Grobe Beschreibung unseres Webshops

Unser Ziel war es, ein Webshop zu programmieren. Der Webshop soll alle Produkte in verschiedenen Kategorien auf der Startseite anhand von Bildern anzeigen. Diese Produkte werden in einer Datenbank gespeichert, in der die zugehörigen Daten, wie Preis, Name und Anzahl aufgelistet sind.

Auf der Startseite kann man auf den Warenkorb zugreifen. Wählt man ein Produkt aus, wird man auf eine Detailansicht des Produktes weitergeleitet. Dort kann man die Menge auswählen und in den Warenkorb legen. Möchte man weiter einkaufen, gibt es einen Return-Button, möchte man allerdings zahlen, so kann man direkt auf die Seite des Warenkorbes. Dort werden alle Produkte angezeigt, die man ausgewählt hat. Diese Produkte sind für den Käufer in einer gesonderten Datenbank gespeichert. Klickt der User nun auf „kaufen“, so werden die Produkte aus der Datenbank entfernt bzw. abgezogen. Nach dem Kauf gelangt der User wieder auf die Startseite des Shops.

Der User muss sich für den Kauf registrieren oder anmelden, sofern er schon ein Konto hat, diese Daten sind ebenfalls in einer Datenbank gespeichert. Bei der Anmeldung mit den richtigen Daten setzen wir eine Session, die gelöscht wird, wenn dieser sich wieder abmeldet.

Im Grunde ist uns dies in den meisten Punkten auch gelungen.

Der Webshop besitzt eine Startseite, welche alle Artikel, die wir im Shop haben, anhand eines Bildes zeigt. Zusätzlich wird Information darüber gegeben, wie das Produkt, in unserem Fall Stoffbären, heißt, wie viel es kostet und wie groß der Bestand ist. Die Informationen zu den Bären haben wir in einer Datenbank „produkte“ gespeichert.

Auf der Startseite wird der Nutzer ebenfalls darüber informiert, dass er sich erst anmelden oder registrieren, sollte er noch kein Konto besitzen, muss, um einen Teddy erwerben zu können. Man kann dann entweder erst sich die Bären angucken und dann sich erst, wenn man sie kaufen möchte einloggen oder registrieren, oder man loggt sich gleich zu Beginn ein. Beim Registrieren werden die Daten: Benutzername, Passwort, Straße, Postleitzahl und Stadt in der Datenbank „data“ gespeichert, um beim Einloggen zu vergleichen, ob der angegebene Benutzername existiert oder nicht und ob das Passwort richtig ist, wenn dies nicht der Fall ist, wird eine Fehlermeldung gegeben und der Nutzer wird gebeten, sich erneut anzumelden. Auch beim Registrieren wird überprüft, ob alle Felder ausgefüllt wurden, ansonsten wird der User darauf hingewiesen, dass bestimmte Felder noch nicht ausgefüllt wurden.

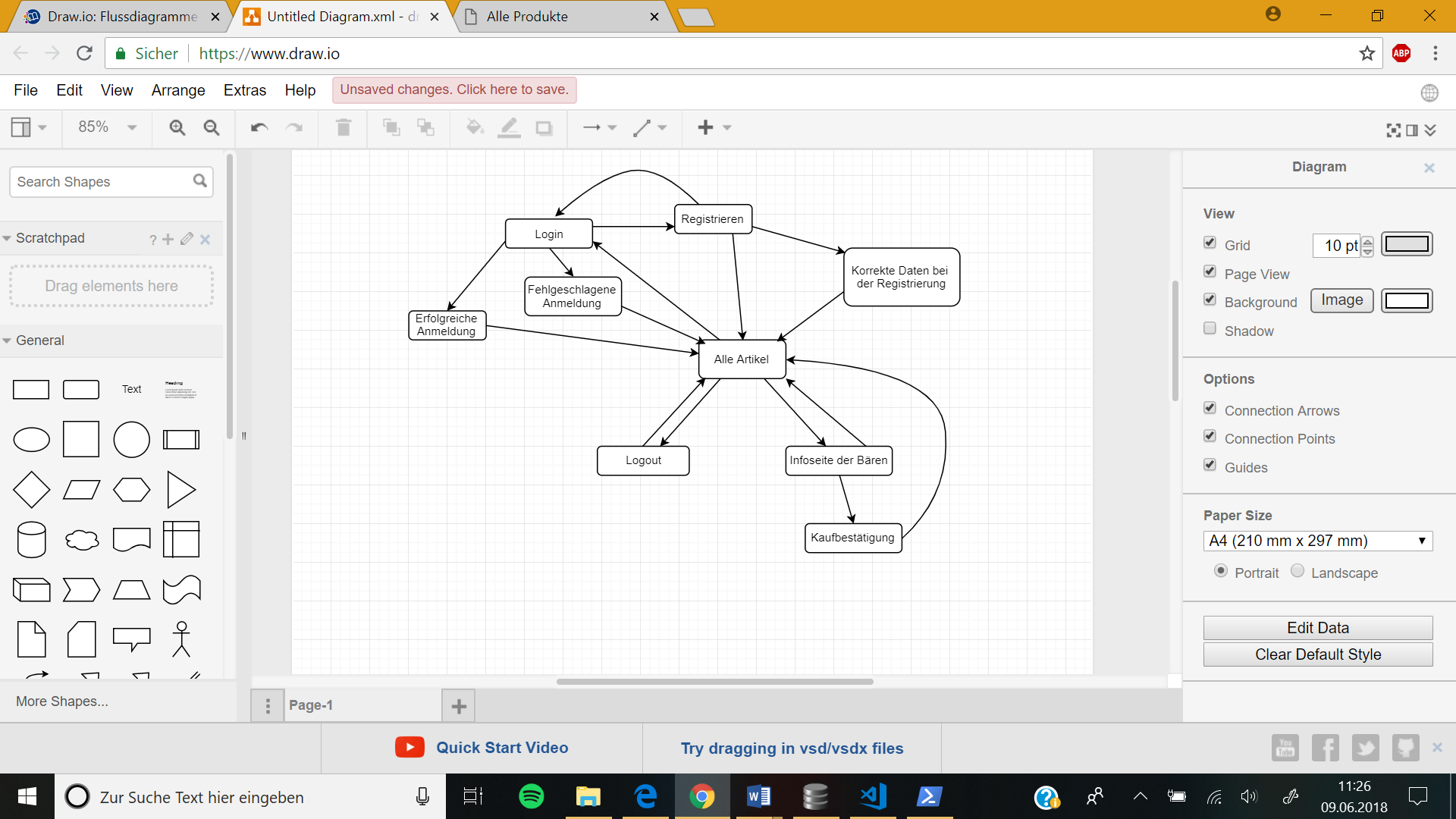
Ist man nun eingeloggt, wird für den Benutzer eine Sessionsvariable gesetzt, dieser kann dann zurück auf die Startseite gehen und sich die Stoffbären ansehen, in dem er auf den Button „Ansehen“ klickt. Dies leitet den Benutzer auf eine weitere Seite, wo nur noch der einzelne Bär angezeigt wird, den man ausgewählt hat. Dies wird dadurch erreicht, dass die ejs Datei beim Anklicken des Buttons die Id des Bären in der Datenbank an den Server übergibt, dieser vergleicht diese Id mit denen in der Datenbank und gibt die Werte aus, die der Id entsprechen. Die Werte werden gespeichert und an die ejs „Info“ weitergegeben, welche die dann rendert und ausgibt. Nun kann man den Bären kaufen, wenn man möchte.

