Dokumentation

Webshop

Gruppe 13

Phillip Kelsch und

Gabriel Haardt

Projektidee und Projektziele

Bei der Suche nach einer Projektidee hatte für uns die Erfüllung der Anforderungen die höchste Priorität. Mit einem Webshop war es möglich ein “Responsives Frontend” zu realisieren und auch die anderen Anforderungen, wie verschiedene Ansichten, zwei Anwendungsbereiche und die Darstellung/Einbindung von Inhalten aus einer Datenbank haben sich gut darauf anwenden lassen, da ein normaler Webshop diese Funktionalitäten haben sollte. Auch die Bonus-Anforderungen waren damit gut realisierbar.

Bei der Wahl der Programmiersprachen haben wir uns auf einen Stack aus folgenden Programmiersprachen und Bibliotheken geeinigt:

- php -> Einsteigerfreundlich, viele einfache Funktionen, bei komplexeren Anwendungen wird es dann ein wenig komplizierter
- mysql -> Standard-Datenbankservice mit guter Einbindung in php als Programmiersprache
- html -> zur strukturierten Darstellung von Inhalten -> gut kombinierbar mit php - daher anfängerfreundlich
- css -> Stylesheet Sprache, zum Anpassen des Designs -> gut kombinierbar mit dem bisherigen Stack
- Bootstrap -> Erweiterung von css durch Framework, einfachere Darstellung für Produktseite und angepasstes Design
- PHPMailer -> php Bibliothek zum Versandt von E-Mails über SMTP

Unser Ziel war, wie es auch im Mockup in der Präsentation sichtbar ist, einen Webshop mit folgenden Funktionalitäten zu erstellen:

- Navigationsbar horizontal am Seitenbeginn zum Navigieren auf der Website
- Eigener Login-/Registrierungsbereich, der nur sichtbar ist, wenn kein Benutzer angemeldet ist -> Logout nur sichtbar, wenn Benutzer angemeldet ist
- Verschiedene Ansichten für Produkte (Übersicht, eigene Produkte, hinzufügen/bearbeiten/löschen), Kontaktformular und Produkttests (Videos)
- Klare Änderung der Sichtbarkeiten, wenn angemeldet/nicht angemeldet
    - Eigene Produkte und Produkte hinzufügen
- Eine Art Marketplace-System, bei dem Benutzer eigene Produkte hinzufügen können, die dann von anderen gesehen werden können
- E-Mail-Benachrichtigungen bei Registrierung etc. per PHPMailer
- YouTube Videos einbetten auf der Seite Produkttests
- Warenkorbsystem
- Filter-System für die Produktseite

Installation

Zur lokalen Installation der Webanwendung sollte das Programmpaket XAMPP von der Produktwebsite https://www.apachefriends.org/de/download.html heruntergeladen und über die Standardroutine installiert werden.

Zum Clonen/Herunterladen der Websitedaten kann die Git Bash von https://git-scm.com/downloads heruntergeladen und nach Standardroutine installiert werden.

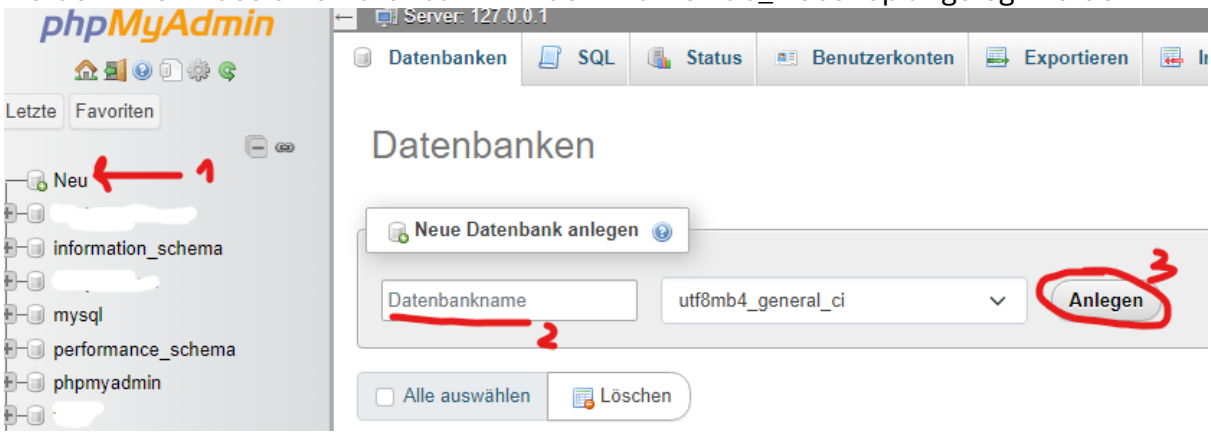
Sobald die Git Bash korrekt eingerichtet und geöffnet ist kann mithilfe von

cd /c/xampp/htdocs/
in das Webverzeichnis des Apache2 Webservers gewechselt werden und dann mit

git clone https://github.com/marvinbalsam/web-technologien-gruppe13.git

vom GitHub-Repository in das lokale Webverzeichnis geklont werden.

