Patrick Tomasi

Jakov Ratkovic

Lauren Arekhi

Analyse E-Shop Waku



**Inhaltsverzeichnis**

[**1** **Einleitung** 2](#_Toc27320559)

[**1.1** **Ressourcen** 2](#_Toc27320560)

[**1.2** **Initialisierung** 3](#_Toc27320561)

[**2** **IST-Zustand** 3](#_Toc27320562)

[**2.1** **Analyse der funktionalen Aspekte** 3](#_Toc27320563)

[**2.1.1** **Starseite** 3](#_Toc27320564)

[**2.1.2** **Angebote** 3](#_Toc27320565)

[**2.1.3** **Login** 3](#_Toc27320566)

[**2.1.4** **Registrieren** 3](#_Toc27320567)

[**2.1.5** **Suchen** 3](#_Toc27320568)

[**2.2** **Analyse nicht funktionale Aspekte** 4](#_Toc27320569)

[**2.3** **Layout** 4](#_Toc27320570)

[**2.4** **UML Zustandsdiagramm** 5](#_Toc27320571)

[**2.5** **UML Anwendungsdiagramm** 5](#_Toc27320572)

[**2.6** **HW Architektur** 6](#_Toc27320573)

[**2.7** **Datenbank** 7](#_Toc27320574)

[**3** **SOLL-Zustand** 8](#_Toc27320575)

[**3.1** **UML-Zustandsdiagramm** 9](#_Toc27320576)

[**3.2** **UML-Anwendungsdiagramm** 10](#_Toc27320577)

[**3.3** **Verbesserungsvorschläge** 10](#_Toc27320578)

[**4** **Testfälle** 11](#_Toc27320579)

[**4.1** **Login** 12](#_Toc27320580)

[**4.1.1** **Beschreibung:** 12](#_Toc27320581)

[**4.1.2** **Tests:** 12](#_Toc27320582)

[**4.1.3** **Erwartungen:** 12](#_Toc27320583)

[**4.2** **Registrierung** 12](#_Toc27320584)

[**4.2.1** **Beschreibung:** 12](#_Toc27320585)

[**4.2.2** **Tests:** 12](#_Toc27320586)

[**4.2.3** **Erwartungen:** 12](#_Toc27320587)

[**4.3** **Warenkorb** 13](#_Toc27320588)

[**4.3.1** **Beschreibung** 13](#_Toc27320589)

[**4.3.2** **Tests** 13](#_Toc27320590)

[**4.3.3** **Erwartungen** 13](#_Toc27320591)

# **Einleitung**

Dieses Dokument gilt der Analyse des E-Shops Waku.

Das vorliegende Dokument enthält die verwendeten Ressourcen, Analysierten Techniken, IST und SOLL Zustand des Produktes, Verbesserungsvorschläge so wie Vorgesehene Testfälle.

Der Kunde hat uns eine ZIP Datei abgegeben. Und den Auftrag erteilt die Webseite in Betrieb zu nehmen und Analysieren.

## **Ressourcen**

Verwendete Ressourcen, um die Webseite und die Datenbank in Betrieb nehmen zu können:

**Webserver:**

* Apache Webserver 2.4.41

**Programmiersprachen:**

* PHP-Version: 7.3.9
* HTML 5
* CSS 3

**Datenbanktyp:**

* MariaDB Version 10.4.6

**Versionierung:**

* GitHub

## **Initialisierung**

Die Datenbank wurde mit phpMyAdmin und dem vom Kunden erhaltenem SQL-Script initialisiert.

Der Datenbank Name musste erstellt werden und danach konnte das Script ohne Probleme initialisiert werden.

Um die Webseite zu starten und um mit der Datenbank zu kommunizieren wurde XAMPP in der Version 3.2.4 verwendet.

Der Code für die Webseite ist in PHP. Dazu mussten alle Dateien in den htdocs Ordner von XAMPP kopiert werden. Damit die Webseite aufrufbar ist und mit der Datenbank kommunizieren kann muss in XAMPP Apache und MySQL laufen.

# **IST-Zustand**

Die in der IST-Zustand ermittelten Daten werden einer detaillierten Analyse unterzogen. Sie dient der Ermittlung des aktuellen Zustands und dem Erkennen von Mängeln und Schwachstellen in der Applikation.

