Inhaltsverzeichnis

1 Technische Grundlagen

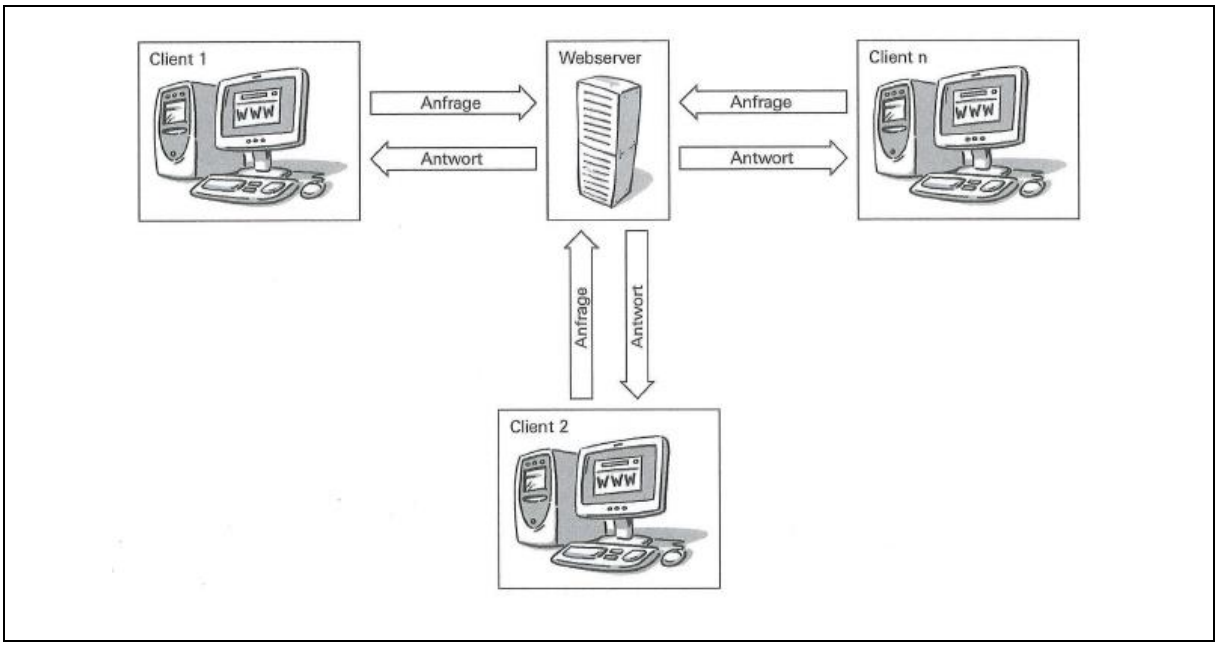
Bestimmt haben Sie schon einmal im Internet gesurft. Sie können in die Adresszeile Ihres Browsers die Adresse einer Webseite eingeben und diese erscheint – je nach Art der Verbindung zum Internet – mehr oder weniger schnell auf Ihrem Bildschirm. Was dabei genau abläuft, besprechen wir in diesem Modul M133.

1.1 Wie funktioniert das Web?

Kommunikation zwischen Browser und Server: Zeichnen wir der Reihe nach die einzelnen Aktionen auf…

1. Client frag Webserver an
2. Server verarbeitet die Anfrage
3. Server sendet Client eine Antwort

Kommunikation zwischen Browser und Server



HTTP-Protokoll / GET-Anfrage

Die Kommunikation zwischen Browser und Server läuft über ein standardisiertes Protokoll, das HTTP-Protokoll:

- Hypertext Transport Protocol

HTTP wurde von der Internet Engineering Task Force (IETF) und dem **World Wide Web Consortium** (W3C) standardisiert. Es gibt weiter auch **HTTPS** für die Verschlüsselung übertragener Inhalte – und darauf aufbauende Standards wie das Übertragungsprotokoll **WebDAV**.

Eine Anfrage kann wie folgt aussehen:

- HTTP/1.1 /index.html

Der Server sendet folgendes zurück:

```
HTTP/1.1 200 OK
Date: Sun, 28 Jun 2015 08:20:00 GMT
Server: FrontPage/5.0.2.2510 Signature/2.0 Apache/1.3.26 (Unix)
Last-Modified: Thu, 23 Jun 2015 13:45:31 GMT
Etag: "166b2el-2fc-36c2d751"
Accept-Ranges: bytes
Content-Length: 7 64
Connection: close
Content-Type: text/html
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC 11 - //W3C // DTD XHTML 1.0 S t r i c t / / E N"
"http://www.w3.org/TR/xhtmll/DTD/xhtmll-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de">
<head>
<title>Seitentitel</title>
</head>
<body>
Seitentext
</body>
</html>
```

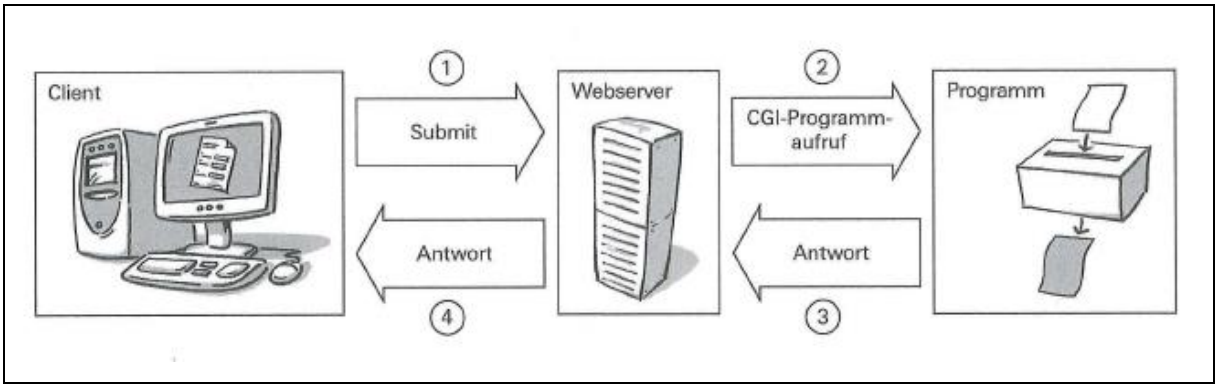
Die komplette HTTP-Spezifikation finden Sie im Internet auf der Site des **World-Wide-Web-Consortiums**:

- W3C

Dynamische Seitengenerierung

Wenn ein Internet-Benutzer eine dynamische Seite anfordert, passiert Folgendes:

1. Der Client fordert die Seite an und der Webserver erkennt sie anhand ihrer Dateiendung als dynamische Seite
2. Der Webserver startet das dafür verantwortliche CGI-Programm und übergibt dem Programm den Aufruf, den er bereits vom Benutzer erhalten hat
3. Das CGI-Programm führt das Skript aus, das in der gewünschten Seite gespeichert ist. Dabei werden allfällige Werte, die vom Benutzer mitgegeben wurden, berücksichtigt. Das Resultat bzw. den Output gibt das CGI-Programm an den Webserver zurück
4. Der Webserver wiederum leitet das Resultat bzw. den Output an den Internet-Benutzer zurück, der die Seite angefordert hat. Diese wird beim Benutzer angezeigt



CGI (Common Gateway Interface)

Das Common Gateway Interface (CGI) ist eine verbreitete, plattformunabhängige und standardisierte Methode zum Aufruf von Programmen. Es gibt ein breites Einsatzspektrum für CGI-Skripts:

1. Webshop
2. Buchungswebsite
3. ...
4.
5.
6.

1.2 HTML und CSS

HTML ist eine einfach aufgebaute Sprache zur Strukturierung von Informationen. Text wird mit sog. **HTML-Tags** ergänzt. HTML-Tags formatieren Textabschnitte, Überschriften, Listen oder Bilder, etc.

Jeder HTML-Tag bildet dabei eine Klammer um das auszuzeichnende Element.

Beispiel:

- <tag></tag>

Während **HTML** der Strukturierung von Informationen dient,
verwendet man **CSS** zur Beschreibung der Darstellung.

CSS-Anweisungen können direkt in ein HTML-Dokument geschrieben werden, besser ist es jedoch, die Darstellungsangaben in separate Textdateien zu schreiben und diese Dateien in Form von Links einzubinden.

Beispiel:

- .yourmom { color: #ffffff }

1.3 Serverseitige Programmierung

Eine Web-Applikation besteht immer aus (mindestens) zwei Teilen:

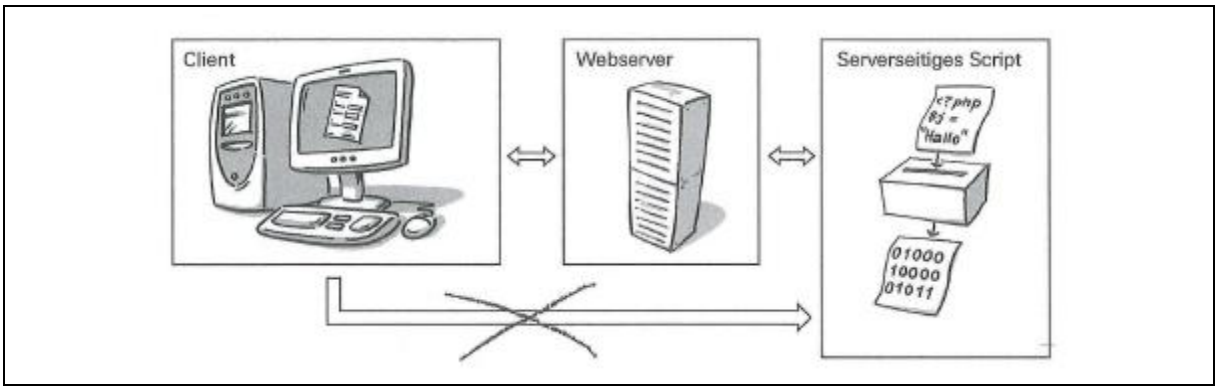
Da ist erstens der Server (oftmals Backend genannt), auf dem die eigentliche Applikation läuft, und zweitens der Webbrowser des Benutzers (auch als Frontend bezeichnet), der für die Anzeige der Inhalte und die Interaktion mit dem Benutzer zuständig ist.