Der Nutzer klickt dafür auf „Kaufen“, dabei wird der Wert der Anzahl des Bären wieder an den Server übergeben, dieser reduziert die Anzahl dann um 1. Der Nutzer wird währenddessen entweder darauf hingewiesen, dass er sich noch anmelden muss oder auf die Seite weitergeleitet, dass der Bär in ein paar Tagen eintreffen wird. Auf dieser Seite wird man dann hingewiesen, dass man noch weitere Bären kaufen kann, wenn man auf den Link für die Startseite klickt. Möchte man sich nun abmelden, klickt man nur auf den zugehörigen Link und bestätigt mit einem „Ja“, dass man dies auch wirklich möchte. Daraufhin wird die Session gelöscht.

Das, was wir nicht erreicht haben, sind die einzelnen Kategorien, jedoch erschien uns das, da wir statt Kleidung uns Accessoires für Stoffbären entschieden haben, nicht mehr so relevant; und der Warenkorb.

Diagramm der Struktur unserer HTML Seiten

Wie die einzelnen Seiten verknüpft sind und von welcher Seite man auf welche kommen kann.



Stundenzettel

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Termine** | **Beschreibung** | **Nhi** | **Kira** | **Sandy** | **Summe** |
| 23. April | * Konzeptformulierung und Mockups | 0,5 |  | 1 | 1,5 |
| 09. Mai | * Einarbeitungen zu JavaScript und Datenbanken (SQlite) * Datenbank für Produkte | 2 |  | 4 | 6 |
| 10. Mai | * Nacharbeitung der Tutorials und Module * Fehlerbehebung und Erstellung Startseite | 2 |  | 3 | 5 |
| 15. Mai | * Überarbeitung der Datenbank |  |  | 2 | 2 |
| 16. Mai | * Nacharbeitung SQlite * CSS | 1 | 3 |  | 4 |
| 18. Mai | * Datenbank fertigstellen |  |  | 3 | 3 |
| 21. Mai | * Benutzerverwaltung * Datenbank für Benutzer | 7 |  |  | 7 |
| 22. Mai | * Benutzerverwaltung fortsetzen | 4 |  |  | 4 |
| 29. Mai | * CSS Einarbeitung und Implementation * Projektbericht |  | 6 | 3 | 9 |
| 03. Juni | * Sessions * CSS Layout * Infoseite | 3 | 4 | 2,5 | 9,5 |
| 04. Juni | * Cookies * Bugfixing und Fehlerbehebungen | 3 |  |  | 3 |
| 05. Juni | * Bugfixing * Kaufenfunktion * Sessions löschen * Feinschliff CSS | 3 | 2 | 2 | 7 |
| 08. Juni | * Zusammenfassung * Projektbericht |  |  | 4 | 4 |
| 09. Juni | * Projektbericht * Diagramm der Struktur * Datei-Upload |  |  | 4 | 4 |
| **Summe** |  | **25,5** | **15** | **28,5** | **69** |