## **Analyse der funktionalen Aspekte**

Hier werden die einzelnen funktionalen Aspekte analysiert und dokumentiert.

### **Starseite**

Mit einem Klick auf den Button Startseite gelangt der User zur Startseite

### **Angebote**

Über den Button Angebote werden dem User Links zu den Kategorien angezeigt, die er anklicken kann, aber es wird nichts angezeigt. Die Daten werden nicht aus der Datenbank gelesen oder dem User angezeigt.

### **Login**

Es ist ein Login Formular enthalten mit dem man den Benutzername wie das Passwort eingeben kann und Absenden. Der Login funktioniert nicht und man kann sich nicht als User anmelden.

### **Registrieren**

In der Navigation ist auch ein Link Registrieren vorhanden. Mit Klick auf den Link funktioniert noch nichts. Man kann sich nicht als neuer User Registrieren.

### **Suchen**

Im Header der Webseite unter dem Titelbild ist ein Suchfeld eingefügt mit der der User auf der Webseite etwas suchen kann. Die Suchfunktion bringt momentan keine Resultate.

## **Analyse nicht funktionale Aspekte**

In der Datenbank ist eine Tabelle Kategorien vorhanden aus der die Kategorien ausgelesen werden und dann auf der Webseite angezeigt werden.

Es sind noch weitere Tabellen vorhanden für den Warenkorb, Bestellungen, Kunden, Produkte, admins die keinen Einfluss in die Webseite haben.

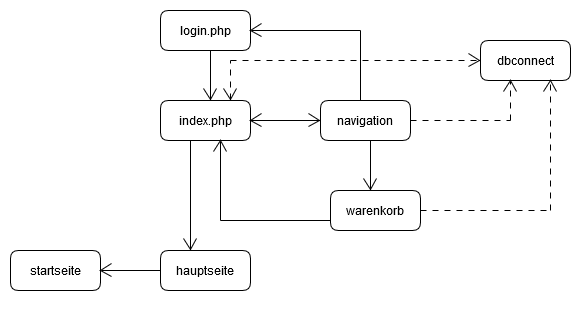
## **Layout**

Mit einem Mockup wird kurz aufgezeichnet wie der momentane Stand der Webseite ist und was schon vorhanden ist. Da die ganzen Untermenus wie Login, Registrieren anzeigen von Angeboten momentan keine Resultate bringt ist das Mockup nur aus einer Seite.

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

## **UML Zustandsdiagramm**

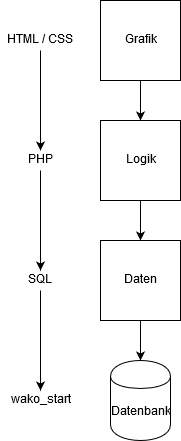


## **UML Anwendungsdiagramm**

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

## **HW Architektur**



HW-Architektur Auf der obersten Schicht ist die Grafische Nutzeroberfläche. Ausgegeben wird alles von dem Web-Server. Mit HTML und CSS wird diese oberste Schicht geschrieben. Auf der logischen Schicht wird PHP verwendet. Damit wird die Verbindung zwischen der Nutzeroberfläche und der Datenbank gemacht. Mit PHP wird auch eine Session gestartet und auch beendet. Auf der untersten Schicht wird mit SQL kommuniziert. Hier werden Abfragen direkt in die Datenbank geschickt. Diese Daten werden über die zweite Schicht mit PHP zur obersten Schicht gesendet wo es dann mit HTML und CSS, für den Nutzer lesbar ausgeschrieben wird.