Die folgende Tabelle enthält eine Liste der am meisten genutzten Skript- und Programmiersprachen. Ergänzen Sie die Tabelle:

Bezeichnung	Name	Form	Extension
ASP.NET	Active Server Pages VBScript	Kompiliert / Skript	.asp .aspx
JSP	Java Server Pages	Kompiliert / Skript	.jsp
PHP	Hypertext Preprocessor	Skript	.php
Python	Python Software Foundation	Skript	.py

Die in der obigen Tabelle aufgeführten Sprachen haben gemein, dass sie nur auf dem Server ablaufen. Der Benutzer sendet über seinen Browser Daten an den Server. Der Server leitet die Daten über die CGI-Schnittstelle an das entsprechende Skript weiter. Das Skript wiederum verarbeitet die Daten und sendet die Resultate zurück an den Webserver. Dieser erst leitet die Daten an den Benutzer zurück. Der Benutzer kommt also nie direkt in Kontakt mit dem Skript, das seine Daten verarbeitet.

Weil es nur auf dem Server läuft, nennt man es **serverseitig**.



Der Client hat keinen direkten Kontakt mit einem serverseitigen Skript!

Vor- und Nachteile von serverseitigen Skripts:

Vorteile:	Frontend muss keine Javacsript beinhalten
	Meistens sehr schnell
	Gut für Seiten welche Datenbankbasiert sind
Nachteile:	Bei sehr grossen Datenmengen braucht es lange
	wenn nicht AJAX benutzt muss Seite immer neugeladen werden
Anwendung:	CMS

1.4 Clientseitige Programmierung

Moderne Webbrowser können sehr viel mehr als nur HTML-Seiten darstellen. Sie enthalten komplexe **Skript-Interpreter**. Eine typische (und heute am meisten eingesetzte) clientseitige Sprache ist **JavaScript**. Alle modernen Browser können sehr gut mit JavaScript umgehen. Der JavaScript-Code wird zusammen mit den Webseiten auf den Computer des Benutzers übertragen und erst dort ausgeführt. JavaScript ist eine sehr mächtige Sprache, die eng mit HTML und CSS verknüpft ist. In den letzten Jahren hat sich JavaScript von einer Sprache für einfache Aufgaben und solche, die keine Interaktion mit anderen Teilen des Servers (z. B. einer Datenbank) bedingen, zu einer Sprache für komplexe Applikationen entwickelt. Heute werden mit JavaScript ganze Web-Anwendungen geschrieben, die vollständig im Browser des Benutzers ablaufen. Solche Applikationen können Netzwerkverbindungen zu einem Server aufbauen und beliebige Daten austauschen. Umfangreiche Bibliotheken erlauben es den Programmierern, raffinierte und auch sehr komplexe **Benutzerinteraktionen** einfach zu programmieren.

Ein gewichtiger Nachteil der clientseitigen Programmierung ist die (fehlende) **Sicherheit**: Der Benutzer kann den Code jederzeit einsehen und die Daten verändern, die verarbeitet werden. Dieses Manko existiert bei serverseitigen Skripts nicht, weil dort der ganze Datenverkehr zuerst vom Server abgewickelt wird.

Vor- und Nachteile von clientseitigen Skripts:

Vorteile:

Nachteile:

Anwendung:

1.5 Applikation und Sitzungsverfolgung

HTTP-Cookies

Ein Cookie ist eine **Textinformation**, die die besuchte Website («Server») über den Browser im Rechner des Betrachters («Client») platziert. Der Cookie wird entweder vom Webserver an den Browser gesendet oder von einem Skript (etwa JavaScript) in der Website erzeugt.

Der Client sendet die Cookie-Information bei späteren, **neuen Besuchen dieser Seite** mit jeder Anforderung wieder an den Server.

Im Cookie wird vom Server eine eindeutige **Session-ID** gespeichert, um genau diesen Client bei weiteren Aufrufen **wiederzuerkennen**.

Sind Cookies gefährlich?

> Cookies werden zwar auf die Festplatte des Benutzers geschrieben, aber da sie nur aus einem String bestehen und sich nicht wie ein Virus verbreiten oder unkontrollierte Aktionen ausführen, sind sie ungefährlich. Cookies können also zum Beispiel weder Daten löschen noch Mails versenden oder sich reproduzieren

Was kann ich gegen Cookies tun?

Wenn Sie auf Nummer sicher gehen wollen, haben Sie bei Ihrem Browser die Möglichkeit, die Verwendung von Cookies einzuschränken oder ganz abzuschalten. Der Browser verbietet dann die Speicherung von Cookies generell

Welchen Effekt stellt sich ein, wenn Sie Cookies ausschalten?

Sie müssen auf viele interaktive Web-Dienste verzichten, so wie beispielsweise Web-Shops, Web-Mail, etc.

Sind Cookies sicher?

Da Cookies irgendwo auf der Festplatte des Benutzers abgelegt werden, kann man grundsätzlich davon ausgehen, dass die Möglichkeit der Veränderung von Cookie-Daten besteht. Aus diesem Grund sollte man sicherheitskritische Werte nicht in Cookies abspeichern

Browser-Sessions

Eine **Session** ist ein Mechanismus, der es einer Web-Applikation erlaubt, Daten eines Benutzers **während der ganzen Dauer** seines Besuchs auf der Webseite zu speichern. Die Daten einer Session, dazu gehören sowohl die **Session-ID** als auch die **Nutzdaten** (Benutzer-ID, Warenkorbinhalt etc.), speichert der Webserver standardmässig in einem dafür bestimmten Verzeichnis auf der Festplatte, häufig ist es das temporäre Verzeichnis des Betriebssystems (/tmp). Solch eine Datei sieht dann so aus:

```
/tmp/sess_hvb0es1qdv5o91ogspmfk9ck77
Use-
rID|i:212;Warenkorbinhalt|a:2:{i:0;s:8:"Artikel1";i:1;s:8:"A
rtikel2";}
```

Was wird bei Browser-Sessions gespeichert?

Bei Browser-Sessions wird nur die Session-ID beim Benutzer gespeichert und nicht alle Variablen. Deshalb ist eine Session um einiges besser gegen Manipulationen geschützt als Cookies. Das soll aber nicht darüber hinwegtäuschen, dass auch Sessions keinen 100%igen Schutz gegen Manipulationen bieten.

Wozu werden Browser-Sessions verwendet?

Browser-Sessions können beispielsweise für Shops mit Warenkorb eingesetzt werden, für geschützte Bereiche (Extranets), Telebanking etc.

Zusammengefasst

Das **HTTP-Protokoll** besteht aus einfachen Anweisungen, die eine Kommunikation zwischen Client (Browser) und dem Webserver ermöglichen.

Sie haben erfahren, dass es sich bei **CGI** um eine Schnittstelle handelt, mit deren Hilfe ein Webserver ein weiteres Programm oder Skript aufrufen kann. Sie kennen somit den Unterschied zwischen **serverseitigen Skripts**, die nur auf dem Server ausgeführt werden, und clientseitigen Skripts, die zur Ausführung zuerst auf den Computer des Benutzers kopiert werden müssen.

Cookies bestehen aus einer einzigen Zeile Text, die über den Browser auf dem Computer des Besuchers einer Webseite abgelegt und gelesen oder gelöscht werden kann. Um Daten auf dem Computer des Besuchers zwischen zu speichern, gibt es nur diese Möglichkeit.

Cookies werden vor allem für **personalisierte Dienste** eingesetzt. Ein Benutzer, der z. B. eine Webseite seinen spezifischen Bedürfnissen anpasst, trifft immer wieder die gleichen Ein-stellungen an, wenn er diese Seite besucht. Dies geschieht zumindest so lange, bis er das betreffende Cookie löscht oder bis dieses von alleine abgelaufen ist. Die Lebensdauer eines Cookies lässt sich nämlich auch individuell definieren.

Cookies sind **sicherheitstechnisch unbedenklich**. Unter dem Aspekt des **Datenschutzes** besteht ein gewisses Risiko, da mithilfe von Cookies grundsätzlich festgestellt werden kann, ob jemand schon einmal eine Webseite besucht hat.

Eine **Browser-Session** ist ein Mechanismus, mit dem man Besuchern einer Webseite **Variablen zuweisen** kann, die während des ganzen Besuchs erhalten bleiben, auch wenn der Besucher zwischenzeitlich andere Sites besucht.

Kontrollfragen

1. Beschreiben Sie in wenigen Sätzen den Unterschied zwischen dynamischen und statischen Webseiten!

   statisch: websites die sich sehr selten verändern
   dynamisch: websites die sich bei jedem neuen laden verändern und variable daten beinhalten

2. Erklären Sie stichwortartig den Kommunikationsvorgang zwischen Client und Server, wenn ein Benutzer eine dynamische Seite anschauen möchte!

   1. Client fragt Server nach Resource
   2. Server entscheidet ob Resource dynamisch / statisch ist

   statisch -> sendet dem Client die Resource
   dynamisch -> verarbeitet ein Skript und sendet dessen Output

3. Erklären Sie mit wenigen Worten, was ein Cookie ist!

   Ein kleiner Datenspeicher für Session Ids, und Benutzereinstellungen,
   welches vom Client immer mit an den Browser gesendet wird, und dann vom Browser verworfen, verändert oder belassen werden kann.

4. Nennen Sie den Grund, warum Cookies nicht geeignet sind, um sicherheitskritische Daten zu speichern!

   Jeder der Zugriff auf den Client hat kann die Cookies einsehen

5. Nennen Sie den Grund, weshalb Browser-Sessions (im Gegensatz zu Cookies) für sicherheitskritische Anwendungen besser geeignet sind. Nennen Sie drei Einsatzgebiete für Browser-Sessions!

   Browser-Sessions beinhalten lediglich einen zufälligen kryptographischen Wert, mit welchem auf serverseite der Client identifiziert werden kann, und in welcher die Session Variablen auf dem Server existieren

6. Nennen Sie drei Einsatzgebiete für Browser-Sessions!

   Webshop
   Buchungssystem
   Lernmanagmentsystem

2 Frontend-Entwicklung mit JavaScript und HTML5

Für die Programmierung im Frontend gibt es diverse Möglichkeiten.

2.1 Frontend-Programme im Browser

Ursprünglich war Java die bevorzugte Sprache für die plattformübergreifende Programmierung. Damit kann man sog. Applets erstellen, die in eine HTML-Seite eingebettet werden. Der Browser selbst kann keine Java-Applets ausführen, sondern benötigt hierzu eine externe Java-Umgebung. Java-Applets sind sehr mächtig, aber grafische Benutzerschnittstellen, die in Java erstellt werden, fühlen sich oftmals etwas träge an und fügen sich optisch nicht immer gut in die grafische Oberfläche des eigenen Betriebssystems ein.

