CAMPSCH WEBSHOP

Nico Schenk & Thomas Campbell

Inhaltsverzeichnis

Einführung

Webshop bzw. E – Commerce bezeichnet Einkaufsvorgänge, welche mittels Internet getätigt werden. Werben, Kaufen und Verkaufen ohne das Haus zu verlassen, einfach ganz bequem vom eigenen Rechner aus.

Bei der objektorientierte Programmierung ist die Zielsetzung, dass Quellcode einfacher wiederverwendet werden kann, der Quellcode übersichtlicher wird und zukünftige Erweiterungen einfacher werden. Ein weiterer grosser Vorteil entsteht, dass durch ein Grundverständnis auch fremde objektorientiert programmierte Klassen im eigenen Projekt eingesetzt werden können und dadurch massig Zeit gespart werden kann.

Projektbeschreibung

Wir haben den Auftrag erhalten, einen Webshop mit PHP zu programmieren. Es wurden uns insgesamt 32 Lektionen Zeit zur Verfügung gestellt. Vorausgesetzt wird eine objektorientierte Programmierung mittels Models, Controllers und Views.

Nach einem Brainstorming haben wir uns zum Ziel gesetzt, eine Seite zu programmieren, welche das Login als Startseite hat. Eine Registerfunktion, sowie die Eingabe der kompletten Adresse sollen möglich sein und alles, in zwei verschiedene Textdateien gespeichert werden. Es soll eine Shop Oberfläche verfügbar sein, mit max. 10 Produkten, welche per Klick in den Warenkorb wandern. Ausserdem soll der Inhalt per Mausklick „bestellt“ werden indem er in eine weitere Textdatei gespeichert wird. Ein kleiner Adminbereich, in dem die User und Bestellungen eingesehen werden können, wäre ein toller Abschluss.

Genauer haben wir alles in unseren Anforderungen definiert.

Verwendete Ressourcen

Betriebssystem:	Microsoft Windows 7
Virtuelle Maschine:	Virtualbox
Programmierungumgebung:	PHPStorm – Nico Schenk Notepad++ – Thomas Campbell
Server:	Square 7 (Windows) – Nico Schenk XAMPP (Windows) – Thomas Campbell Apache (Linux)
Hosting:	
Backup:	Dropbox/Modul133
Sharing:	Github nicoschenk/Campsch

Tabelle 1 Verwendete Ressourcen

Anforderungen

Funktionale Anforderungen

- Session Handling
  - *Produkte sollten im Warenkorb an die Session gebunden werden, um bei einer Aktualisierung der Seite weiterhin angezeigt zu werden*
- Login Funktion
  - *Voraussetzung definiert durch Herrn Buchs*
- Authentifizierung
  - *Prüfung von Passwort und Username anhand der Textdatei*
- Registration
  - *Voraussetzung definiert durch Herrn Buchs*
  - *Userdaten in Textdatei schreiben*
- Kundendaten
  - *Separate Erfassung der Kundendaten*
  - *Speicherung in Textdatei*
- Warenkorb
  - *Soll sich dynamisch anpassen*
  - *Warenkorb Inhalt soll angezeigt werden können*
- Admin Funktionen
  - *Kann registrierte User einsehen via Einlesen der Textdatei*
  - *Kann die Kundendaten zu den Usern einsehen*
  - *Kann getätigte Bestellungen einsehen*

Nichtfunktionale Anforderungen

- Alle Verlinkungen laufen über die index.php
  - *In der Index werden alle Funktionen aufgerufen*
  - *Saubere Trennung aller Klassen*
- Der Webshop sollte eine geringe Fehleranfälligkeit haben
  - *Sollte stabil laufen, solange sich die Fehler auf wenige Kleinigkeiten beschränken, ist dies nicht weiter von Relevanz.*
- Der Webshop sollte schnell und einfach in der Benutzung sein
  - *Eine übersichtliche und angenehme Benutzeroberfläche sollte vorhanden sein*
- Die Webseite soll mit PHP, HTML und CSS programmiert sein
  - *Da dies eine Voraussetzung in diesem Projekt ist, ist die Wichtigkeit hoch*
- Die Webseite sollte sich der Browsergrösse anpassen
  - *Hier reicht es aus, die Seite für Mozilla Firefox zu optimieren*
- Die Software sollte gesetzeskonform und ethisch nicht anstossend sein
  - *Das Datenschutzgesetz ist zu berücksichtigen, sowie die Politik der Bevölkerung*