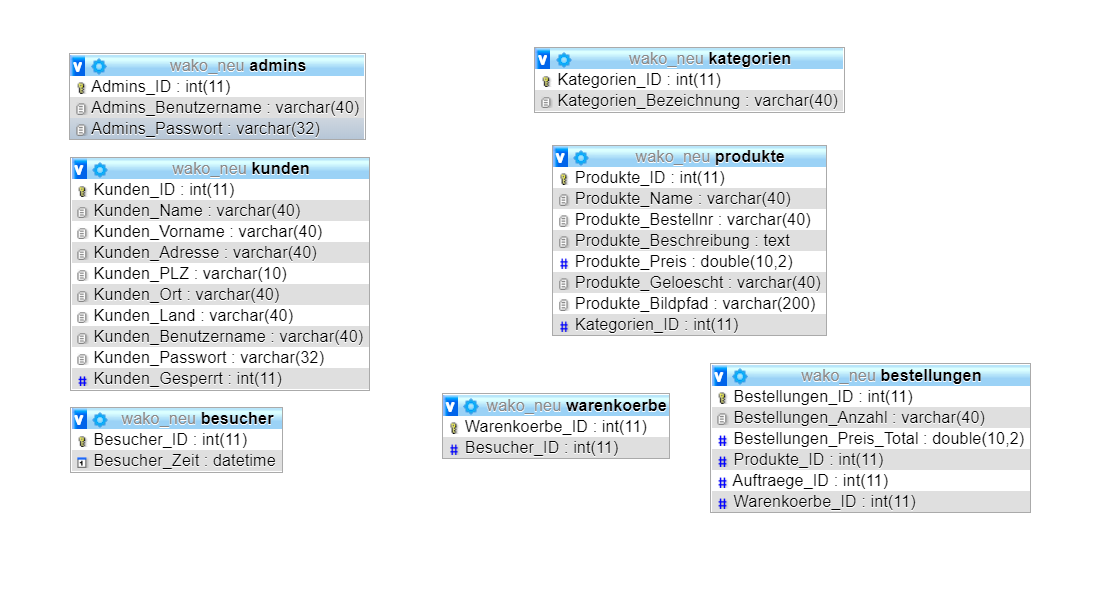
## **Datenbank**

Der ist Zustand der Datenbank ist wie folgt:

Es sind 7 Tabellen vorhanden die schon teilweise mit Daten befüllt sind.

Die Datenbanken haben momentan noch keine Relation zu einander.

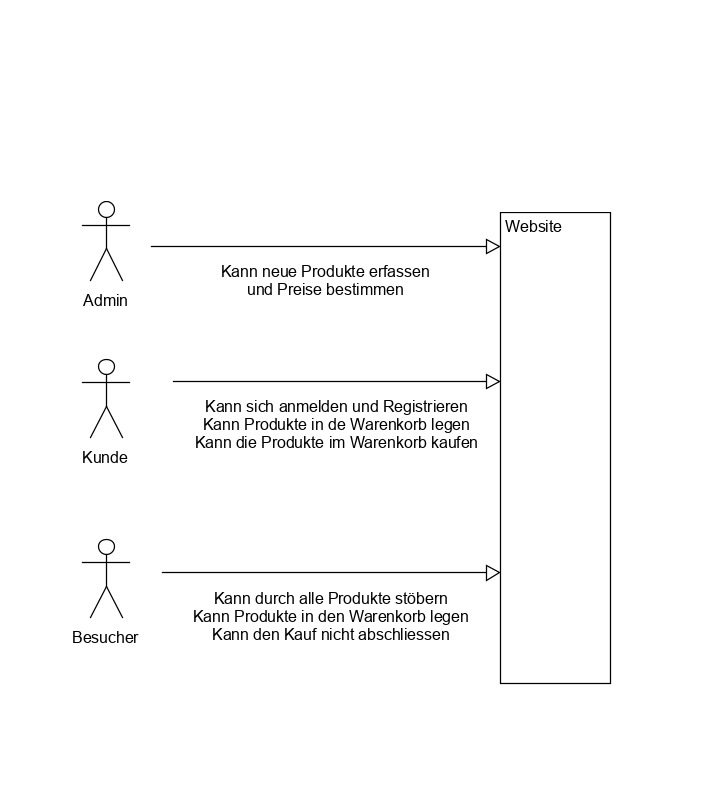
**Die Datenbank sieht wie folgt aus**.

****

In der Applikation sind schon einige SQL-Abfragen eingebaut, diese sind mit der Technologie MySQLI implementiert worden. Die Abfragen werden nicht prepared sprich eine SQL-Injection wäre hier problemlos möglich.

# **SOLL-Zustand**

Der SOLL Zustand soll aufzeigen wie die Applikation auszusehen hat. Um den SOLL Zustand grafisch auch darzustellen wurde wie in der IST Analyse eines Zustands und Anwendungsdiagramm erstellt.