Adobe Flash ist eine Technik, die zwar intern ganz anders aufgebaut ist als Java, aber ebenfalls nicht direkt im Browser läuft, sondern ein externes Plug-in benötigt. Beide Plug-ins, sowohl das für Java-Applets wie auch der Flashplayer, waren in den letzten Jahren immer wieder Ziel von Schadsoftware. Leider werden laufend neue Sicherheitsprobleme bekannt. Beide Technologien sind nicht für den Einsatz in Smartphones geeignet, da Java-Applets relativ viel Arbeitsspeicher benötigen und insbesondere Flash für ein Gerät mit Batteriebetrieb zu hohe Anforderungen an die Prozessorleistung stellt.

Aus diesen Gründen haben Java-Applets für die Frontendprogrammierung keine Bedeutung mehr. Dies gilt auch für Flash, welches zwar noch anzutreffen ist, aber immer mehr durch HTML5 und JavaScript abgelöst wird.

2.2 Was ist JavaScript?

JavaScript ist eine interpretierende Programmiersprache, die sich syntaktisch an C anlehnt, und wurde Mitte der 1990er-Jahre von Netscape entwickelt. JavaScript hat übrigens nichts mit der Programmiersprache Java zu tun. Der Name für die Skriptsprache wurde von Marketingexperten gewählt, weil zu jener Zeit die Sprache Java stark an Bedeutung zunahm und Netscape mit der Namensähnlichkeit vom Java-Boom profitieren wollte. Seit jener Zeit hat sich die Sprache stark entwickelt. Ein besonderes Merkmal ist die Typen-Freiheit, d.h. Variablen benötigen keinen expliziten Datentyp und können immer mit beliebigen Daten befüllt werden. Zudem kann mit JavaScript objektorientiert programmiert werden. Jeder Webbrowser kann heute problemlos mit JavaScript direkt umgehen, es sind keine Plug-ins notwendig. Dadurch können JavaScript-Programme auch auf schwächeren Geräten gut ausgeführt werden. Dies ist besonders bei Smartphones im Batteriebetrieb ein grosser Vorteil.

JavaScript wird als Quelltext in HTML eingebettet und direkt im Browser ausgeführt. So hat ein JavaScript-Programm Zugriff auf den ganzen HTML-Objektbaum und kann diesen dynamisch auslesen und verändern. Seit einigen Jahren bereits stellen alle Browser JavaScript-Objekte zur Verfügung, über die ein JavaScript-Programm unabhängig von einer Benutzerinteraktion Netzwerkverbindungen zu einem Server aufbauen kann. Für den Datenaustausch mit dem Server wird meist das Protokoll und seit einiger Zeit JSON verwendet.

2.3 Was ist AJAX?

AJAX ist nicht nur ein Putzmittel oder ein Fussballklub, sondern vor allem ein Marketingbegriff in der Webtechnologie. Es ist die Abkürzung von engl. Asynchronous JavaScript and XML; asynchron, weil die Netzwerkkommunikation unabhängig von einer Benutzerinteraktion

erfolgen kann. Er fasst eigentlich nur die verschiedenen Technologien zusammen, die heute bei clientseitigen Web-Anwendungen zum Einsatz kommen.

2.4 Wo liegen die Vorteile von HTML5 und CSS3?

HTML5 und CSS3 sind die jeweils neuesten Versionen von HTML und CSS. Beide haben starke neue Möglichkeiten erhalten. So können z.B. mit CSS3 komplexe Animationen und Transformationen erstellt werden, ohne dass auf Flash oder ein Java-Applet zurückgegriffen werden muss. HTML wurde, vor allem gegenüber XHTML, vereinfacht. Zudem ist die Integration von HTML, CSS und JavaScript stark vorangetrieben worden. Damit stehen einem Entwickler faszinierende neue Möglichkeiten zur Gestaltung von Benutzeroberflächen im Browser zur Verfügung. Alle modernen Browser haben integrierte Entwicklerwerkzeuge. Damit lassen sich DOM-Strukturen direkt in der angezeigten Seite manipulieren und CSS-Angaben können verändert werden. So kann das Aussehen einer Seite interaktiv getestet werden. Für JavaScript bringen die Browser Kommandokonsolen und Debugger mit.

2.5 Entwicklungsumgebungen zur Programmierung

Für die Programmierung der Serversoftware und des Clients stehen verschiedene Entwicklungsumgebungen zur Verfügung. Auf dem Server wird in grossen Unternehmen oft die Sprache Java eingesetzt. Im privaten Bereich und für kleinere und mittelgrosse Projekte bietet sich PHP an. Clientprogramme werden heute oft kombiniert mit JavaScript erstellt. JavaScript und die modernen Versionen HTML5 und CSS3 arbeiten eng zusammen und erlauben es, ganze interaktive Anwendungen im Browser zu programmieren. Bei den Entwicklungsumgebungen unterscheidet man zwischen integrierten Umgebungen IDE und spezialisierten Texteditoren.

Eine IDE umfasst nebst einem komfortablen Texteditor zum Schreiben der Programme auch Werkzeuge zur Fehlersuche und oftmals auch gleich eine integrierte Laufzeitumgebung zum Testen der Anwendung. Sowohl für die Server- wie auch für die Clientprogrammierung stehen Frameworks zur Verfügung, die einerseits den Umgang mit der entsprechenden Programmiersprache und Laufzeitumgebung vereinfachen, andererseits stellen solche Frameworks auch ganze Funktionalitäten zur Verfügung, um die man sich dann nicht mehr selbst kümmern muss.

Wir arbeiten hier im Modul M133 mit der Entwicklungsumgebung «Microsoft Visual Studio Code» und der XAMPP Umgebung.

Kontrollfragen

1. Was meint man damit, wenn man von der Philosophie eines Frameworks spricht?

   Die theorethischen Konzepte welche ein frontend Framework verwendet
   und dessen implementation im technischen Sinne.

   Im Allg. was das zu lösende Ziel eines Frameworks ist, ein Framwork dien immer einem
   bestimmten Problem, um dessen Ausführung zu vereinfachen.

2. Was ist eine IDE?

   Integrated Developer Environment
   Entwicklung von Software

3. Nennen Sie drei JavaScript Frameworks. Wozu setzen Sie diese ein?

   Angular: Entwicklung von grossen Geschäftsapplikationen

   React: Etnwicklung von kleineren aber immer noch modularen Applikationen

   NextJs: Entwikclung von statischen Websiten, anhand von React code

4. Warum sollte man heute keine Client-Anwendungen mehr mit JavaApplets oder Flash erstellen?

   Da dies die modernen Browser nicht mehr unterstützen und die Welt sich
   auf Javascript Framworks fokussierte.

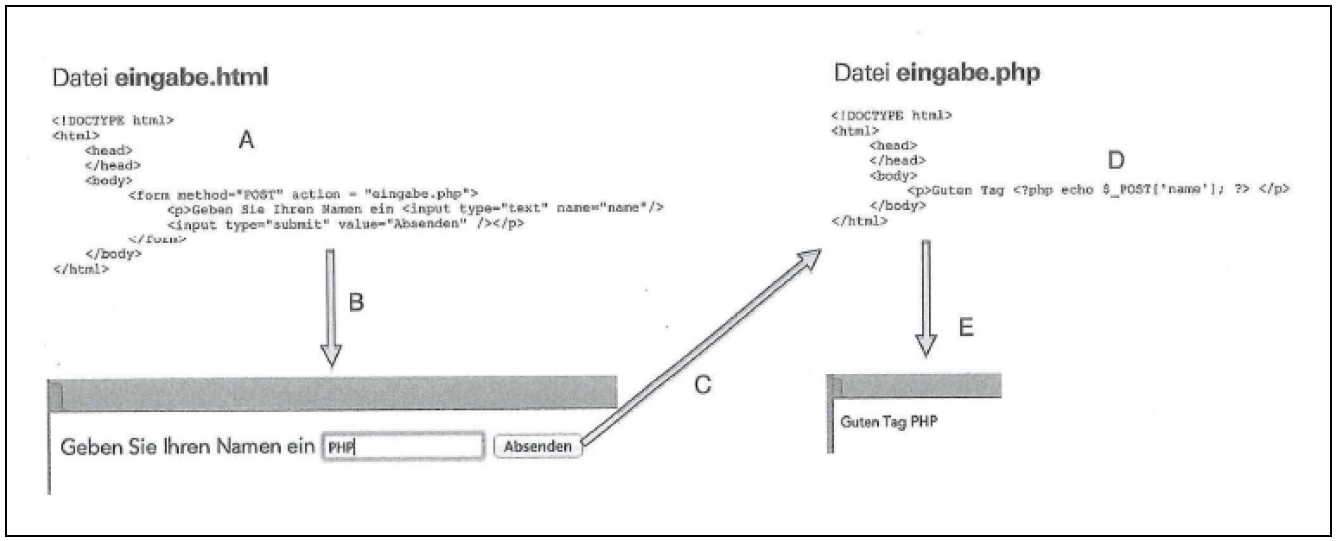
3 Benutzereingaben und Datenüberprüfung

Die Interaktion mit dem Benutzer und dort vor allem die Dateneingabe ist ein wichtiger Punkt bei einer Anwendung. Die Dateneingabe findet im Browser, also im Client statt, die Datenaufbereitung und -speicherung passiert in der Serverapplikation (auf dem Server).

3.1 Zusammenspiel Frontend / Backend

Betrachten wir anhand eines einfachen Beispiels, was passiert, wenn ein Benutzer eine Eingabe macht.

Zusammenspiel Client / Server



Auf dem Server befindet sich die Datei **eingabe.html** (A). Diese wird vom Webserver an den Browser ausgeliefert, wo deren Inhalt **interpretiert und dargestellt** wird (B). In diesem Beispiel enthält die Datei ein Formular mit einem Textfeld und einer Absenden-Schaltfläche. Wenn der Benutzer im Textfeld eine Eingabe macht und danach den Knopf aktiviert, verpackt der Browser die Eingabedaten, hier den Text «PHP» aus dem Textfeld, und schickt diese an die Seite **eingabe.php** (C). Dieses Ziel ist beim Formular als **action="eingabe.php"** notiert.

Auf dem Server passiert nun Folgendes: Die angeforderte Datei hat als Dateinamen-Erweiterung nicht **html** sondern **php**. Für den Webserver ist dies das Zeichen, dass die Datei vor dem Ausliefern an den **PHP-Prozessor** übergeben werden muss. Dieser sucht sich die **PHP-Befehle** (D) heraus und führt diese aus. Hier versucht er, den übergebenen Wert «name» auszulesen und schreibt ihn dann in die Datei. Jetzt gibt der PHP-Prozessor die überarbeitete Datei an den Webserver zurück. Dieser schickt sie als Antwort an den **Browser** (E), wo sie dargestellt wird.