Zeitplan

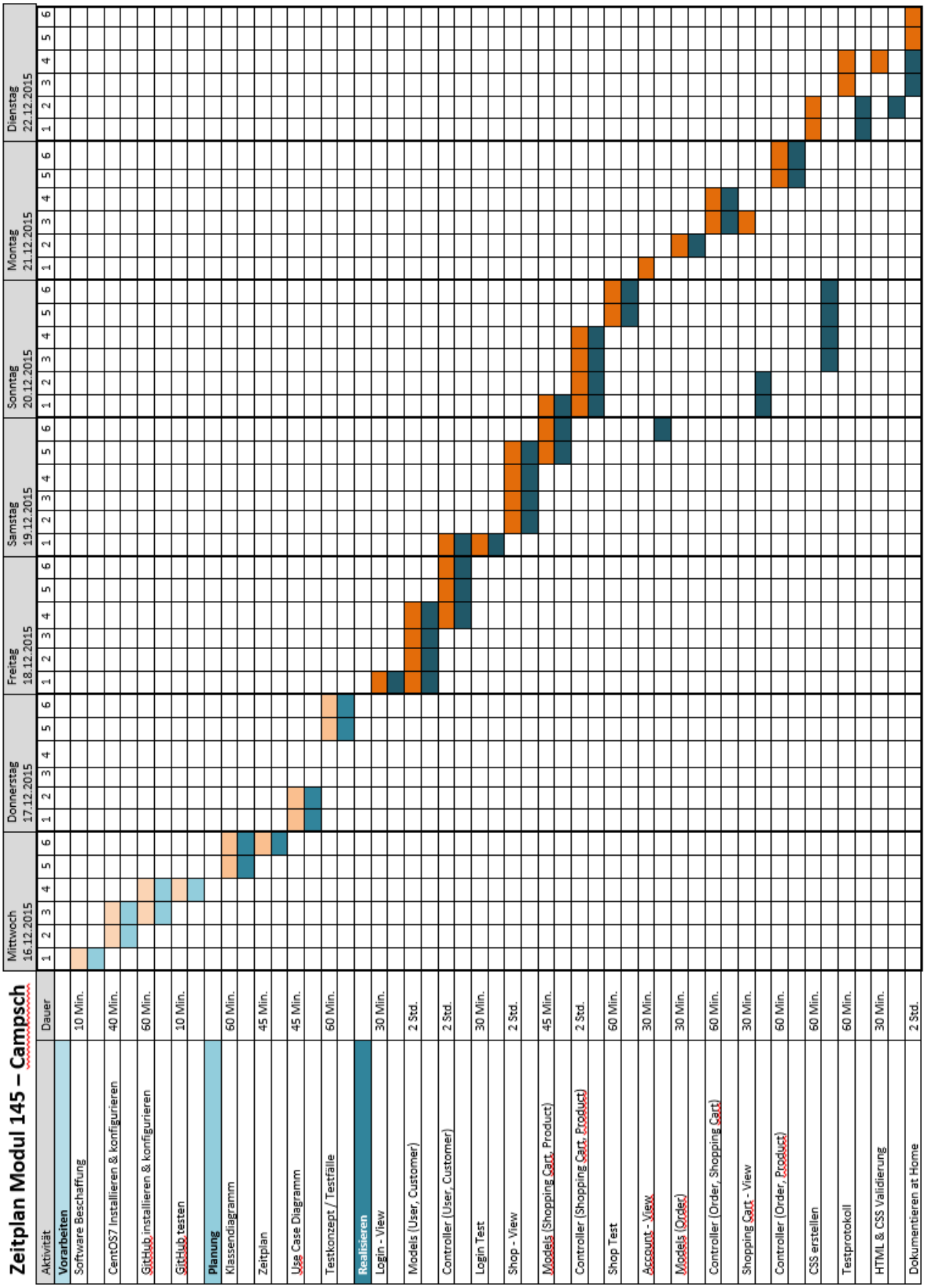

Abbildung I Zeitplan

Use – Case Diagramm

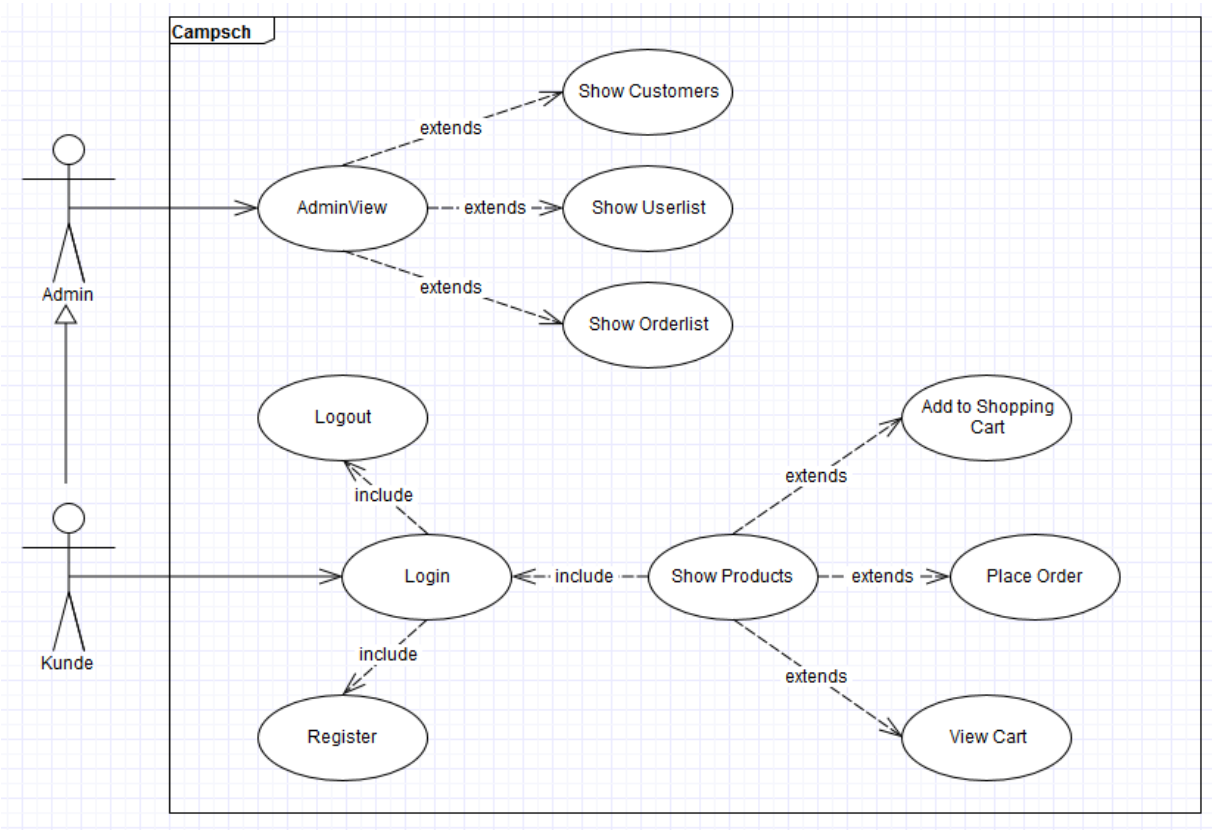

Abbildung II Use - Case Diagramm

Klassendiagramm

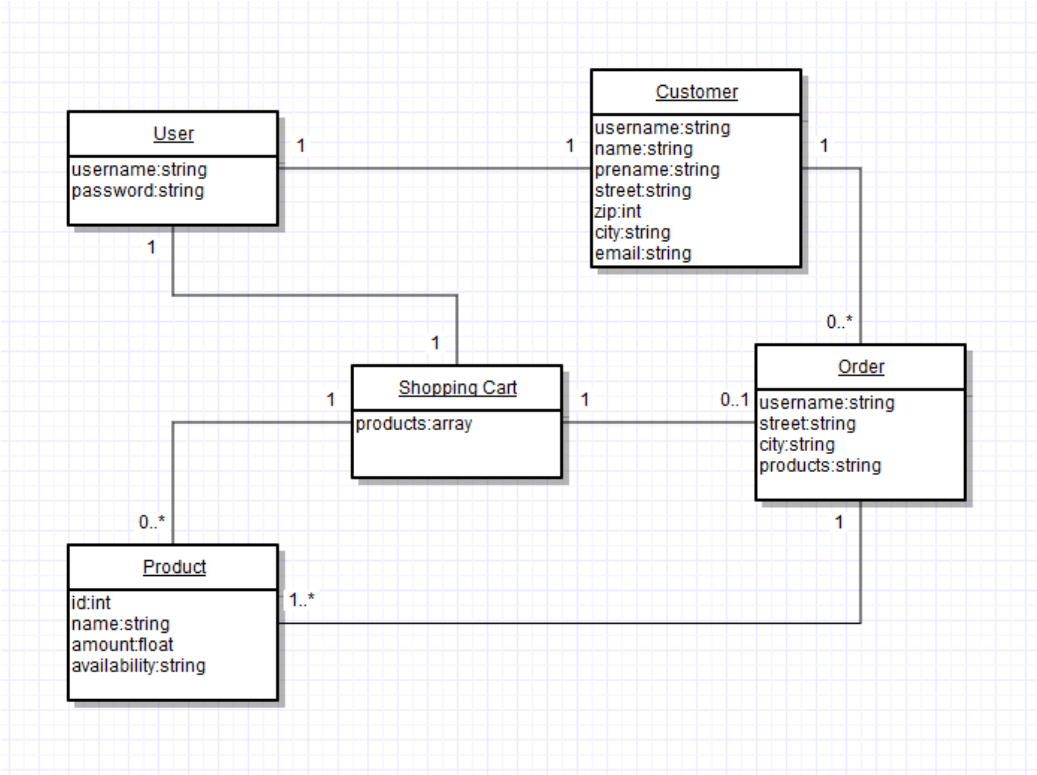

Abbildung III Klassendiagramm

Testfälle

1 Test	**Titel : Webshop**
Datum : 17.12.2015	**Art : Verfügbarkeit**
Autor : Nico Schenk	**Bezug : Startseite.php**
Vorbedingung :	keine
Schritt 1 Schritt 2	Browser öffnen URL eingeben:
Nachbedingung :	Webshop Login erscheint

Tabelle 2 Testfall 1

2 Test	**Titel : Login**
Datum : 17.12.2015	**Art : Funktionalität**
Autor : Nico Schenk	**Bezug : Login.php**
Vorbedingung :	Startseite wird angezeigt
Schritt 1 Schritt 2 Schritt 3	Username eingeben Passwort eingeben Einloggen Button betätigen
Nachbedingung :	Eingeloggt im Webshop

Tabelle 3 Testfall 2

3 Test	**Titel : Login**
Datum : 17.12.2015	**Art : Blackbox**
Autor : Nico Schenk	**Bezug : Login.php**
Vorbedingung :	Startseite wird angezeigt
Schritt 1 Schritt 1 Schritt 2	Falschen Usernamen eingeben oder Falsches Passwort eingeben Einloggen Button betätigen
Nachbedingung :	Meldung über fehlerhafte Eingabe

Tabelle 4 Testfall 3

4 Test	**Titel : Register**
Datum : 17.12.2015	**Art : Funktionalität**
Autor : Nico Schenk	**Bezug : Register.php**
Vorbedingung :	Register Button wurde getätigt
Schritt 1 Schritt 2 Schritt 3	Username eingeben Passwort eingeben Registrieren Button betätigen
Nachbedingung :	Account im Textfile; Einloggen möglich

Tabelle 5 Testfall 4

5 Test	**Titel : Webshop**
Datum : 17.12.2015	**Art : Funktionalität**
Autor : Nico Schenk	**Bezug : Webshop.php**
Vorbedingung :	User ist eingeloggt
Schritt 1 Schritt 2	Kann die einzelnen Artikel sehen Kann Artikel in den Warenkorb legen
Nachbedingung :	Artikel befinden sich im Warenkorb