Jetzt kann mithilfe des XAMPP Control Panels der Apache und MySQL Dienst gestartet werden und mit einem Klick auf „Admin“ hinter MySQL der phpMyAdmin geöffnet werden. Hier muss eine Datenbank mit dem Namen db_webshop angelegt werden:



In diese Datenbank kann mithilfe der „Importieren“ Funktion das Datenbankdump aus der Datei „db_webshop.sql“ aus dem Verzeichnis, in das das Repository geklont wurde, importiert werden.

Funktionen und Bereiche

Startseite

Die Startseite (index.php) ist der Einstiegspunkt des Webshops und bietet angemeldeten sowie unangemeldeten Usern eine ansprechende Darstellung über die Qualitäten des Webshops und bietet eine Übersicht über die wichtigsten Fakten.

Produkte

Die Produkte werden mit der „products.php“ dargestellt. Zur Erstellung und Darstellung der Produktbilder iteriert eine while-Schleife gegen die Anzahl an Zeilen der Tabelle

„products“. Diese Anzahl wird mit einem „mysqli_result“ erstellt. Dieses „mysqli_result“ wird im Vorhinein mit einer „mysqli_query“ erstellt, die alle Daten aus der Tabelle „products“ holt.

In der while-Schleife wird dann die „card.php“ included, diese ist dafür zuständig die Produktdaten darzustellen. Das heißt nur, dass die „card.php“ an der Stelle aufgerufen wird. Die „card.php“ arbeitet mit dem card-container von Bootstrap, der eine breite Auswahl an Möglichkeiten und Einstellungen bietet. Dieser card-container wird nun mit den Daten aus der Datenbank gefüllt. Ganz oben (card-img-top) wird das Bild eingefügt, darunter (card-title) kommt der Produktname und in den card-body kommt die Beschreibung und der Preis. An die Daten kommen wir, da das „include card.php“ die Variable „$row“ mit an die Klasse „card.php“ übergibt.

Shopping Cart: Zusätzlich wird in der Schleife der Button „In den Warenkorb“ zu jedem Produkt hinzugefügt, dieser übergibt beim Klick die „product_id“ an die Variable „addToCart“. Diese wird anschließend nach der Schleife mit dem „$_GET“-Befehl abgespeichert und das Produkt mit der abgespeicherten „product_id“ und der „user_id“ ($loggedInuser) wird in den Warenkorb gelegt. Hier ist es wichtig die “user_id” einzubinden, da jeder Benutzer seinen eigenen Warenkorb haben soll. Die aktuelle “user_id” erhalten wir mit einer “session-variable”. Beim erfolgreichen Anmelden wird eine “session” gestartet, anschließend wird eine “session-variable” mit der “user_id” gefüllt. Wenn nun Produkte in den Warenkorb gelegt werden, wird oben rechts in der Leiste bei “Warenkorb()” eine Zahl in die Klammern eingetragen, ein Produkt ist also in den Warenkorb gewandert. Auf diesen Warenkorb kann der User auch zugreifen in dem er auf “Warenkorb(x)” klickt, damit gelangt er auf die “cart.php”. Hier werden vorerst nur alle Produkte angezeigt, ergänzt müsste noch werden, welche Anzahl von welchem Produkt in dem Warenkorb ist.

Produkte hinzufügen: In der „products.php“ werden zusätzlich am Anfang der Seite noch 2 bzw. 3 Buttons hinzugefügt, der Erste davon ist für das Produkt hinzuzufügen. Durch Klicken des Buttons wird der User auf die „product_add.php“ weitergeleitet. Auf dieser Seite werden mehrere Eingabefelder angezeigt, in denen der User den Produktnamen, die Beschreibung und den Preis eingeben kann. Als Letztes kann noch ein Bild hochgeladen werden. Wenn auf „Hinzufügen“ geklickt wird, werden die eingegebenen Werte mit der Methode „post“ gespeichert. Um diese dann wieder zu nutzen, werden sie mit „$_POST“ aufgerufen. Die Werte werden dann in lokale Variablen gespeichert und mittels einer "mysqli_query” in die Tabelle „products“ eingefügt. Dadurch entsteht ein neues Produkt. Für die Bilder ist zu beachten, dass sie vorher „base64-encoded“ werden müssen, da die Bilder in der Datenbank als „longblob“ gespeichert werden, und dieses Format eine Kodierung benötigt.

Eigene Produkte anzeigen: Einer der anderen Buttons auf der Produktseite ist dafür da, um auf seine eigenen Produkte zuzugreifen. Dieser leitet den User beim Klicken also auf die „userProducts.php“ weiter. Hier hat der User eine Übersicht über seine eigenen Produkte. Die Funktionalität ist fast gleich wie von “products.php”, mit dem Unterschied, dass die Datenbankabfrage mit der Bedingung “WHERE user_id = $loggedInUser” gemacht wird, damit nur die Produkte angezeigt werden, die auch zu dem angemeldeten Benutzer gehören. Hier kann der User nun auch wieder auf 2 Buttons zugreifen, einmal zum Produkt “Bearbeiten” und zum “Löschen”.

