem Bugfixing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| # | Testfall | Vorbedingung | Vorgehen, Eingaben | Erwartetes Resultat | Effektives Resultat | i.O. |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Tabelle 21 - Testprotokoll nach Bugfixing

**Upload / Live-Schaltung des Produktes**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

## Einführung beim Kunden

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

# Auswerten

## Reflexion der Vorgehensweise

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

## Bewertung des Produktes

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

## Persönliches Schlusswort und Bilanz

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

# Glossarino

C

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

# Glossar

A

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

B

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

C

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

D

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

E

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

F

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

G

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

H

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

I

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

J

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

K

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

L

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

M

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

N

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

O

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

P

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

Q

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

R

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

S

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

T

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

U

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

V

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

W

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

X

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

Y

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

Z

|  |  |
| --- | --- |
| Begriff / Abkürzung | Erklärung |
|  |  |

# Quellenverzeichnis

Literaturverzeichnis

Literatur, die man für die eigene Arbeit benützt hat, muss an **2 Orten** verzeichnet werden:

**1. im Quellenverzeichnis am Schluss der Arbeit,**

**2. im laufenden Text, wo sie wörtlich oder sinngemäss zitiert wird.**

**Eintrag im Quellenverzeichnis**

Jede Arbeit weist am Ende des Textes ein Literaturverzeichnis auf, in dem **alle zitierte Literatur zum Überblick** gesammelt wird. Das Literaturverzeichnis am Schluss der Arbeit ist alphabetisch nach Nachnamen der Autor/innen bzw. Herausgeber/innen geordnet.

Neben dem Autorennnamen müssen mindestens enthalten sein: Der vollständige Titel und Untertitel, der Erscheinungsort und das Erscheinungsjahr.

In das Quellenverzeichnis gehören auch Name und Adresse der Institution, in der die interviewten Personen arbeiten.

**Beispiel für einen Eintrag eines verwendeten Buchs im Quellenverzeichnis**

Grossenbacher, René: Die Medienmacher. Eine empirische Untersuchung zur Beziehung zwischen

Public Relations und Medien in der Schweiz. Solothurn 1986

Internetquellen

Auch Zitate aus dem Internet müssen an **2 Orten** verzeichnet werden: sowohl im Literaturverzeichnis am

Schluss der Arbeit als auch im laufenden Text.

1. Vollständiger Name des Verfassers des Internet-Dokuments; *falls dieser fehlt:*

Titelzeile der Hypertextseite oder Name der Organisation/Institution, die für die Website verantwortlich ist.

2. Titel, gegebenenfalls Untertitel des Internet-Dokuments; falls diese Angaben fehlen:

Beschreibung der Website („Persönliche Homepage“, „Website der Firma“ usw.).

3. Ev. Verweis auf die Online-Zeitschrift, in der das Dokument publiziert wurde, oder weitere Angaben zur Website.

4. Erstellungsdatum oder Datum der letzten Änderung; falls diese Angaben fehlen: „o. J.“.

5. Vollständige URL (Internetadresse).

**Beispiele:**

• Beck, Daniel: Sportübertragung am Fernsehen. Vom Publikumsmagneten zum teuren Verlustgeschäft?

In: Medienheft, 6.9.2016. URL: <http://www.medienheft.ch/kritik/bibliothek/k16_BeckDaniel.html> (22.10.2014).

• Payer, Margarete: Wir katalogisieren das Internet. URL’s, URN’s und Co. Vortrag auf der 1. InetBib-Tagung, Dortmund. 21.4.2014. URL: http://users.aol.com/margpayer/urlso.html (7.11.2016).

• Pellegrino, Joseph: Persönliche Homepage. 24.9.2016. URL:

http://www.english.eku.edu/pellegri/personal.htm (7.11.2016). • SRG SSR Idée Suisse: Die Trägerschaft. Ein Verband von Vereinen. O. J. URL: http://www.srgssrideesuisse.ch/de/corporation/welcom.html (20.3.2016).