3.2 POST- und GET-Anfragen an den Server

Formulardaten können auf zwei unterschiedliche Weisen an den Server übermittelt werden. Sicher haben Sie schon URLs gesehen, die «?» und «&» enthalten.

```
http://www.meinserver.ch/benutzer?name=Meier&vorname=Peter
```

Hier werden die Werte für Name und Vorname an den Server übergeben. Die Werte stehen dabei als **Schlüssel-Wert-Paare** direkt in der URL, abgetrennt durch ein «?» und untereinander getrennt durch ein «&». Dies macht der Browser automatisch, wenn man im Formular **method="GET"** einträgt. Der Vorteil einer solchen URL besteht darin, dass Sie diese auch in eine Lesezeichenliste aufnehmen oder z.B. per E-Mail versenden können. GET-URLs werden hauptsächlich für **Datenabfragen** verwendet; die Werte in der URL entsprechen den **Suchvorgaben**. Da eine URL nicht beliebig lang sein kann (einige Tausend Zeichen maximal), können Sie nicht beliebig grosse Datenmengen übermitteln. Wenn Sie im Formular **method=" POST"** setzen, wird der Browser die Daten nicht in der URL aufführen, sondern im **Inneren der Anfrage** an den Server mitliefern. Die Informationen sind dann von aussen nicht direkt sichtbar und dadurch auch besser vor (böswilligen) Manipulationen geschützt. Bei einer POST-Übermittlung gibt es **keine Grössenbeschränkung**; so können auch grosse Datenmengen (z.B. Videodateien) übertragen werden.

3.3 Formulare und Formularelemente

Der Benutzer fordert vom Server eine HTML-Datei an, die oftmals **Eingabeelemente** hat. Nach dem Eingeben der Daten kann der Benutzer diese mittels eines Knopfs absenden. Der Browser übermittelt die Daten entweder mit der URL an den Server (bei einer GET-Anfrage) oder verpackt sie innerhalb der Anfrage selbst (bei einer POST-Anfrage). Das Programm auf dem Server hat Zugriff auf die übertragenen Daten und kann diese weiterverarbeiten. Sowohl der Client wie auch der Server müssen sicherstellen, dass keine ungültigen Daten übertragen werden. Auf dem Client kann das gut mit JavaScript gemacht werden.

Um vom Benutzer Eingaben entgegenzunehmen, stehen verschiedene **Eingabeelemente** zur Verfügung. Alle Elemente müssen innerhalb eines Formulars angeordnet werden.

Unter http://www.w3schools.com/tags/default.asp finden sich alle HTML-Tags mit guten Beispielen. Die wichtigsten **Eingabeelemente** finden Sie im nächsten Abschnitt.

3.4 HTML-Eingabeobjekte

-

```
<input type="text" name="name" id="name" />
```

-

```
<textarea name="beschreibung" id=" beschreibung "></textarea>
```

-

```
<select>
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="mercedes">Mercedes</option>
  <option value="audi">Audi</option>
</select>
```

-

```
<input type="checkbox" name="red" value="RED">
```

-

```
<input type="radio" name="red" value="RED">
```

-

```
<input type="submit" value="submit" id="submit" />
```

Kontrollfragen

1. Was bedeuten die an die URL angehängten Werte?

   ```
   http://www.Server.ch/suche?marke=VW&modell=Golf
   ```

   Dies sind GET-Parameter, in PHP mit $_GET verfügbar

2. Wenn Sie im Browser Eingaben validieren wollen, welche Programmiertechnik werden Sie vermutlich verwenden?

   HTML: Feldervalidierung mit Auswahlt richtiger Input type (Email, Telefonnummer etc)

   Javscript: für erweiterte Validierung von unüblichen Inputs

3. Warum ist es nicht empfehlenswert, längere Texte per URL zu übergeben? Beschreiben Sie den Grund in einem kurzen Satz!

   Urls haben ein Limit, je nach Browser anders...

4. Wie sieht die URL aus, wenn Sie die beiden Werte «Hans» und «28» als Name und Alter auf der Seite **adresse.php** haben möchten? Notieren Sie die URL, wie Sie sie beim Browser eingeben würden!

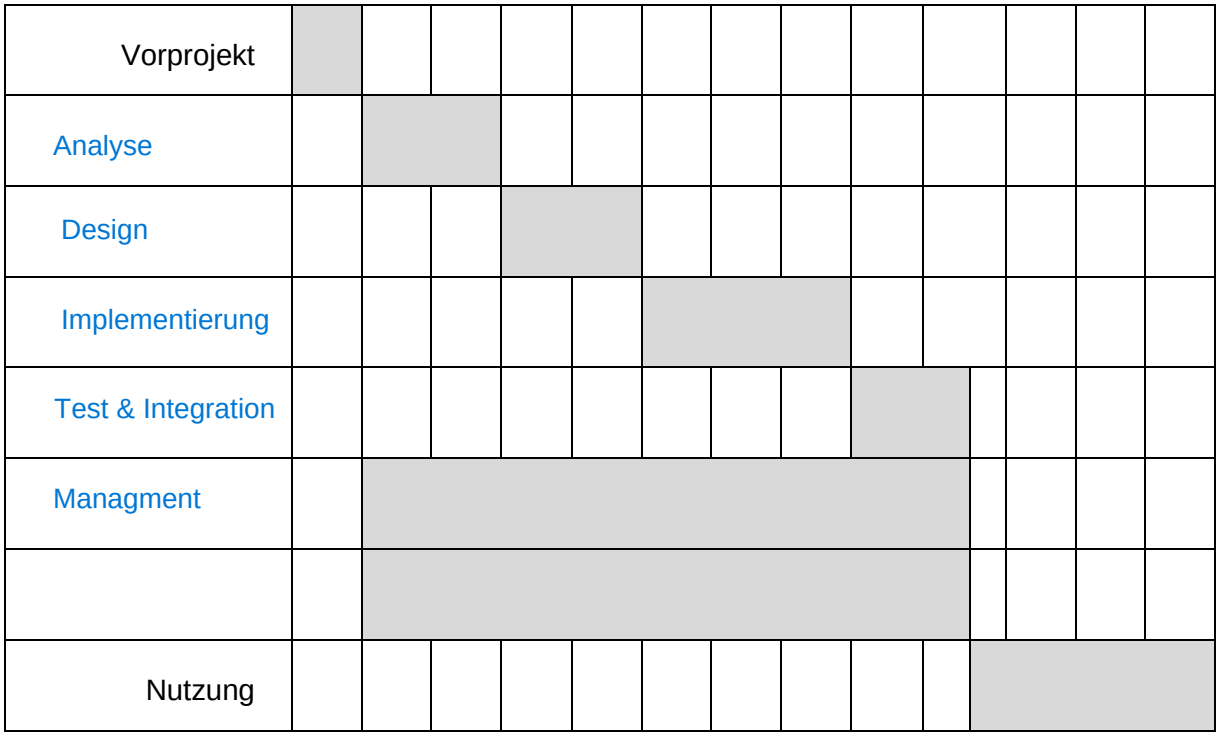
   host/?name=Hans&alter=28

4 Analyse und Entwurf

Bevor wir mit der Realisierung beginnen können, müssen wir zuerst einmal wissen, was denn unsere Applikation alles können muss. Wir müssen **Vorgaben analysieren** und die **Anforderungen spezifizieren**. Anhand eines einfachen Projekts wollen wir den Ablauf genau betrachten und uns ein Vorgehensmodell erarbeiten.

Der Kunde wünscht, dass interne Firmenmitteilungen von verschiedenen Personen erfasst und auf der internen Homepage angezeigt werden können. Das Projekt läuft unter dem Namen **Verwaltungs-News**.

Wie Sie schon wissen, unterteilen wir ein Projekt in sogenannte **Projektphasen**, die da sind:



Sie erkennen, dass **Analyse- und Entwurfsphase mit rund 40%** einen grossen Teil am gesamten Projekt ausmachen. Es sind dies ganz wichtige Phasen und hier muss genau gearbeitet werden, damit nichts unberücksichtigt bleibt.

4.1 Funktionale Anforderungen

Für die Programmierung der Applikation ist folgende Frage von **zentraler Bedeutung**:

> «Was soll diese Applikation können beziehungsweise
> welche Funktionen soll sie erfüllen?»

Diese Frage können Sie direkt anhand der Anforderungen beantworten. Die Anforderungen des Projekts **Verwaltungs-News** lauten wie folgt:

> «Um die Interaktion und Bindung zum System zu verbessern, sollen auch interne Neuigkeiten online dargeboten werden, wobei vorgesehen ist, dass die jeweiligen Abteilungsleiter diese Neuigkeiten unter ihrem Namen publizieren können. Diese Neuigkeiten sollen eine bestimmte Zeit lang sichtbar sein und dann verschwinden. Ausserdem sollen einfache Hilfsautoren eingesetzt werden. Dies alles soll ohne fremde Hilfe geschehen können»

Lesen Sie die relevanten Informationen heraus und interpretieren Sie sie:

Information	Interpretation
Interne Neuigkeiten online publizieren	Es geht also darum, im Internet News-Items zu platzieren
Bestimmte Zeit lang sichtbar	Wegen der zeitlichen Beschränkung wird eine dynamische Lösung bevorzugt
Abteilungsleiter wollen unter ihren eigenen Namen publizieren	Ein Log-in-System erscheint angemessen, das auf Benutzerkonten basiert
Hilfsautoren sollen eingesetzt werden	Es sollen mehrere Benutzer angelegt werden können
Ohne fremde Hilfe	Gefordert wird eine möglichst einfache Bedienung und Eingabe

Aus diesen Interpretationen kristallisiert sich folgender **Funktionalitätskatalog** heraus:

1. _____
2. _____
3. _____
4. _____

4.2 Zielpublikum

Wichtig ist immer auch die Frage: An wen richtet sich die Anwendung? Wer soll damit arbeiten? Man spricht vom **Zielpublikum**. Aus den Anforderungen geht hervor, dass allgemein Mitarbeitende angesprochen werden sollen, diese lesen die Nachrichten. Aber die Applikation soll auch von Abteilungsleitern verwendet werden. Diese sollen auf einfache Art und Weise neue Nachrichten erfassen können. Wir unterscheiden also zwei Gruppen von Personen, die mit der Anwendung in Kontakt kommen. Für beide Gruppen soll sich die Applikation unterschiedlich verhalten. Dies muss bei der **Realisierung** berücksichtigt werden. Vor allem muss der Administrationsteil dort, wo neue Meldungen erfasst werden, einfach gehalten

werden, da Abteilungsleiter keine Informatiker sind. Aber es gibt noch eine weitere wichtige Unterscheidung: Sie müssen sich fragen, von wo aus die Benutzer mit der Applikation arbeiten. Sind es Benutzer, die vom internen Firmennetz aus zugreifen oder sollen Benutzer auch von ausserhalb Zugriff haben? Und was genau heisst hier «aussen»?