Produkte bearbeiten: Der Button “Bearbeiten” führt den User auf die “product_change.php”, die von der Funktionalität der “product_add.php” ähnelt, aber auch hier können wieder ein paar Unterschiede festgestellt werden. Einmal sind die Felder zum Eingeben der Änderungen schon gefüllt, indem die “label” im “form” mit den Daten des aktuell ausgewählten Produkts gefüllt werden. Hiermit wird vermieden, dass das Produkt bei nicht Ausfüllen eines “labels” in dieser Spalte keine Werte mehr hat. Beim Klick auf “Ändern” werden also die Daten des ausgewählten Produkts “WHERE….” mit einer “mysqli_query” geupdated, das heißt vorhandene Daten werden überschrieben. Nach der Änderung wird der User wieder zurück auf die eigene Produktseite geleitet. Hier befindet sich auch wieder ein Button “Zurück zur Produktübersicht”, mit dem der User einfach wieder zurück zur “products.php” gelangt, ohne eine Änderung zu vollziehen.

Produkte löschen: Der Button “Löschen” in der “userProducts” ruft die “product_delete.php” auf, hier wird mit einer “mysqli_query” wieder das richtige Produkt ausgewählt und einfach gelöscht. Anschließend geht es wieder zurück auf “userProducts”.

Kontakt

Das Kontaktformular wird über die contact.php realisiert. Hier können angemeldete und unangemeldete User in den dafür vorgesehenen drei Feldern ihren Namen, ihre E-Mail-Adresse und eine Nachricht zu ihrem Anliegen eintragen und das Formular mithilfe des „Absenden“ Buttons verschicken. Im Hintergrund werden dann per POST Methode die eingegebenen Daten in Variablen gespeichert. Damit das Kontaktformular auch an den Betreiber der Website übermittelt wird, wird hier statisch die E-Mail-Adresse des Betreibers unter der Variable „mail_recipient“ hinterlegt. Unter der Variable „mail_subject“ wird statisch der Betreff auf „Neue Nachricht ueber Kontaktformular“ festgelegt. Als letzte Variable „mail_message“ wird dynamisch aus den Inhalten des Kontaktformulars ein Text generiert: „Von (eingegebenem Namen) mit E-Mail (eingegebener E-Mail Adresse) kommt Nachricht: (eingegebene Nachricht).

Im Anschluss wird die Funktion zum Mailversandt aufgerufen.

Mailversandt

Damit über den Webshop bei bestimmten Funktionen Mails versendet werden können, wurde die „sendmail.php“ geschrieben. Das Projekt bedient sich zum Versand von Mails über SMTP an der Bibliothek „PHPMailer“. Diese ist über das Klonen des Repositorys im Projekt enthalten und wird über die „sendmail.php“ gesteuert.

Mithilfe der Übergabe von Daten in den Variablen mail_recipient, mail_subject und mail_message können aus allen anderen PHP-Unterseiten Daten für eine Mail übergeben werden und somit redundanter Code gespart werden.

Registrierung

Auf der Seite zur Registrierung (register.php) befindet sich ein Formular mit vier Feldern zur Eintragung von Vornamen, Nachnamen, E-Mail und Passwort. Noch nicht registrierte User können sich damit an der Webanwendung registrieren.

Nachdem die genannten Daten ausgefüllt worden sind und mit dem Button „Registrieren“ abgeschickt wurden, wird die PHP-Seite per GET-Methode erneut aufgerufen über „register.php?register=1“

Wenn dieser GET-Request erkannt wird, werden die Daten verarbeitet und in die Datenbank geschrieben. Außerdem wird über die bereits erklärte sendmail.php eine Bestätigungsnachricht versendet. Auf der Seite erscheint dann der Hinweis: “ erfolgreich! Wir haben Ihnen eine Mail gesendet.“ Jetzt kann sich der neue User über die Loginseite anmelden.

Login

Auf der Loginseite (login.php) befindet sich ein einfaches Formular mit den Feldern „Mail“ und „Passwort“. Der User muss diese Felder ausfüllen und den „Login“ Button klicken. Die Daten werden dann per POST-Methode an die loginsubmit.php übermittelt und weiterverarbeitet.

Zum Abgleich auf die Korrektheit der Daten wird eine SQL Abfrage nach dem Datensatz mit den eingegebenen Daten durchgeführt. Dann wird ausgewertet, ob es ein Ergebnis oder kein Ergebnis gibt. Wenn die angegebenen Daten auch den Daten in der Datenbank entsprechen wird der Login durchgeführt und es wird eine Nachricht „Login erfolgreich. Sie werden auf das Dashboard weitergeleitet“ ausgegeben, sowie der User auf die products.php weitergeleitet.
Falls kein übereinstimmendes Ergebnis mit den eingegebenen Daten gefunden wird, schlägt der Login fehl und der User erhält die Nachricht „Login fehlgeschlagen. Bitte versuchen Sie es erneut.“