Abbildungsverzeichnis

**No table of figures entries found.**

Tabellenverzeichnis

[Tabelle 1 - Adressinformation Auszubildender 6](#_Toc529451342)

[Tabelle 2 – Ausführungszeitraum 6](#_Toc529451343)

[Tabelle 3 – Termine 6](#_Toc529451344)

[Tabelle 4 - Involvierte Personen 7](#_Toc529451345)

[Tabelle 5 - Arbeitsprotokoll Tag 1 9](#_Toc529451346)

[Tabelle 5 - Arbeitsprotokoll Tag 2 10](#_Toc529451347)

[Tabelle 5 - Arbeitsprotokoll Tag 3 11](#_Toc529451348)

[Tabelle 5 - Arbeitsprotokoll Tag 4 12](#_Toc529451349)

[Tabelle 5 - Arbeitsprotokoll Tag 5 13](#_Toc529451350)

[Tabelle 5 - Arbeitsprotokoll Tag 6 14](#_Toc529451351)

[Tabelle 5 - Arbeitsprotokoll Tag 7 15](#_Toc529451352)

[Tabelle 5 - Arbeitsprotokoll Tag 8 16](#_Toc529451353)

[Tabelle 5 - Arbeitsprotokoll Tag 9 17](#_Toc529451354)

[Tabelle 5 - Arbeitsprotokoll Tag 10 18](#_Toc529451355)

[Tabelle 5 - Arbeitsprotokoll Tag 11 19](#_Toc529451356)

[Tabelle 5 - Arbeitsprotokoll Tag 12 20](#_Toc529451357)

# Anhang

[Projektjournal 35](#_Toc529454774)

[Gesprächsprotokoll vom xx. Monat yyyy 35](#_Toc529454775)

[Erster Expertenbesuch 36](#_Toc529454776)

[Zweiter Expertenbesuch 36](#_Toc529454777)

[Code 37](#_Toc529454778)

[Filename.endung 37](#_Toc529454779)

[Filename.endung 37](#_Toc529454780)

[Zusätzliche Manuals, Skripts und weiteres 38](#_Toc529454781)

[Handbuch A 38](#_Toc529454782)

[Skript B 38](#_Toc529454783)

# Projektjournal

Gesprächsprotokoll vom xx. Monat yyyy

Teilnehmer: <Vorname Name>, <Vorname Name>

Datum, Zeit, Ort: <Date>, <Time>, <Place>

**Besprechungsnotizen**:

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

**Getroffene Entscheidungen:**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

Erster Expertenbesuch

Teilnehmer: <Vorname Name>, <Vorname Name>

Datum, Zeit, Ort: <Date>, <Time>, <Place>

**Besprechungsnotizen**:

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

**Getroffene Entscheidungen:**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

Zweiter Expertenbesuch

Teilnehmer: <Vorname Name>, <Vorname Name>

Datum, Zeit, Ort: <Date>, <Time>, <Place>

**Besprechungsnotizen**:

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

**Getroffene Entscheidungen:**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium.

Code

Filename.endung

<Code>

Filename.endung

<Code>

Zusätzliche Manuals, Skripts und weiteres

Handbuch A

<Code>

Skript B

<Code>

1. Aufgabenstellung Original gemäss Eingabe aus PkOrg [↑](#footnote-ref-1)
2. Aufgabenstellung Original gemäss Eingabe aus PkOrg [↑](#footnote-ref-2)
3. Aufgabenstellung Original gemäss Eingabe aus PkOrg [↑](#footnote-ref-3)
4. Aufgabenstellung Original gemäss Eingabe aus PkOrg [↑](#footnote-ref-4)
5. Aufgabenstellung Original gemäss Eingabe aus PkOrg [↑](#footnote-ref-5)
6. Aufgabenstellung Original gemäss Eingabe aus PkOrg [↑](#footnote-ref-6)
7. Aufgabenstellung Original gemäss Eingabe aus PkOrg [↑](#footnote-ref-7)
8. Aufgabenstellung Original gemäss Eingabe aus PkOrg [↑](#footnote-ref-8)
9. Aufgabenstellung Original gemäss Eingabe aus PkOrg [↑](#footnote-ref-9)
10. (DarkaOnLine, 2023) [↑](#footnote-ref-10)
11. (Laraveldaily, 2023) [↑](#footnote-ref-11)