Tabelle 6 Testfall 5

6 Test	**Titel : Warenkorb**
Datum : 17.12.2015	**Art : Funktionalität**
Autor : Nico Schenk	**Bezug : Warenkorb.php**
Vorbedingung :	Artikel befinden sich im Warenkorb
Schritt 1 Schritt 2	Auf Warenkorb klicken Artikel im Warenkorb werden angezeigt
Nachbedingung :	Bestellung möglich

Tabelle 7 Testfall 6

7 Test	**Titel : Bestellung**
Datum : 17.12.2015	**Art : Funktionalität**
Autor : Nico Schenk	**Bezug : Payment.php**
Vorbedingung :	Artikel befinden sich im Warenkorb
Schritt 1 Schritt 2	Auf Bestellung klicken Bestellung wird in Text File geschrieben
Nachbedingung :	Ware ist bestellt

Tabelle 8 Testfall 7

8 Test	**Titel : Accounts**
Datum : 17.12.2015	**Art : Funktionalität**
Autor : Nico Schenk	**Bezug : Account.php**
Vorbedingung :	Accounts wurden erstellt
Schritt 1 Schritt 2	Account Übersicht aufrufen Alle registrierten Accounts werden aufgelistet
Nachbedingung :	Übersicht über vorhandene Accounts

Tabelle 9 Testfall 8

9 Test	**Titel : Accounts**
Datum : 17.12.2015	**Art : Funktionalität**
Autor : Nico Schenk	**Bezug : Account.php**
Vorbedingung :	Bestellungen wurden durchgeführt
Schritt 1	Bestellungen können angezeigt werden
Nachbedingung :	keine

Tabelle 10 Testfall 9

10 Test	**Titel : Logout**
Datum : 17.12.2015	**Art : Funktionalität**
Autor : Nico Schenk	**Bezug : Logout.php**
Vorbedingung :	User ist eingeloggt
Schritt 1 Schritt 2	Käufe sind getätigt Logout klicken
Nachbedingung :	User ist ausgeloggt

Tabelle 11 Testfall 10

Arbeitsjournal

Mittwoch, 16.12.2015

Tätigkeiten	Person	Aufwand geplant (Lektion)	Aufwand effektiv (Lektion)
• *Kurze Einführung durch Herrn Buchs*	Nico Schenk Thomas Campbell	0.5	0.5
• *Github installieren und konfigurieren*	Nico Schenk Thomas Campbell	1.5	1.5
• *Anforderungen definieren & Zeitplan erstellen*	Nico Schenk Thomas Campbell	1	1
• *Klassendiagramm*	Nico Schenk Thomas Campbell	1	1
Total:		**4**	**4**

Tagesablauf

- *Herr Buchs führte uns kurz in die objektorientierte Programmierung ein und schilderte uns den Ablauf mit Models, Controller und Views.*
- *Anschliessend haben wir uns um die Installation des Github gekümmert*
- *Nach der grossen Pause haben wir uns erneut zusammengesetzt, um die Anforderungen zu definieren und uns einen Zeitplan zusammenzustellen*
- *Zum Abschluss des Tages konnten wir das Klassendiagramm erstellen, welches leider etwas länger dauerte, weil wir es insgesamt 3 Mal neu machen mussten, aufgrund der Technik.*

Hilfestellungen

https://www.digitalocean.com/community/tutorials/how-to-install-git-on-centos-7
Schritt für Schritt Installation um Github in Betrieb zu nehmen

Reflexion

Nico Schenk

- *Die Einführung durch Herrn Buchs fand ich es etwas zu kurz. Für das Projekt steht nicht wahnsinnig viel Zeit zur Verfügung und da wir bisher nur prozedural gearbeitet haben mit PHP, könnte die Zeit sehr knapp werden. Es wird wohl bedeuten, dass am Wochenende relativ viel gearbeitet werden muss.*
- *Ansonsten verlief die Planung des Projektes sehr gut, bis auf die Erstellung des Klassendiagramms, was allerdings daran lag, dass die Seite web4b.mooo vor der Speicherung des Projekts nicht mehr verfügbar war.*

Thomas Campbell

- *Github ist sicher ein nützliches Werkzeug. Allerdings wäre vielleicht eine kurze Einführung oder Übung von Vorteil. Braucht einfach zu viel Zeit um sich auseinander zu setzen: Zeit ist knapp.*
- *Mit der Planungsarbeit mit meinem Projektpartner war ich sehr zufrieden. Wir haben beide Ideen und Vorschläge in die Struktur, Klassen usw. eingebracht.*
- *Der Zeitplan war etwas schwierig zu erstellen, da die gebrauchte Zeit pro Aufgabe nicht vom Anfang an immer klar war.*

Nächste Schritte

- *Use – Case Diagramm, Testfälle*

Tabelle 12 Arbeitsjournal Mittwoch

Donnerstag, 17.12.2015

Tätigkeiten	Person	Aufwand geplant (Lektion)	Aufwand effektiv (Lektion)
• *Use – Case Diagramm*	Nico Schenk Thomas Campbell	1	1
• *Leistungsbeurteilung*	Nico Schenk Thomas Campbell	1	1
• *Testfälle definieren*	Nico Schenk	2	2
• *Recherche im Internet*	Thomas Campbell	2	2
Total:		**4**	**4**

Tagesablauf

Nico Schenk

- *Als erstes heute Morgen haben wir noch unser Use – Case Diagramm erstellt.*
- *Anschliessend hatten wir unsere Leistungsbeurteilung*
- *Nach der grossen Pause kümmerte ich mich um die Erstellung der Testfälle, da uns ans Herz gelegt wurde, dass das Testing ein wesentlicher Bestandteil des Projektes sein soll*

Thomas Campbell

- *In der ersten Lektion haben wir zusammen das Use-Case Diagramm erstellt.*
- *Anschliessend hatten wir eine Leistungsbeurteilung*
- *In den dritten/ vierten Lektionen habe ich mich der Informationssuche gewidmet. Ich suchte nach Ansätzen und Ideen um eine Strukturierung für unseren Code zu finden und Verständnis für die nötigen Elemente aufzubauen(session, form handling, arrays).*

Hilfestellungen

http://stackoverflow.com/questions/6613473/php-application-structure
Forum Besprechung zu Strukturierung von PHP-Applikationen

Reflexion

Nico Schenk

- *Der heutige Tag verging wie im Flug. Die Planung konnte komplett abgeschlossen werden. Allerdings bin ich der Meinung, dass die Planung wie sie aktuell steht, am Ende nochmals überarbeitet werden muss, da es unmöglich ist, alles so umzusetzen wie es geplant ist. Zumindest nicht in der zu Verfügung gestellten Zeit. Prozedural wäre es vermutlich möglich gewesen.*

Thomas Campbell

- *Das Fertigstellen der Planungsarbeiten verlief eigentlich sehr gut, allerdings war uns klar, dass z.B der Zeitplan angepasst werden muss da uns die Zeit, mit unseren Anforderungen, nicht genügt. Trotzdem finde ich es sehr wichtig sich genügend Zeit nimmt für die Planung und ich glaube wir haben uns Ärger erspart indem wir sauber und genügend planten.*

Nächste Schritte

- *Programmierung*

Tabelle 13 Arbeitsjournal Donnerstag

Freitag, 18.12.2015

Tätigkeiten	Person	Aufwand geplant (Lektion)	Aufwand effektiv (Lektion)
• *Login View*	Nico Schenk	0.5	0.5
• *User –und Customer Model*	Nico Schenk Thomas Campbell	1.5	1.5
• *User Controller (index.php)*	Thomas Campbell	2	2
• *Dokumentationsvorlage aufbereiten* • *CSS erstellen*	Nico Schenk	2	2
Total:		**4**	**4**

Tagesablauf

Nico Schenk

- *Erstellung des LoginView.php*
- *Kurze Einführung in die Models, um zu verstehen wie diese gehandhabt werden*
- *Erarbeitung einer Vorlage für unsere Dokumentation, mit allen Titeln, Tabellen und Verzeichnissen*
- *Zum Schluss Ausarbeitung zu einem angenehmen CSS für den Shop*