4.2.1 Intranet

Ein Intranet ist ein Computernetzwerk, das mit den gleichen Protokollen arbeitet wie das Internet, aber gegenüber dem Internet abgeschottet und von dort aus nicht erreichbar ist. Ein LAN ist normalerweise ein Bestandteil des Intranets. Die Absicherung gegenüber dem Internet erfolgt durch spezielle Netzwerkkomponenten, so genannte Firewalls. Eine Firewall sitzt an der Schnittstelle zweier Netzwerke und überwacht den Netzwerkverkehr. Sie lässt normalerweise Netzwerkverbindungen nur in eine Richtung (von innen nach aussen) zu und kann auch solche Verbindungen gezielt kontrollieren und regeln. Das Intranet ist also ein privates Netzwerk.

4.2.2 Was ist ein Extranet?

Als Extranet bezeichnet man den Teil eines Intranets, der für bestimmte Benutzer von aussen her – also vom Internet her – erreichbar ist. Das Extranet ist «halb-privat». Der Zugang muss normalerweise auf die eine oder andere Art beantragt werden. Das System muss sicherstellen, dass nur Berechtigte Zugriff haben. Ein typischer Fall von Extranet stellt zum Beispiel angemeldeten Kunden einen Zugang zu Supportdokumenten bereit.

Bei der Bestimmung des Zielpublikums muss eine Unterscheidung Intranet, Extranet und Internet – also «privat», «halb-privat» oder «öffentlich» – gemacht werden. Je nachdem stellen sich vor allem bezüglich Sicherheit zusätzliche Anforderungen.

Im vorliegenden Fall sind beide Benutzergruppen innerhalb des lokalen Intranets zu Hause.

4.3 Nichtfunktionale Anforderungen

Neben den sog. funktionalen Anforderungen gibt es auch nichtfunktionale Anforderungen. Die funktionalen Anforderungen beschreiben, was die Applikation alles können soll, die nichtfunktionalen Anforderungen ergeben sich aus Fragen nach dem «Wie erledigt die Applikation ihre Aufgabe?». Dies sind Fragen nach der Performance, der Barrierefreiheit oder auch der Sicherheit. Auch wenn eine funktionale Anforderung an eine Applikation erfüllt ist (neue Nachrichten können erfasst werden), kann es sein, dass die Bedienung hierbei so kompliziert ist, dass ein Benutzer damit nicht zurechtkommt. Die nichtfunktionale Anforderung «einfache Bedienbarkeit» ist dann nicht erfüllt.

4.3.1 Performance

Typische Anforderungen bezüglich Performance können sein:

- Seitenladezeit

- Scalability -> mehr User, mehr Servreload

-

4.3.2 Barrierefreiheit

Beachten Sie, dass nicht jeder Benutzer über beste Augen verfügt oder mit Maus und Tastatur problemlos umgehen kann. Wenn es Ihre Applikation solchen Benutzern schwer macht, spricht man von Barrieren. Schriften sollten gross genug gewählt werden, dass sie auch für Fehlsichtige lesbar sind. Blinde Personen verwenden oftmals sog. Bildschirmleser (Screenreader). Dies sind Programme, die einen Text in gesprochene Sprache umsetzen. Damit eine Webseite automatisch sinnvoll vorgelesen werden kann, muss sie einige Voraussetzungen erfüllen. Eine davon ist die strikte Trennung zwischen Struktur und Darstellung. Ein Bildschirmleser muss die Struktur der Web-Applikation erkennen und umsetzen. Alles, was mit Darstellung zu tun hat, ist beim Vorlesen nicht relevant. Wenn Sie sauber mit HTML und CSS arbeiten, sind Sie auf dem richtigen Weg. Vermeiden Sie zudem spezielle Farbkombinationen, die Menschen mit Farbblindheit nicht oder nur schlecht unterscheiden können.

Eine weitere Barriere sind kleine, nur wenige Pixel grosse Knöpfe oder andere anklickbare Bereiche. Vor allem älteren Menschen oder solche mit einer Bewegungsstörung fällt es oftmals schwer, mit der Maus genau zu zielen. Machen Sie deshalb Knöpfe und andere Bedienelemente gross genug, versuchen Sie auch alternative Bedienungsmöglichkeiten zu schaffen.

4.3.3 Datenintegrität

Wenn ein Benutzer Daten eingibt, erwartet er, dass diese Daten richtig und vollständig vom System übernommen werden und auch wieder so abgerufen werden können. Dies muss das System sicherstellen, auch wenn zum Beispiel während eines Speichervorgangs ein Problem auftritt. Beispielsweise darf eine Bestellung nicht gespeichert werden ohne zugehörige Information zum Besteller (Kunde). Denken Sie an einen Benutzer mit einem Mobilgerät im Zug: Es besteht immer die Möglichkeit, dass im ungünstigsten Moment die Netzwerkverbindung abbricht und so nur ein Teil der Daten übermittelt werden.

4.3.4 Verfügbarkeit

Wenn wir von Verfügbarkeit sprechen, müssen wir Antworten auf die folgenden Fragen finden:

- Wann soll die Web-Applikation erreichbar sein? 24 Stunden pro Tag? Ununterbrochen das ganze Jahr? Oder reicht es, wenn die Applikation während der Bürozeiten erreichbar ist? Vergessen Sie dabei nicht Ihre Filiale in Australien. Die Schweizer Bürozeiten sind nicht deckungsgleich mit denen auf der anderen Seite der Erdkugel
- Sind Zeiten erlaubt, in denen die Applikation nicht erreichbar sein muss? Wie sollen Sicherheitskopien der Daten erstellt werden? Darf hierzu der Zugriff auf die Datenbank gesperrt werden oder muss die Sicherung parallel zum normalen Betrieb erfol-

gen? Die Antworten auf diese Fragen haben Einfluss darauf, welche Produkte zu Datenhaltung und Backup eingesetzt werden. Nicht jedes Datenbanksystem kann im laufenden Betrieb gesichert werden

- Welche Konsequenzen hat es, wenn der Server durch einen technischen Defekt (zum Beispiel eine defekte Festplatte) ausfällt? Wie lange darf so ein Ausfall dauern?

4.3.5 Authentizität

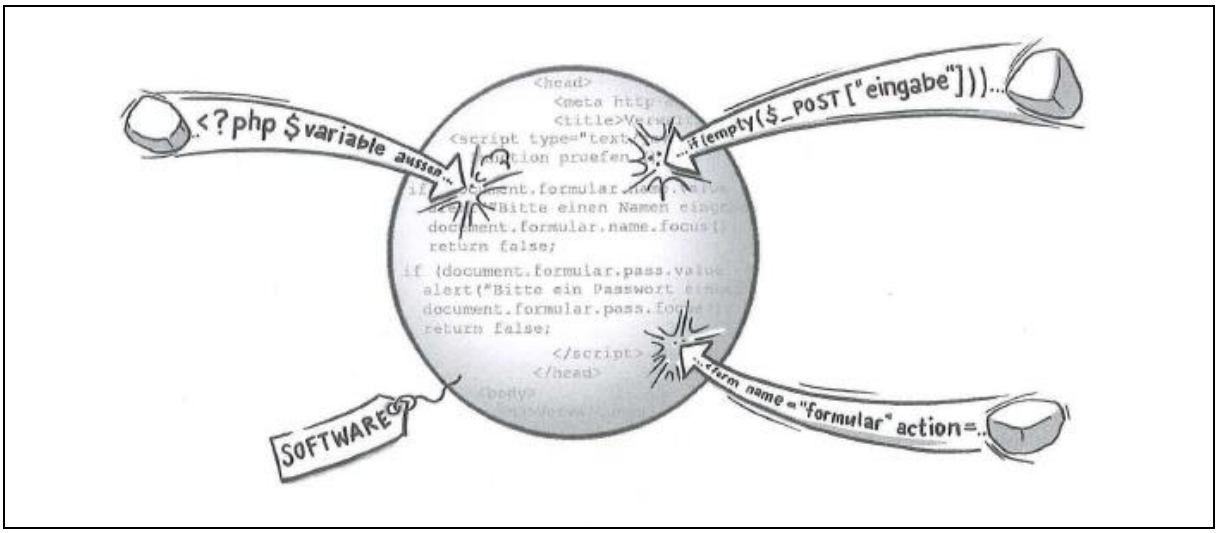
Sicher haben Sie auch schon E-Mails erhalten, worin Sie aufgefordert werden, einen Link anzuklicken, um auf Ihrer Bankhomepage Ihr Konto zu bestätigen. Die Internetseite, auf die der Link verweist, sieht genau gleich aus wie die Homepage Ihrer Bank – also ist doch alles in Ordnung oder nicht? Sie als Benutzer müssen überprüfen können, dass Sie wirklich bei Ihrer Bank gelandet sind und nicht auf einer nachgemachten Seite, deren Ziel es ist, an die Zugangsdaten für Ihr Konto heranzukommen. Als Betreiber einer Web-Applikation sind Sie auf der anderen Seite ebenfalls daran interessiert, ob «Herr Meier» auch wirklich Herr Meier ist, der soeben eine Bestellung in Ihrem Webshop getätigt hat.

4.3.6 Vertraulichkeit

Stellen Sie sich vor, Ihre Web-Applikation bietet den Benutzern die Möglichkeit, für einen einfachen Einkauf in der Zukunft die kompletten Benutzerdaten inklusive Kreditkartennummern zu hinterlegen. Dazu muss der Benutzer sich mit einem Benutzernamen und einem Passwort anmelden. Jeder Benutzer wird davon ausgehen, dass solch heikle Informationen von Ihnen vertraulich gehandhabt werden. Schon die Übertragung der Kreditdaten und des Passworts darf nicht von unberechtigten Personen abgefangen werden. Wie gehen Sie mit solchen Herausforderungen um? Kann Ihre Web-Applikation diese Vertraulichkeit garantieren? Wie speichern Sie die Passwörter und Kreditkartennummern? Kann da jemand unberechtigterweise darauf zugreifen? Das sind ganz wichtige Fragen. Natürlich sind nicht alle Daten so heikel wie Kreditkarteninformationen. Für welche Informationen sind höhere Ansprüche an die Vertraulichkeit notwendig? Gerade in diesem Bereich kann es gesetzliche Vorgaben geben, die Sie berücksichtigen müssen.