****

## **UML-Zustandsdiagramm**

## **UML-Anwendungsdiagramm**

Ein Bild, das Text, Karte enthält.

Automatisch generierte Beschreibung

## **Verbesserungsvorschläge**

In den Verbesserungsvorschlägen möchten wir dem Kunden aufzeigen was verbessert werden kann oder auch muss um die Applikation Ordnungsgemäss zu Implementieren.

**Login:**

* Da das Login momentan nicht geht sollte die Funktion hinter dem Login implementiert werden.
* Die Datenbank muss mit den Entsprechenden Felder ergänz und angepasst werden.

**Registrierungsformular:**

* Da sich ein neuer Kunde auf der Webseite nicht registrieren kann müsste ein Registrierungsformular implementiert werden.
* Die Logik zur Registrierung.

**Produkte Anzeigen:**

* Um den Kunden eine Auswahl der vorhandenen Produkte anzeigen zu können sollte das Anzeigen der Produckte implementiert werden.

**Suche:**

* Der Suchbutton müsste angepasst werden damit der Kunde auch nach gewünschten Produkten auf der Webseite suchen kann.

**Warenkorb:**

* Um eine Bestellung auch auf der Webseite zu tätigen müsste einen Warenkorb implementiert werden.
* Die ganze Logik hinter dem Warenkorb.
* Datenbank Anpassungen.

**Datenbank:**

* Die Datenbank müsste überarbeitet werden. Einzelne Tabellen entfernt oder Spalten hinzugefügt werden.
* Die Datenbank braucht eine Relation.

# **Testfälle**

Wir definieren hier 3 Tests, die wir beim Entwickeln oder Erweitern der Applikation durchführen werden. Tests während der Entwicklung dienen dazu, die Applikation möglichst fehlerfrei in Betrieb zu nehmen.

## **Login**

### **Beschreibung:**

Um sicher zu stellen das die implementierte Login Logik richtig funktioniert werden wir das Login Verfahren testen.

### **Tests:**

* Password wird verschlüsselt
* Die Abfrage mit der Datenbank funktioniert
* Validierung

### **Erwartungen:**

Es wird erwartet das das Login zu 100% funktioniert. Das das eingetragene Password mit SHA256 verschlüsselt wird bevor es zur Datenbank gesendet wird zur Überprüfung. Die Eingaben sollen validiert werden. Der Registrierte User kann sich mit dem richtigen Benutzernamen und Passwort einloggen und eindeutig identifiziert werden.

## **Registrierung**

### **Beschreibung:**

Da ein komplett neues Registrierungsformular erstellt werden muss und auch die Logik dahinter implementiert werden muss möchten wir die Registrierung in die Testfälle integrieren.

### **Tests:**

* Korrektes Formular mit allen nötigen Eingab Felder
* Validierung der Eingabefelder
* Passwort Verschlüsslung
* Korrektes abspeichern der Daten in der Datenbank

### **Erwartungen:**

Es wird erwartet das das Registrierungsformular zu 100% funktioniert. Das die Eingabefelder alle nötigen Informationen für eine korrekte Registrierung enthalten. Das die einzelnen Felder Validiert werden und das das Passwort verschlüsselt wird. Es wird auch erwartet das die eingegebenen Daten korrekt in der Datenbank abgespeichert werden.

## **Warenkorb**

### **Beschreibung**

Der Warenkorb wird komplett neu implementiert. Um sicherzustellen das der Warenkorb richtig funktioniert werden wird den Warenkorb und die Implementierte Logik dahinter testen.

### **Tests**

* Artikel lassen sich in den Warenkorb legen
* Artikel lassen sich bearbeiten oder Entfernen im Warenkorb
* Der Warenkorb enthält alle nötigen Informationen, und speichert den Warenkorb in der Datenbank ab

### **Erwartungen**

Es wird erwartet das der Warenkorb zu 100% funktioniert. Das die gewünschten Artikel in den Warenkorb gelegt werden. Der Warenkorb muss bearbeitet werden können oder auch ein Artikel wieder entfernt werden. Der Warenkorb muss korrekt in die Datenbank gespeichert werden.