Thomas Campbell

- *Erstellung der User und Customer Models*
- *Erklärung / Absprach mit Partner zu den Models (und umgekehrt zum LoginView)*
- *Erstellung UserController mit ersten Funktionalitäten (Login Funktion) und Index erstellen*

https://wiki.selfhtml.org/wiki/Schnell-Index/CSS
Nachschlagewerk für die Erstellung des CSS
https://www.youtube.com/watch?v=B-4PM9e1KaI
Youtube Video: Beispiel eines MVC Login in PHP

Reflexion

Nico Schenk

- *Das Projekt ist komplexer als angenommen. Heute zeigte sich deutlich, dass wir am Wochenende noch viel Zeit investieren müssen.*
- *Wir haben bereits festgestellt, dass es nicht möglich ist, alles umzusetzen was wir definiert haben. Ein grosses Problem dabei ist auch, dass wir die Möglichkeit Herrn Buchs zu Fragen nicht wahrnehmen können, da es sonst wieder von der Bewertung abgezogen wird.*

Thomas Campbell

- *Wir haben beide relativ schnell heute gemerkt, dass das Projekt, wie wir es realisieren wollen (mit MVC und Index als einziger Eingangspunkt), etliches komplexer ist als angenommen.*
- *Dank der sehr guten Erklärung eines Youtube Videos, könnte ich die Login Funktion in der MVC einigermassen umsetzen.*
- *MVC finde ich sehr interessant und ich hätte mir gewünscht, dass wir (wenn wegen der Zeit möglich) vielleicht zusammen eine einfaches Beispiel oder sogar das YT-Video anschauen.*

Nächste Schritte

- *Programmierung, Nerven bewahren*

Tabelle 14 Arbeitsjournal Freitag

Samstag, 19.12.2015 & Sonntag, 20.12.2015

Tätigkeiten	Person	Aufwand geplant (Lektion)	Aufwand effektiv (Lektion)
• *ShopView, CustomerView, CartView, OrderView, AdminView, UserListView, RegisterView*	Nico Schenk	4	5
• *Restlichen Models, UserController, CustomerController, index.php*	Thomas Campbell	4	5
• *Dokumentation angepasst* • *Einführung, Projektauftrag, Ressourcen* • *Arbeitsjournal nachführen*	Nico Schenk	1	1.5
• *Authentifizierung -> Text Datei einlesen* • *Daten (User&Customer) in Text Datei schreiben* • *Kommentieren* • *Testen*	Thomas Campbell	2	5
Total:		**4.5**	**6.5**

Tagesablauf

Nico Schenk

- *Erstellung aller Views, die für das Projekt verwendet werden. Definierung aller richtigen IDs um die Weiterleitung von der index.php zu den richtigen Funktionen zu gewährleisten.*
- *Dokumentieren der ersten Fortschritte. Erarbeiten der Einführung, des Projektauftrages und der Ressourcen.*
- *Nachführen der Arbeitsjournale, welche ich in den letzten Tagen vernachlässigt habe, da einfach keine Zeit vorhanden war.*

Thomas Campbell

- *Zuerst habe ich alle fehlenden Models erstellt und dazu (noch leere) Controllers*
- *Danach habe ich den UserController & CustomerContoller Login- bzw. Register -Funktionen hinzugefügt.*
- *Index Struktur erstellt (switch-case) und Weiterleitungen auf den korrekten Controller für Login oder registrieren.*
- *Kommentieren & Testen (Ziel lauffähige Lösung login/register bis Montag)*

Hilfestellungen

http://www.amazon.de
Produkte von Amazon ausgewählt um für unseren Shop zu verwenden
http://stackoverflow.com/questions/15130289/php-fwrite-new-line
Problemen mit fwrite und neue Zeile beheben

Reflexion

Nico Schenk

- *Beim Erstellen der Views konnte ich deutlich feststellen, dass der Aufbau wesentlich komplexer ist, wenn man für jeden Button und jeden Bereich eine bestimmte ID braucht, um diese von der Index.php überhaupt auslesen zu können. Mit einer direkten Weiterleitung wie im Prozeduralen Bereich war es definitiv schneller.*
- *Das Dokumentieren läuft mir inzwischen gut, beim Arbeitsjournal konnte ich allerdings feststellen, dass es schwer ist, alles nachzuführen, wenn bereits ein paar Tage vergangen sind und in der Zeit bereits wieder viel produktiv gearbeitet wurde.*

Thomas Campbell

- *Die Erstellung der Klassen verlief relativ mühelos, bis auf die etwas komplexeren Controllers.*

Vor allem hatte ich anfangs mühe die Controller Funktionen korrekt im Index aufrufen zu lassen und anhand übergebenen Parameter (username&passwort) zu überprüfen. • *Weil wir schon in der Klasse eine gute, einfache Übung zu Text-einlesen und schreiben hätten, könnte ich diese Funktionalität ziemlich schnell erstellen. Allerdings brauchte ich sehr viel Zeit um alle Schritte zu kombinieren (login info->index->usercontroller->checklogin->checkauthentifizierung->true/false->index-> erfolgreich eingeloggt)* • *Ich verbrachte zwar viel Zeit am Wochenende mit dem Codieren, aber ich habe wirklich das Gefühl als hätte ich viel dazugelernt. Auch bemerkte ich, dass z.B die Fehlereingrenzung mit jedem Projekt einfacher verlauft (vor allem wegen Tipps & Tricks vom Dozent).*
Nächste Schritte
• *Fertigstellung der letzten Funktionen*

Tabelle 15 Arbeitsjournal Samstag

Montag, 21.12.2015

Tätigkeiten	Person	Aufwand geplant (Lektion)	Aufwand effektiv (Lektion)
• *Fertigstellung der Order und Product Models und Controller*	Nico Schenk Thomas Campbell	3	3
• *Überarbeitung des Klassendiagramms*	Nico Schenk Thomas Campbell	0.5	0.5
• *Kurzes Meeting bzgl. Abgabe des Projektes*	Nico Schenk Thomas Campbell	0.5	0.5
• *Funktion Implementation: Login als Startseite*	Thomas Campbell	1	1
• *Überarbeitung CSS, Definierung Error Message*	Nico Schenk	1	1
Total:		**5**	**5**

Tagesablauf

- *Heute haben wir uns gemeinsam um die PHP Struktur gekümmert. Wir konnten die letzten Funktionen gemeinsam programmieren und die Webseite soweit fertigstellen.*
- *Anschliessend mussten wir unser Klassendiagramm etwas überarbeiten, da wir festgestellt haben, dass uns die Zeit nicht mehr reichen wird, um die restlichen geplanten Funktionen alle zu implementieren.*
- *Vor der Mittagspause haben wir ein kleines Meeting gehalten, um die restlichen Aufgaben zu verteilen.*
- *Zu Hause hat Thomas Campbell sich noch um die Implementation der Funktion für den Aufruf der Login Seite beim Start gekümmert.*
- *Nico Schenk kümmerte sich derweilen um die Überarbeitung des CSS, welche noch paar Kleinigkeiten aufwiesen und definierte noch die Error Messages.*

Hilfestellungen

https://wiki.selfhtml.org/wiki/CSS/Eigenschaften/Anzeige/display
Produkteanordnung im Shop mittels display: inline

Reflexion

- *Wir konnten heute effektiv und gut arbeiten. Alle Funktionalitäten konnten fertig gestellt werden und der Webshop läuft einwandfrei.*
- *Feststellen konnten wir, wie wichtig es ist, die ganzen Berechtigungen immer direkt zu vergeben, da es enorm zeitfressend ist, nach und nach erst alles umzustellen.*
- *Die Zeit wurde etwas zu grosszügig geplant. Zu Beginn dachten wir, wir können noch zwei bis drei Funktionen mehr integrieren, was leider nicht der Fall ist.*
- *Ansonsten konnten wir lernen, dass die Ausgabe im Browser wirklich speziell und anstrengend sein kann und es immer etwas zu verbessern geben würde.*