4.4 Sicherheitsaspekte

Würden Web-Applikationen für sich alleine stehen, wäre die **Anforderung der Sicherheit** bereits erfüllt, wenn sie fehlerlos funktionieren. Da sie typischerweise aber mit Besuchern interagieren und für diese auch erreichbar sein müssen, kommt eine neue Dimension hinzu: Web-Applikationen müssen auch alle Eingaben von Benutzern bestehen können. Unser Augenmerk richten wir also hauptsächlich auf die Stelle einer Applikation, wo Daten von der «Aussenwelt» ins Innere des Skripts gelangen können.



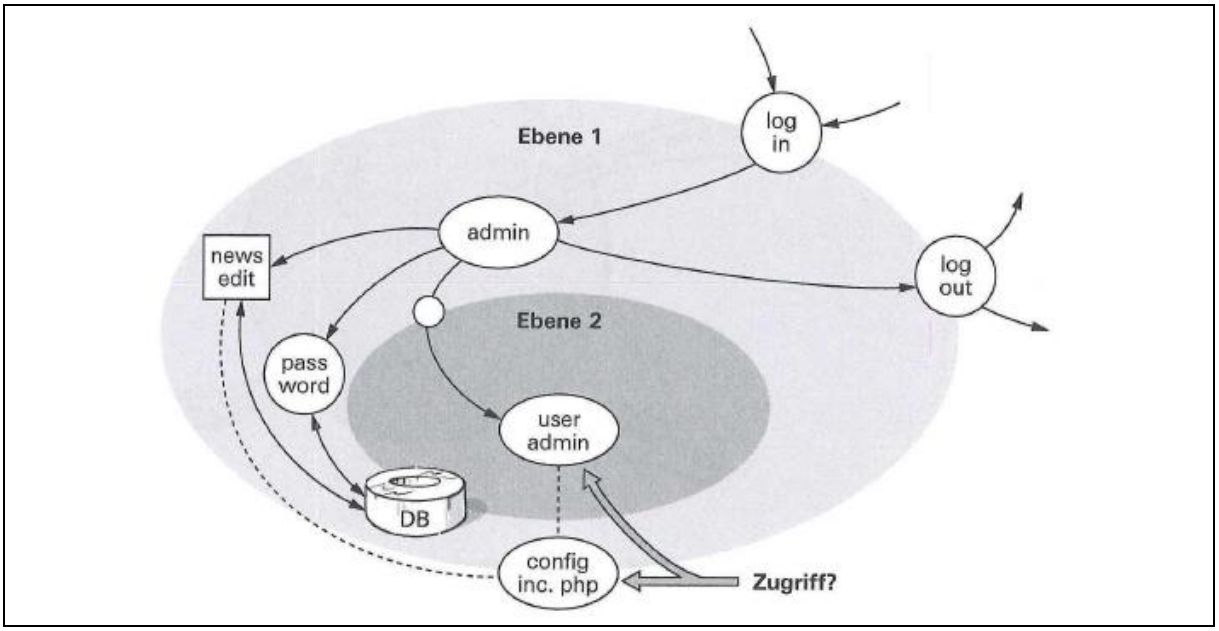
Grundsatz: Sicherheit soll möglichst nicht stören

Sicherheit ist Pflicht, und trotzdem sollten Sie dabei eine Balance zwischen Sicherheit und Benutzerfreundlichkeit suchen. Fühlt sich nämlich ein Benutzer durch allzu strenge Sicherheitsvorkehrungen eingeengt, steigt seine Motivation, diese zu umgehen, um sich das Leben zu erleichtern.

Wenn Sie beispielsweise eine sehr strenge **Passwortrichtlinie** wählen, die die Benutzer jeden Monat zwingt, ein neues Passwort zu wählen, das zudem mindestens 30 Zeichen lang ist und mindestens 5 Sonderzeichen beinhaltet, dann werden sich die Benutzer drei entsprechende Passwörter auf ein Blatt Papier schreiben und dieses an die nächste Wand heften, um jeden Monat wieder das nächste auf der Liste auszuwählen. Betritt dann einmal eine fremde Person das Büro, waren die ganzen Sicherheitsvorkehrungen des Systems umsonst.

Es gilt also, diese Aspekte bei der Planung der **Verwaltungs-News** zu berücksichtigen.

Aufgrund der erwarteten Funktionen erstellen Sie eine Übersicht der Funktionalitäten und der sicherheitsrelevanten Fragen, die dabei auftauchen:



Die einzelnen «Sicherheitszonen», also Domänen, die nur mit einer bestimmten Zugriffsstufe erreicht werden können, sind durch Flächen gekennzeichnet. Die Benutzerpfade, die ein Benutzer normalerweise verwendet, sind durch Pfeile markiert.

4.5 Zusammengefasst

- In der **Analysephase** informieren Sie sich beim Zielpublikum (Auftraggeber) über Ihre genaue Aufgabe. Dies erreichen Sie beispielsweise zuverlässig mithilfe der «W-Fragen» (was, wann, wie? usw.). Anhand dieser Fragen ermitteln Sie, welches die genauen Anforderungen an die Web-Applikation sind und wie Sie diese erfüllen können.
- Sie unterscheiden zwischen **funktionalen** und **nichtfunktionalen Anforderungen**. Bei den nichtfunktionalen Anforderungen müssen Sie Fragen zu Performance, Barrierefreiheit, Datenintegrität, Verfügbarkeit, Authentizität und Vertraulichkeit stellen.
- Um bereits in dieser Phase die **Sicherheitsaspekte** zu berücksichtigen, spielen Sie in Gedanken verschiedene Benutzerszenarios durch, um möglichen Schwachpunkten auf die Spur zu kommen.
- Ein wichtiger Punkt ist die **Eingabevalidierung**. Diese muss in jedem Fall auf dem Server gemacht werden, um ungültige Daten zu vermeiden, jedoch ist es für den Benutzer angenehm, wenn Sie die Validierung zusätzlich (nicht anstatt!) auch auf dem Client machen.

Kontrollfragen

1. Mit welcher Technik können auf einfache Weise die Fakten, die für das Projekt wichtig sind, ermittelt werden?

   Mit W-Fragen Prinzip

   unten mehr eine Phase
   [Anforderunsanalyse (Pflichten, Lastenheft)]

2. Was ist der Unterschied zwischen Vertraulichkeit und Authentizität?

   Ist ein nutzer vertrauenswürdig (kein Bot)

   Ist ein Nutzer wer er wirklich vorgibt zu sein (Login mit Password, 2FA)

3. Warum müssen Sie Eingaben zweimal validieren, einmal auf dem Client und einmal auf dem Server?

   Client: damit der Endbenutzer sieht wo er Fehler gemacht hat, und weniger SErver-Load generiert wird

   Server: da nicht alle Nutzer die Clientseitige Anwendung benutzern (M2M Schnittstelle, böse Nutzer welche direkte Anfragen senden...)

5 Funktionalität entwerfen und Umsetzung planen

Nachdem klar ist, was die Aufgaben der neuen Web-Applikation sein sollen, gehen wir nun daran, die Umsetzung zu planen.

5.1 Aufteilung Frontend / Backend

Wir haben gesehen, dass eine Web-Applikation aus dem Serverteil (Backend) und dem Clientteil (Frontend) besteht. In beiden Teilen können wir Funktionalität realisieren. Jetzt müssen wir entscheiden, was auf dem Server programmiert wird und was auf dem Client gemacht werden soll. Folgende Punkte müssen dabei bedenkt werden:

- Bei der Programmierung auf dem Server stehen der Applikation normalerweise sehr viel mehr Ressourcen (Arbeitsspeicher, CPU-Leistung) zur Verfügung als zum Beispiel auf einem Smartphone. Realisieren Sie deshalb alle **performance**- und **ressource-kritischen Funktionen** auf dem Server
- Alle **sicherheitskritischen Funktionen** gehören ebenfalls ins Backend. Kein Benutzer hat dort die Möglichkeit, auf Ihren Code zuzugreifen und in die Verarbeitung einzugreifen. Auf dem Server ist Ihr Code sehr viel besser vor unberechtigtem Zugriff geschützt
- Alle **Datenvalidierungen** müssen auf dem Server gemacht werden. Nur dann haben Sie die Sicherheit, dass Ihnen keine (böswillig) manipulierten Daten untergeschoben werden
- Machen Sie **Eingabevalidierungen** zusätzlich im Frontend, um dem Benutzer eine möglichst angenehme Bedienung zu bieten
- Die **Benutzerschnittstelle** sollte nah beim Benutzer realisiert werden. Programmieren Sie diese so, dass der Code auf dem Client läuft. Sie können im Client fast beliebig aufwendige Benutzerschnittstellen mit hoher Performanz und Bedienbarkeit realisieren

5.2 Datenmodell (ERD/ERM) und Datenbank

Wenn Ihre Web-Applikation mit einer Datenbank arbeitet, müssen Sie selbstverständlich auch die passende Datenbank respektive die zugehörigen Tabellen entwerfen. Dies stellen Sie in einem ERD/ERM dar, einer Darstellungsart, die Verknüpfungen zwischen den Tabellen mit Symbolen sichtbar macht. Die Datenbank unseres **News Client** ist relativ simpel. Sie besteht aus einer einzigen Tabelle:

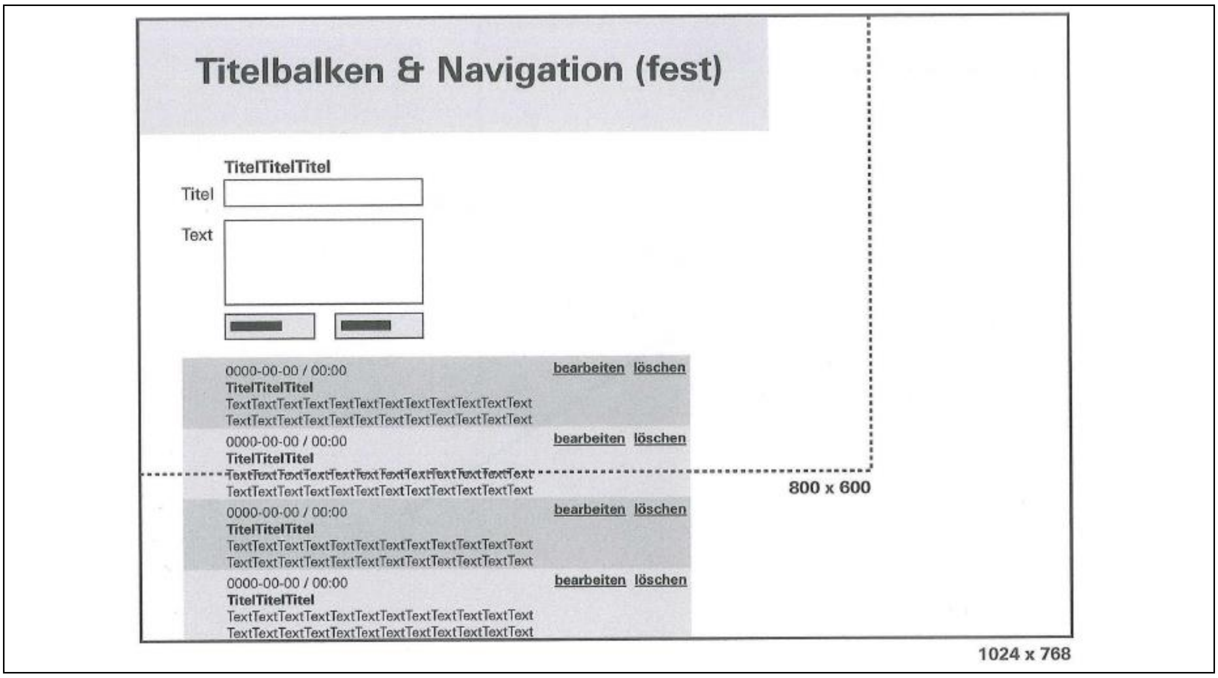
News Client	
id	Int
datum	Date
zeit	Time
titel	Varchar(255)
beschreibung	Text

5.3 Benutzerschnittstelle

Ebenfalls in die Entwurfsphase gehört die Erstellung des **optischen Erscheinungsbilds** (Layout). Dabei geht es weniger darum, nette Hintergrundbilder mit der gewählten Schriftart abzugleichen, sondern vielmehr das Erscheinungsbild **funktional** zu planen, damit der Benutzer optimal damit arbeiten kann.

In unserem Fall wird die Applikation ein Teil einer Webseite werden, welcher andere Mitarbeiter gestalten. Folglich soll das Design dieser Vorlage folgen. Sie können also nur noch die Anordnung der Elemente auf der **News-Bearbeitungsseite** wählen.

Webseiten-Layout



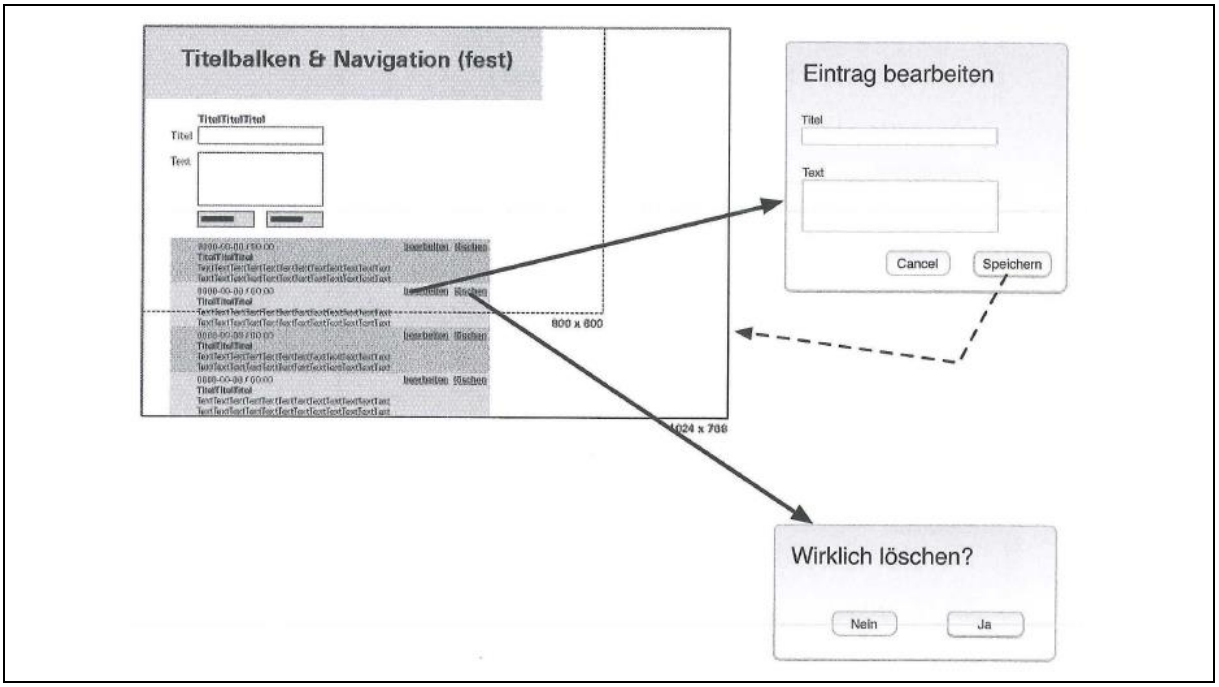
Zum Layout gehört auch eine Beschreibung der Funktionalität. Was «erlebt» der Benutzer auf der Seite? Wie ist die Seite zu bedienen? Grafische Darstellung und Beschreibung zusammen werden auch als **Storyboard** bezeichnet.

In komplexeren Anwendungen bewegt sich der Benutzer zwischen verschiedenen Ansichten hin und her. Dann müssen wir natürlich für jede Ansicht respektive Webseite ein Layout entwerfen. Ebenfalls ganz wichtig ist die **Navigation**: Wie wechselt der Benutzer zur nächsten

Seite? Welche Knöpfe oder Menüs müssen vorhanden sein und wie sind die Navigationspfade?

All das können Sie gut in einer Grafik darstellen. Man nennt eine solche Darstellung **Screenflow**.

Screenflow

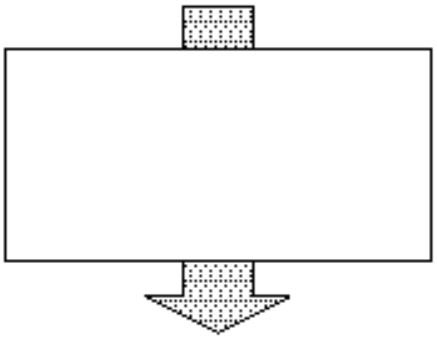


Layout, **Storyboard** und **Screenflow** sind sehr gute Werkzeuge, um mit dem Kunden und den Benutzern schon **vor der Realisierung** Bedienungsabläufe und Funktionen zu klären.

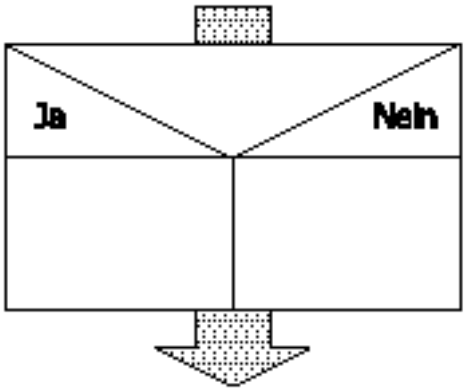
5.4 Strukturarten / Strukturblöcke / Symbole

Zur grafischen Darstellung des Lösungswegs (Algorithmus) eignet sich das **Struktogramm**, welches aus rechteckigen, in sich abgeschlossenen **Strukturblöcken** besteht. Der Name «Struktogramm» hat sich allgemein als Synonym für den Begriff **Nassi-Shneiderman-Diagramm** eingebürgert.

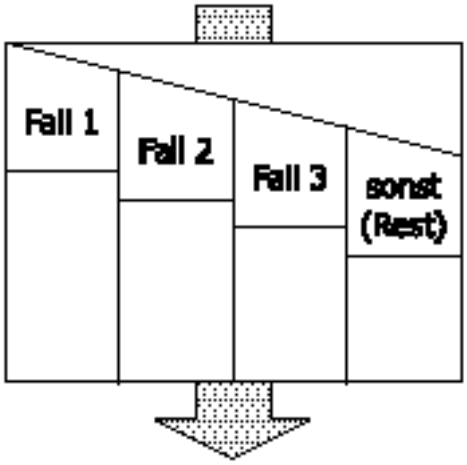
5.4.1 Einfacher Strukturblock / Sequenz (Handlung)

Ein Verarbeitungsschritt	

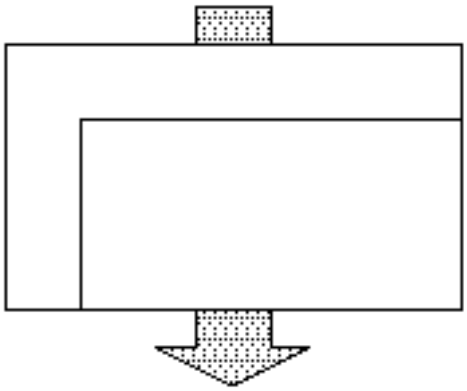
5.4.2 Verzweigungsblock (Entscheidung)

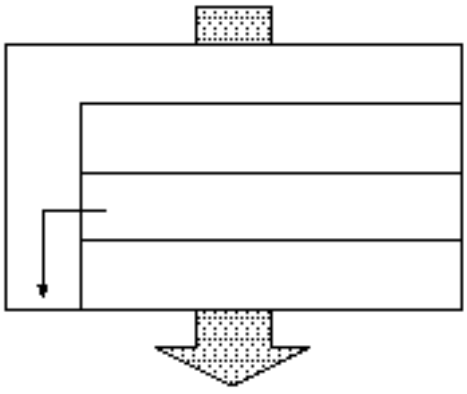
Zweifachverzweigung Aufgrund einer Entscheidung wird im Programm verzweigt. Mit dem «Ja-Entscheid» wird die Anweisung 1 und mit dem «Nein-Entscheid» die Anweisung 2 ausgeführt	

Eine Anweisung kann auch leer sein; sie wird mit folgenden Zeichen angegeben: - / ./.