Nächste Schritte

- *Dokumentation fertigstellen, Arbeitsjournal fertigstellen, Projektabgabe*

Tabelle 16 Arbeitsjournal Montag

Dienstag, 22.12.2015

Tätigkeiten	Person	Aufwand geplant (Lektion)	Aufwand effektiv (Lektion)
• *Testfälle durchführen*	Nico Schenk Thomas Campbell	1	1
• *HTML Validierung*	Nico Schenk	1	1
• *Codesegmente für Dokumentation auswählen*	Thomas Campbell	1	1
• *Arbeitsjournale* • *Fazit*	Nico Schenk Thomas Campbell	2	2
Total:		**4**	**4**

Tagesablauf

- *Heute war es an der Zeit, das Projekt zu beenden, daher mussten die letzten Aufgaben noch erledigt werden*
- *Als erstes haben wir alle Testfälle durchgeführt und konnten diese erfolgreich abschliessen.*
- *Anschliessend hat Nico Schenk, die HTML Validierung durchgeführt, während Thomas Campbell sich um die Codesegmente für die Dokumentation gekümmert hat.*
- *Zum Schluss blieb uns genügend Zeit um die Dokumentation und die Arbeitsjournale nachzuführen und unser Fazit zu erarbeiten.*

Hilfestellungen

https://validator.w3.org/
https://html5.validator.nu/
HTML Validator

http://www.css-validator.org/
CSS Validator

Reflexion

- *Wir sind echt froh, dass wir so gut in der Zeit liegen, dass wir das Projekt heute erfolgreich beenden konnten. Das Arbeiten zu Hause hat uns gut weitergeholfen und wir hätten den Tag Verlängerung, welchen uns Herr Buchs gewährt hat, diesmal gar nicht gebraucht. Dennoch nahm es den Druck etwas raus.*
- *Die Testfälle verliefen ohne Probleme, was auch der immer wiederkehrenden Überprüfung aller Funktionen, während der Programmierung zu verdanken ist.*
- *Bei der HTML Validierung konnten einzelne kleine Fehler verbessert werden. Bei der CSS Validierung erhielten wir vier Fehler, welcher durch seine Redundanz auf insgesamt 18 Fehler gezählt wurde. Z.B. dass background kein Value für submit sei, allerdings war es nicht anders möglich, die Hintergrundfarbe des Buttons anzupassen. Fehler mit webkit.*
- *Ausserdem wurden alle Codesegmente in die Dokumentation eingefügt, welche relevant und kommentiert sind.*
- *Arbeitsjournale und Fazit erledigte sich zum Schluss praktisch von allein.*

Nächste Schritte

- *Projektabgabe*

Tabelle 17 Arbeitsjournal Dienstag

Ordnerstruktur

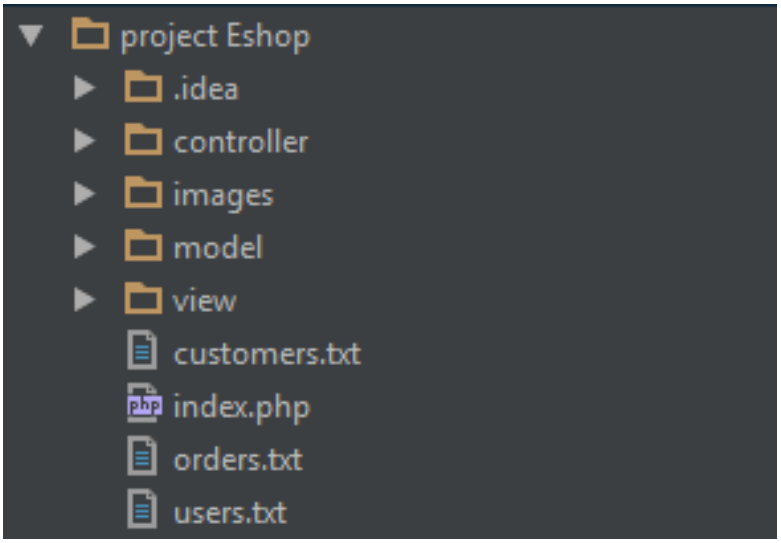

Abbildung IV Ordnerstruktur

Code Strukturierung

Wir haben unseren Code nach dem MVC (Model, View, Controller) Prinzip aufgebaut. MVC zielt auf eine Trennung von Geschäftslogik und Benutzerschnittstelle ab, so dass der Entwickler jeden dieser Bereiche bequem verändern kann, ohne andere zu beeinflussen. In MVC werden die Information (die Daten) und die Geschäftslogik durch das Model (Modell) repräsentiert. Der (auch "die") View (Präsentation) enthält Elemente der Benutzerschnittstelle, wie z.B. Text oder Formularelemente. Und der Controller (Steuerung) verwaltet die Kommunikation zwischen Model und View.

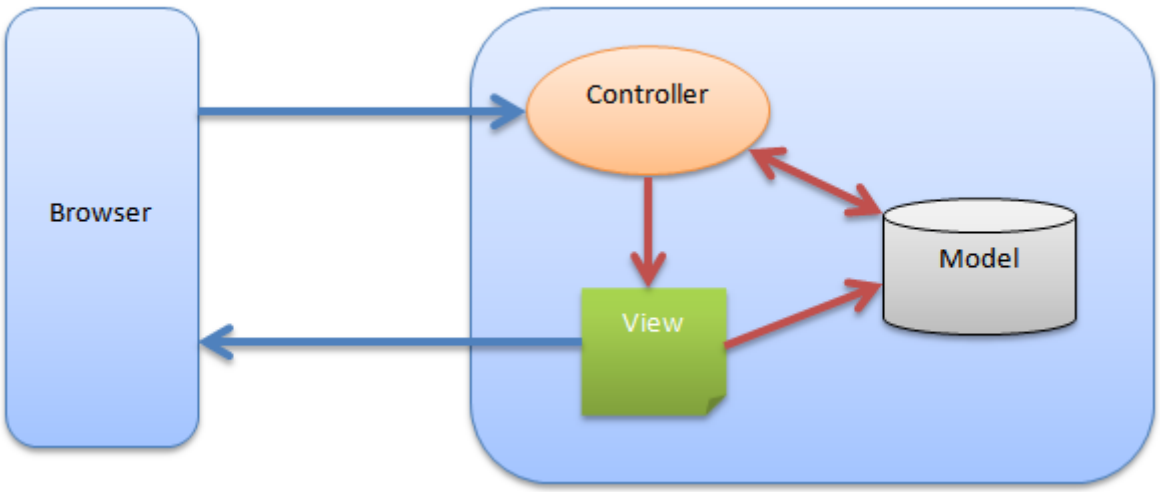

Abbildung V Allgemeine MVC Struktur

Spezifische Struktur

Bei jedem Request, z.B. Login oder Produkte dem Warenkorb hinzufügen, wird dieser dem Index übergeben. Dieser wählt den geeigneten Controller, führt die Funktion aus (falls vorhanden), anhand des passenden Modells. Anschliessend wird dem Index vom Controller einen Response übergeben, z.B. Login erfolgreich oder Produkt hinzugefügt. Der Index wählt das entsprechende View und verlinkt den User darauf.

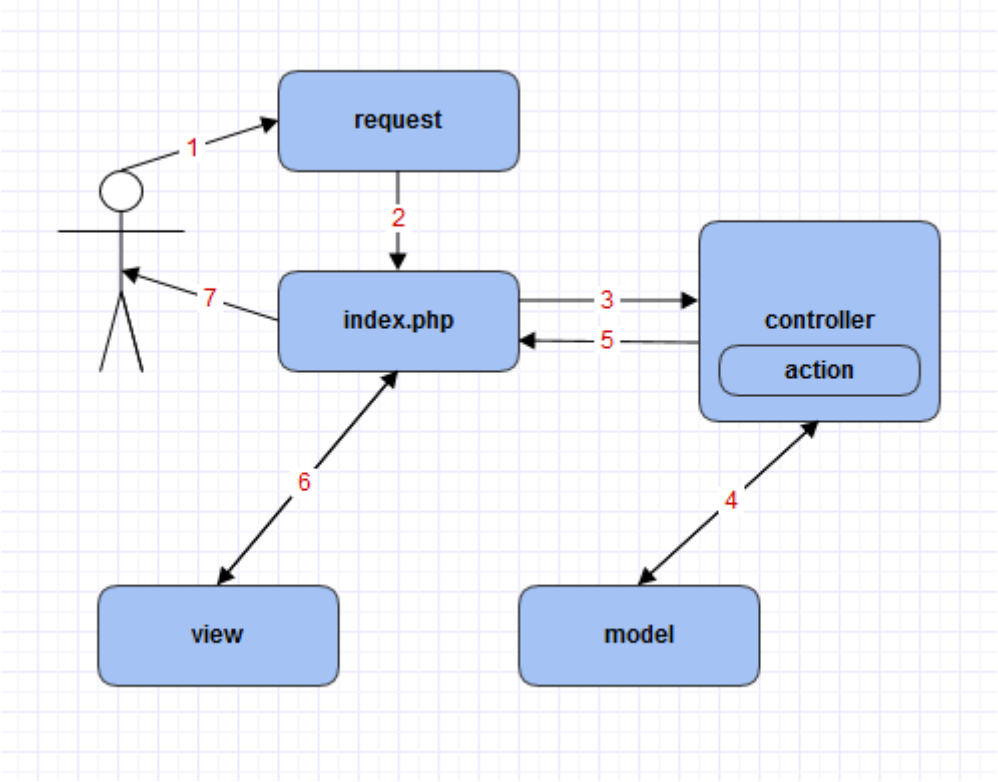

Abbildung VI spezifische Struktur Campsch

Code

Controllers

CustomerController

```php
<?php

    class CustomerController{

        public function __construct(){

        }

            public function registerCustomer() {
                //öffnet die customers.txt file zum schreiben
                $fp = fopen("./customers.txt", 'a');
                // PHP_EOL = newline systemübergreifend
                //schreibt die Kundendaten in einem text File: username:nachname:
                //vorname:strasse:PLZ:Stadt:Email (Trennzeichen = neue Zeile)
                fwrite($fp, $_SESSION['username'] . ':' . $_POST['name'] . ':' . $_POST['prename'] . ':'
                    . $_POST['street'] . ':' .$_POST['zip'] . ':' .$_POST['city'] . ':' .$_POST['email']. PHP_EOL);
                fclose($fp);
                return true;
            }

        public function listCustomers(){
            // erstellt ein session Array mit die zeilen des customers.txt file
            $_SESSION['customers'] = file($_SERVER["DOCUMENT_ROOT"] . "/project Eshop/customers.txt");
                foreach($_SESSION['customers'] as $customer) {
                    $ulines = split(":", $customer);
                    echo implode(":", $ulines). '<br>';
                }

        }



    }

?>
```

Abbildung VII CustomerController

ProductController

```php
<?php
    //absoluter Pfad
include_once($_SERVER["DOCUMENT_ROOT"] . "/project Eshop/model/ProductModel.php");

    class ProductController{
        public $products = array();
        private $nr;
        public function __construct(){
            $this->products[] = new ProductModel("48","Abstract World Card", "99.95","Yes");
            $this->products[] = new ProductModel("57","Waterfall Thailand", "89.95","Limited");
            $this->products[] = new ProductModel("99","Abstract City", "79.95","Yes");
            $this->products[] = new ProductModel("66","Abstract New York", "99.95","Yes");
            $this->products[] = new ProductModel("12","Abstract Nature", "90.00","Yes");
            $this->products[] = new ProductModel("15","Abstract Lion", "75.95","Yes");
            $this->products[] = new ProductModel("62","Sea Sunset", "69.00","Yes");
            $this->products[] = new ProductModel("34","Nature Buddha", "99.95","Yes");
            $this->products[] = new ProductModel("66","New York Night", "89.00","Limited");
            }

        //gibt anhand des indexes das gewählte Produkt ($nr) zurück
        public function listProduct($nr){
            //die Werten des Produkts werden in einem String mit eine neue Zeile am Schluss zurückgegeben
            $prod = $this->products[$nr]->getName()."  ".$this->products[$nr]->getAmount()."  "
                .$this->products[$nr]->getAvailability()."  ". PHP_EOL;
            return $prod;
            }
    }

?>
```

Abbildung VIII ProductController

ShoppingCartController

```php
<?php
include_once "ProductController.php";

    class ShoppingCartController{

        public function __construct(){

            }

        public function addItem($username, $product){

            $prod_controller = new ProductController();

            //wenn $_SESSION['products'] nicht gesetzt ist ist es ein array()
            $_SESSION['products'] = (isset($_SESSION['products']))? $_SESSION['products'] : array();

            // funktioniert gleich wie array_push
            $_SESSION['products'][] = $product;
            print_r($_SESSION['products']);

            //counter für anzahl Produkte im Warenkorb
            $_SESSION['prodAmount'] = count($_SESSION['products']);
        }
    }

?>
```

Abbildung IX ShoppingCartController

UserController

```php
<?php

include_once($_SERVER["DOCUMENT_ROOT"] . "/project Eshop/model/UserModel.php");

    class UserController{

        private $users = array();

        public function __construct(){

        }

        public function listUsers(){

            //session array von Benutzer aus Textdatei einlesen (jede Zeile einen Wert im array)
            $_SESSION['users'] = file($_SERVER["DOCUMENT_ROOT"] . "/project Eshop/users.txt");

                foreach($_SESSION['users'] as $user) {
                    $ulines = split(":", $user);
                    //fügt die Zeilen zu einem String mit jeder Benutzer auf eine neue Zeile
                     echo implode(":", $ulines). "<br>";
                }
        }

        public function login($username, $password){
            /*prüft ob die Funktion authenticate für die parameter $username, $password wahr oder falsch ausgibt
              falls wahr wird die übergeordnete Funktion login wahr übergeben */
            if($this->authenticate($username,$password)){

                //UserModel object  instanzieren (unötig da nicht gebraucht)
                $user = new UserModel($username, $password);
                //user Object an session gebunden
                $_SESSION['user'] = $user;

                return true;

            }else{
                return false;
            }
        }
```

Abbildung X UserController

```php
    static function authenticate(){
        $authentic = false;
        //session array von Benutzer aus Textdatei einlesen (jede Zeile einen Wert im array)
        $_SESSION['users'] = file($_SERVER["DOCUMENT_ROOT"] . "/project Eshop/users.txt");

        foreach($_SESSION['users'] as $user) {
            $ulines = split(":", $user);
            //variable $usern wird definiert als erster wert vom array(im text file erster Wert = Benutzername)
            $usern = trim($ulines[0]);
            //variable $pass wird definiert als zweiter wert vom array(im text file zweiter Wert = Passwort)
            $pass = trim($ulines[1]);
            //wenn der eingegebene Benutzername vorhanden ist und das eingegebene Passwort dazu passt
            if($_SESSION['username'] == $usern && $_SESSION['password'] == $pass){
                    $_SESSION['username'] = $usern;
                    $authentic = true;
            }
        }
        return $authentic;
    }

    public function logout() {

        session_unset();
        session_destroy();
    }

    public function checkAdmin($username){
        $admin = false;
        foreach($_SESSION['users'] as $user) {
            $ulines = split(":", $user);
            $usern = trim($ulines[0]);
            //variable $role wird definiert als dritter wert vom array(im text file dritter Wert = Rolle)
            $role = trim($ulines[2]);

            if($_SESSION['username'] == $usern){
                if($role == "Admin") {
                    $admin=true;
                }
            }
        }
        return $admin;
    }
```

Abbildung XI UserController

```php
    public function registerUser() {
        if(isset($_POST['username'])){
            //prüft ob die zwei eingegebenen Passwörter gleich sind
            if($_POST['password1'] == $_POST['password2']){
                //öffnet die Datei
                $fp = fopen("./users.txt", 'a');
                echo "open file";
                //der eingegebene Username, Password und die standard Rolle Kunde werden in einer Zeile
                // mit Trennzeichen ":" in die Textdatei geschrieben
                fwrite($fp, $_POST['username'] . ':' . $_POST['password1'] . ':' . 'Kunde' . PHP_EOL);
                fclose($fp);
                return true;
            }else {
                return false;
            }
        }
    }
?>
```

Abbildung XII UserController

Models

ProductModel

```php
<?php
    class ProductModel{

        private $id;
        private $name;
        private $amount;
        private $avail;

        public function __construct($id=null,$name=null,$amount=null,$avail=null){
            $this->id = $id;
            $this->name = $name;
            $this->amount = $amount;
            $this->avail = $avail;
        }

        public function getId(){
            return $this->id;
        }

        public function getName(){
            return $this->name;
        }

        public function getAmount(){
            return $this->amount;
        }

        public function getAvailability(){
            return $this->avail;
        }
    }
?>
```

Abbildung XIII ProductModel

View

ShopView

```php
    <!-- Wrapper -->
    <div id="wrapper">
        <div class="shell">
            <!-- Header -->
            <div id="header">
                <!-- Logo -->
                <h1 id="logo"><a title="home" href="../index.php?op=BacktoShop">Abstract Pictures</a></h1>
                <div id="cart">
                    <p><a title="shopping bag" name="op" href="../index.php?op=view-Item">Shopping Cart: 

                            <!-- wenn die session Varible 'prodAmount' (anzahl Produkte) lehr ist soll 0 ausgegeben
                            werden, sonst die gesetzte Anzahl Produkte im Warenkorb-->
                            <?php session_start(); if(!isset($_SESSION['prodAmount'])){echo "0";
                            }else{echo $_SESSION['prodAmount'];} ?> items</a></p>

                    <p><a title="logout" name="op" href="../index.php?op=logout">Logout</a></p>
                </div>
            </div>
        </div>
    </div>
        <!-- END Header -->
```

Abbildung XIV ShopView

LoginView

```php
<p>Please login to use our webshop. <br> If you don't have an account, please register.</p>
<!--wenn dem URL err=37 mitgegeben wird(also fehler bei login) wird ein Error Text ausgegeben-->
<?php if(@$_GET['err'] == 37) {?>
    <div class="error-text">Login incorrect. Please try again</div>
<?php } ?>
```

Abbildung XV LoginView

CartView

```php
<?php
session_start();
//falls session Variable products lehr ist (keine Produkte im Warenkorb)
if(!isset($_SESSION['products'])){
    echo "No products in Shopping cart"."<br>";
}
else{
    /* wenn die $_SESSION['products'] array nicht lehr ist
    soll sie mit dem Trennzeichen neue Zeile (<br>) ausgegeben werden */
    echo implode("<br>",$_SESSION['products']);
    echo "<br>";
}
?>
```

Abbildung XVI CartView

OrderView

```php
<!--Neuer OrderController wird instanziiert und Funktion (listOrders) wird aufgerufen-->
<?php
    include_once("../controller/OrderController.php");
    $order_controller = new OrderController;
    $order_controller->listOrders();
?>
```

Abbildung XVII OrderView

index.php

```php
<?php

//Einbinden von Controllers
include_once 'controller/UserController.php';
include_once 'controller/ShoppingCartController.php';
include_once 'controller/ProductController.php';
include_once 'controller/CustomerController.php';
include_once 'controller/OrderController.php';

//Session wird gestartet
session_start();

//op = operation, Server übergibt den wert von 'op', von allen Server Variablen($_POST,$_GET,usw.)
@$op = $_REQUEST['op'];

//pr = productnummer die vom Shop ausgewählt wird, -1 weil Index bei 0 anfängt
@$pr = $_REQUEST['pr'] -1;

//Controllers werden instanziert, damit wir ihre Funktionen im index brauchen können
$user_controller = new UserController();
$cart_controller = new ShoppingCartController();
$prod_controller = new ProductController();
$cust_controller = new CustomerController();
$order_controller = new orderController();

/*****************************
Anhand des Wertes von op, also der gesuchten Funktion im Index,
werden auf verschiedene Seiten gezeigt und /oder verschiedene Funktionen eines
Controllers durchgeführt. z.B versucht der User einzuloggen wird im Forumular
über $_POST login button (mit name "op"!)die login Funktion des Indexes
aufgerufen. Ist die Authentifizierung erfolgreich wird der User auf den Shop
(ShopView.php) verlinkt.
Alle Verlinkungen verlaufen durch den Index.
*****************************/
switch($op){

    //verweist einen leeren op an die Loginseite (erste Ladung des Shops)

    case "":
        header("Location:./view/LoginView.php");
    break;
```

Abbildung XVIII index.php

```php
//basic login logout operations

case 'Login':
    //session Variablen
    $_SESSION['username'] = $_POST['username'];
    $_SESSION['password'] = $_POST['password'];
    $username = $_SESSION['username'];
    $password = $_SESSION['password'];
    //prüft ob der user existiert und ob das password stimmt
    if($user_controller->login($username, $password)){
        if($user_controller->checkAdmin($username)){
            // admin view
            header("Location:view/AdminView.php");
        }else{

            //normaler User link zum Shop
            header("Location:view/ShopView.php");
        }
    }else {
        /*wenn der Username oder das Passwort nicht stimmt wird der User
            züruck an die Loginseite verwiesen mit einer Error Meldung*/
        header("Location:./view/LoginView.php?err=37");
    }
break;

case 'BacktoLogin':
    header("Location:./view/LoginView.php");
break;

case 'BacktoShop':
    header("Location:./view/ShopView.php");
break;


// register
case 'Register':
    header("Location:./view/RegisterView.php");
break;
```

Abbildung XIX index.php

```php
//Next - customer Daten eingeben
case 'Next':
    $_SESSION['username'] = $_POST['username'];
    //prüft ob der User erfolgreich in die Text Datei gespeichert wurde
    if($user_controller->registerUser()){
    header("Location:./view/CustomerView.php");
    }else{
    /*wenn die Passwörter nicht identisch sind wird der User
    zurück an die RegisterView geschickt und eine Error Meldung wird ausgegeben*/
    header("Location:./view/RegisterView.php?err=12");
    }
break;

//Send
case 'Send':
    //prüft ob die Customer Daten erfolgreich gespeichert wurden
    if($cust_controller->registerCustomer()){
    header("Location:./view/LoginView.php");
    }else{
    header("Location:./view/RegisterView.php?err=13");
    }
break;


//logout
case 'logout':
    $user_controller->logout();
    header("Location:view/LoginView.php");
break;
```

Abbildung XX index.php

```
    // admin operations

    case 'show-Userlist':
        header("Location:view/UserListView.php");
    break;

    case 'show-Customers':
        header("Location:view/CustomerListView.php");
    break;

    case 'show-Orders':
        header("Location:view/OrderView.php");
    break;

    case 'Back':
        header("Location:view/AdminView.php");
    break;


    // cart operations

    case 'add-Item':
        //dem Warenkorb wird ein Produkt anhand pr (gewählter Produktnummer) hinzugefügt
        $cart_controller->addItem($_SESSION['username'], $prod_controller->listProduct($pr));
        //danach wird die Seite neu geladen
        header("Location:view/ShopView.php");
    break;

    case 'view-Item':
        header("Location:view/CartView.php");
    break;

    case 'Order':
        //eine Bestellung wird erstellt
        if($order_controller->createOrder()){
        //falls erfolgreich soll der User ausgeloggt werden
        $user_controller->logout();
        header("Location:view/LoginView.php");
        }
    break;
```

Abbildung XXI index.php

```
    //default -
    //falls der User selber bei op= etwas eingibt soll er ausgeloggt werden
    default:
        $user_controller->logout();
        header("Location:view/LoginView.php");
    break;

}

?>
```

Abbildung XXII index.php

Quellen

- http://stackoverflow.com/questions/13102489/passing-multiple-variables-to-another-page-in-url
- http://www.w3schools.com/php/php_superglobals.asp
- http://stackoverflow.com/questions/5438216/how-do-i-post-button-value-to-php
- http://www.w3schools.com/php/php_arrays.asp
- https://github.com/csbe/133
- http://stackoverflow.com/questions/1547899/which-characters-make-a-url-invalid
- http://stackoverflow.com/questions/15130289/php-fwrite-new-line
- http://stackoverflow.com/questions/6613473/php-application-structure
- https://wiki.selfhtml.org/wiki/CSS/Eigenschaften/Anzeige/display
- http://www.amazon.de
- https://wiki.selfhtml.org/wiki/Schnell-Index/CSS
- https://www.digitalocean.com/community/tutorials/how-to-install-git-on-centos-7
- https://www.youtube.com/watch?v=B-4PM9e1KaI
- http://www.codeproject.com/KB/aspnet/344292/mvc.PNG

Fazit

Nico Schenk

Das Projekt gefiel mir an sich sehr gut. Der Lerneffekt war hier gross und zeigte PHP von einer ganz neuen Seite. Leider war das Projekt komplexer umzusetzen als gedacht, da die Einführung in die objektorientierte Programmierung mit PHP etwas zu wünschen übrig liess.

Die selbstständige Einarbeitung in die objektorientierte PHP Programmierung gestaltete sich etwas schwierig, was dazu geführt hat, dass bei der Umsetzung des Projekts, dennoch etwas prozedurale Codesegmente eingeflossen sind. Die Projektstruktur allerdings wurde komplett objektorientiert gehalten.

Im Vergleich zu der Lösung von Herrn Buchs konnte ich feststellen, dass unsere Arbeit definitiv verbesserungswürdig ist, da wir leider nie auf die Vererbung von Methoden und Eigenschaften von Klassen auf andere Klassen zurückgegriffen haben. Diese „extends“ Methode ist mir bereits durch Java bekannt und daher wüsste ich auch bereits wie sie eingesetzt wird und fände es bestimmt interessant, darauf in Zukunft näher einzugehen.

Das Modul „Webapplikation realisieren“ war mein letztes Applikationsmodul in meiner Ausbildung an der CsBe und es war ein guter Abschluss. Ich bin froh, dass wir noch einmal eine Teamarbeit machen durften, da ich in der Vergangenheit immer sehr gerne mit Thomas Campbell zusammen gearbeitet habe. Schlussendlich kann ich sagen, dass ich nicht ausschliesse, nach meinem EFZ mich in die Applikationsrichtung noch weiterzubilden, für den Moment allerdings fühle ich mich in der Systemtechnik wohler.

Thomas Campbell

Dieses Projekt hat mir sehr gut gefallen. Die Aufgabenstellung von Herrn Buchs war ziemlich ungenau was gewisse Vor- und Nachteile hatte. Einerseits hatten wir sehr viel Freiheit in der Lösungswahl und der Weg dahin. Anderseits waren damit die Zeitplanung und der wirkliche Umfang des Projekts schwer einzuschätzen, so dass wir z.B. länger arbeiten müssten als zuerst geplant. Ein weiterer Vorteil war, dass wir gezwungen waren selber nach Ansätzen Ideen usw. zu suchen was das Verständnis für PHP und dessen Implementierung (+Struktur) sehr stärkt.

Meiner Meinung nach war das interessanteste am Projekt die MVC (Model-View-Controller) Strukturierung mit dem Index als einziger Eingangspunkt. Die Vorteile einer solchen Struktur lassen sich einfach erblicken und ihre dynamische Erweiterbarkeit leicht erklären. Ich hätte mir gewünscht (bzw. würde für zukünftige Modulen vorschlagen), dass wir in die Klasse die Theorie, Vorteile usw. des MVC angeschaut hätten. Meiner Meinung nach wurde dies die Objektorientierte Planung und Realisierung verständlicher machen bzw. dessen Vorteile klar gemacht.

Die Arbeit im Team verlief sehr gut. Ich hatte den Eindruck als könnten wir unser erlerntes Wissen gut austauschen. Ich habe schon andere Projekte mit Nico Schenk durchgeführt und wir könnten immer gut im Team arbeiten. Ich könnte mich auf ihn verlassen, was wichtig ist in einem solchen Projekt.

Ich bedaure, dass wir nicht von Anfang an konsequent mit GitHub geschafft haben. Der Zeitaufwand um sich damit wirklich gut zu arbeiten und zu verstehen war zu hoch. In der Zukunft werde ich aber damit arbeiten, da im Verlauf des Projektes die Vorteile gegenüber z.B Dropbox ersichtlich waren.

Zusammenfassend war dies ein herausforderndes aber sehr lernreiches Projekt. Durch die ungenaue bzw. dynamisch angepasste Anforderungen & Rahmenbedingungen war die Zeitplanung etwas schwierig dafür der Lerneffekt grösser (selber nach Ansätzen suchen->weshalb was wo einsetzen).

Benutzeranleitung

Kunde

Aufruf des Webshops	Über index.php -localhost/project Eshop
Registrierung eines Benutzers	Klicken Sie „Register“
	Geben Sie einen Usernamen ein
	Geben Sie ein Passwort ein
	Geben Sie das Passwort erneut ein
	Klicken Sie „Next“
Eintragen der Kundendaten	Füllen Sie das nachfolgende Formular aus
	Klicken Sie auf „Send“
Einloggen in den Webshop	Geben Sie einen Usernamen ein
	Geben Sie ein Passwort ein
	Klicken Sie auf „Login“
Produkte in den Warenkorb legen und anzeigen	Klicken Sie auf „Buy Product“
	Klicken Sie auf „Shopping Cart“
Bestellung auslösen	Klicken Sie auf „Order“ im Shopping Cart
	8-tung! Sie werden ausgeloggt und Ihre Session wird zerstört!
Logout	Klicken Sie auf Logout

Tabelle 18 Benutzeranleitung – Kunde

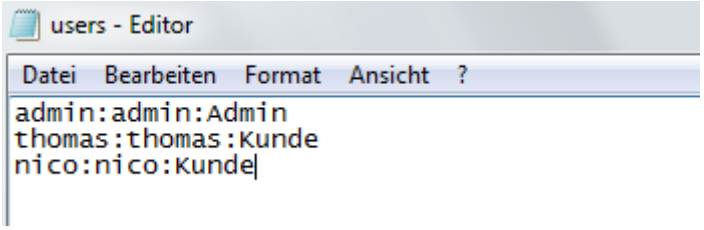

Abbildung XXIII Bestehende Users

Admin

Aufruf der Adminfunktionen	Einloggen mit Admin Account
	Username: admin Passwort: admin
Anzeigen der User	Klicken Sie „Show UserList“
Anzeigen der Kundendaten	Klicken Sie „Show Customers“
Anzeigen der Bestellungen	Klicken Sie „Show OrderList“
Shop anzeigen	Klicken Sie auf den Banner (Logo)

Tabelle 19 Benutzeranleitung – Admin

Admin Funktionen

UserList

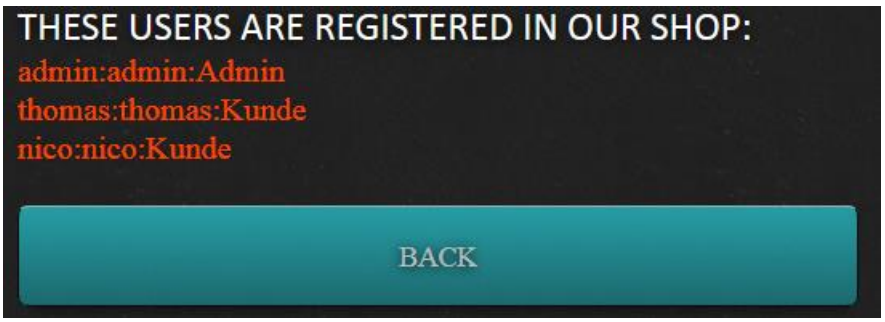

Abbildung XXIV UserList

Customers

```
1  nico:schenk:nico:teststreet 7:700:london:asfasl@gngn.com
2  thomas:campbell:thomas:coolstreet 65a:bt7i 6rl:new york:the_tester@gmail.com
```

Abbildung XXV Customers

OrderList

```
thomas
coolstreet 65a
new york
Abstract Lion  75.95  Yes
Abstract City  79.95  Yes
Waterfall Thailand  89.95  Limited

nico
teststreet 7
london
Waterfall Thailand  89.95  Limited
Abstract World Card  99.95  Yes
Waterfall Thailand  89.95  Limited

```

Abbildung XXVI OrderList

Abbildungsverzeichnis

Tabellenverzeichnis