Mehrfachverzweigung Sie dient zur Darstellung von Verzweigungen mit mehr als zwei Möglichkeiten (Fallunterscheidungen). Trifft keiner der Fälle zu, wird die Anweisung des Restblocks ausgeführt	

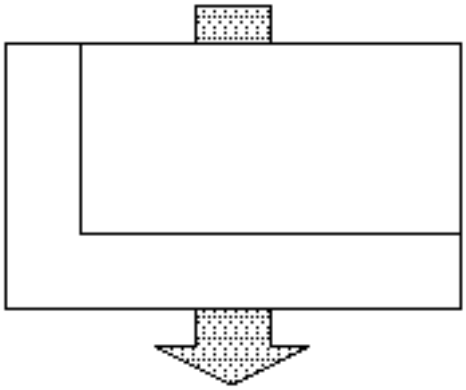
5.4.3 Wiederholungsblock (Schleife)

Mit Anfangstest Auf diese Art wird eine Schleife dargestellt. Der Rumpf wird solange durchlaufen, bis die Bedingung nicht mehr erfüllt ist. Der Bedingungstest erfolgt vor dem Eintritt in den Rumpf. Falls die Bedingung nicht erfüllt ist, wird direkt ans Blockende verzweigt	

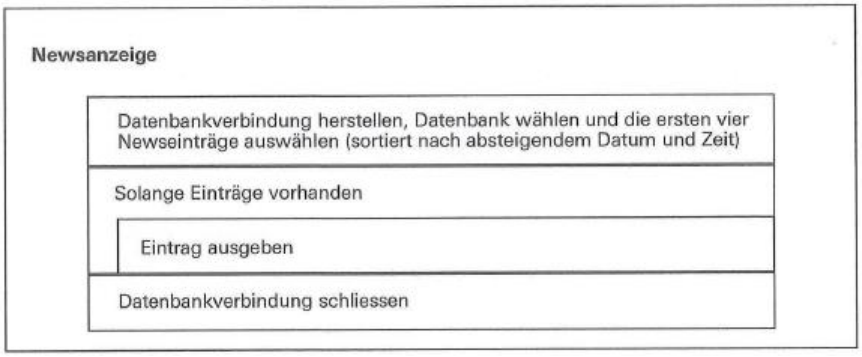
Mit Abbruchtest Die Wiederholung enthält mindestens eine Abbruchbedingung. Ist die gestellte Abbruchbedingung erfüllt, wird ans Blockende verzweigt. Andernfalls wird die Schleife wiederholt	

Mit Endtest

Der Schleifenrumpf wird mindestens einmal abgearbeitet. Ist die Wiederholbedingung nicht erfüllt, wird die Schleife erneut durchlaufen, sonst wird ans Blockende verzweigt

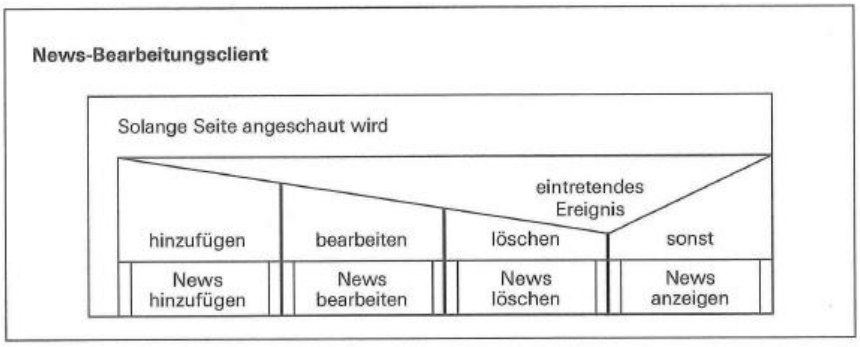


Struktogramm «Newsanzeige»

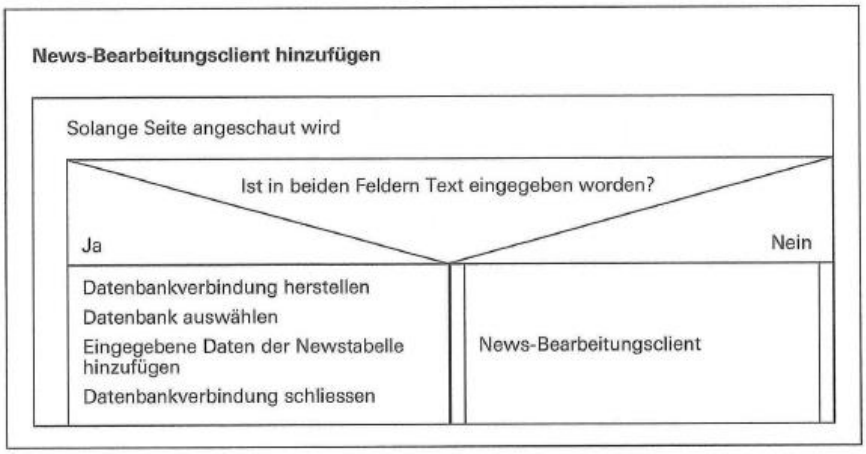


Die Entwurfsphase ist erst abgeschlossen, wenn Sie für jeden einzelnen Teil Ihrer Applikation eine grafische Darstellung (ein Struktogramm) vorweisen können. Im Projekt «News Client» fehlen also noch Struktogramme. Diejenigen nämlich des **News-Bearbeitungsclients** (Struktogramm).

Struktogramm «News-Bearbeitungsclient»



Struktogramm Unterprogramm «News hinzufügen»



In der **Dokumentation** platzieren wir nicht nur diese Struktogramme. Falls wir einen Schritt auf eine besonders spezielle Art gelöst haben, verfassen wir ebenfalls einen kurzen Erklärungstext, der den **Lösungsweg** begründet. Zum News-Bearbeitungsclient könnte er so lauten:

Dokumentation / Lösungsweg

«Der News-Bearbeitungsclient zeigt, wenn er geladen wird, alle Einträge an. Hinter jedem Eintrag findet man einen Bearbeitungs- und einen Löschlink. Da es sich bei der Routine zum Löschen eines Eintrags lediglich um zwei Zeilen Code handelt, wurde er direkt in diese Seite integriert.

*Auch auf dieser Seite findet sich **ein Formular**, mit dem man neue Newseinträge erfassen kann. Dies bietet sich an, damit man beim Erstellen eines Eintrags die älteren News immer im Auge behalten kann. Es hat sich herausgestellt, dass viele Benutzer während der Eingabe von solchen Daten den Wunsch verspüren, die bisherigen Einträge anzuschauen.*

*Beim **Absenden eines Formulars** wird – wie beim Löschen – eine Routine auf derselben Seite aufgerufen, da es dafür auch nur wenige Zeilen Code braucht.*

*Nur beim **Bearbeiten eines Eintrags** wird auf eine zweite Seite gewechselt. Dies bietet sich an, da so der Code der ersten Seite relativ einfach gehalten werden kann. Auf dieser zweiten Seite befindet sich ein ähnliches Formular wie auf der ersten Seite. Der zu bearbeitende Eintrag wird darin an-gezeigt. Nach Betätigung der Schaltfläche [Speichern] wird der Eintrag in der Datenbank aktualisiert und der Benutzer gelangt wieder zur Hauptbearbeitungsseite, in der die Einträge aufgelistet sind.»*

Diese Beschreibung sollte mit dem **Storyboard** übereinstimmen.

5.5 Realisierungskonzept erstellen

Nachdem wir die Anforderungen an unser Projekt **Verwaltungs-News** in der Analysephase erfasst haben, erarbeiten wir uns für den weiteren Projektablauf ein **Realisierungskonzept**, in dem alle relevanten Informationen des Projekts aufgeführt sind. Ziel ist es, ein **Dokument** zu erhalten, das den beteiligten Parteien als **verbindliche Abmachung** über die **Ziele des Projekts** sowie der **abgemachten Termine** dient. Uns soll es als Entwicklungshilfe dienen, da darin die zu realisierende **Funktionalität** vollständig beschrieben ist.

Üblicherweise ist das Realisierungskonzept ein **Vertragsbestandteil**, also massgebliches Dokument zur Kontrolle, ob der Auftrag im Sinne der **Auftragsstellung** erfüllt wurde:

- Allgemeine Projektangaben
- Meilensteine des Projekts (Termine)
- Anforderungen: Eigenschaften und Funktionen der Applikation
- Analyse: Ist-Situation
- Analyse: Zielgruppe(n)
- Technische Rahmenbedingungen

Zudem können wir bereits erste Ideen zu **Lösungsansätzen** und möglichen **Lösungsvarianten** notieren.

Realisierungskonzept «Verwaltungs-News»

Projekt: Verwaltungs-News	
Auftraggeber	*Name des Kunden*
Auftragserteilung	*tt.mm.jjjj*
Ausführung durch	*Vorname Name, Firma*
Realisierungskonzept	*tt.mm.jjjj*
Programmentwurf	*tt.mm.jjjj*
Quellcode	*tt.mm.jjjj*
Testplan	*tt.mm.jjjj*
Testbericht	*tt.mm.jjjj*

Aus diesen Interpretationen kristallisiert sich ein **Funktionalitätskatalog** heraus.

Funktionalitätskatalog «Verwaltungs-News» (Muster)

1 – Analyse
Ist-Situation *Intranet des Kunden ermöglicht den Mitarbeitenden, interne Informationen wie Dienst- und Ferienpläne sowie Dokumente abzurufen. Vor einigen Wochen wurden dezentrale Informationen in ein Portal zusammengeführt, bei dem nun zentral auf alle Ressourcen zugegriffen werden kann. Viele Mitarbeitende tun sich schwer mit der neuen zentralen Plattform.* ***Zielgruppe(n)*** *Zielgruppe des News-Bearbeitungsclients sind Mitarbeitende des Kunden. Diese haben eine mittlere Schulbildung mit KV-Abschluss. Alle Mitarbeitenden sind geübt im Umgang mit Computern (Word, Excel, Verwaltungssoftware). Zielgruppe des News-Outputs (auf der Startseite) sind alle Mitarbeitenden* ***Technische Rahmenbedingungen*** *XAMPP (Apache mit PHP und MySQL) unter Windows 10 GB Disk und 2 GB RAM. Clientseitig JavaScript™ und HTML5. PHP-Entwicklungsumgebung: Visual Studio Code. Textverarbeitung und Tabellenkalkulation: Word und Excel. Grafiken: Gimp*

2 – Anforderungen

Eigenschaften der Applikation
Erwartet wird Web-Applikation, mit deren Hilfe man News auf der Webseite publizieren kann. Die Eingabe neuer Einträge und die Veränderung bestehender sollen durch jeden Mitarbeitenden durchgeführt werden können.

Funktionalität allgemein
Man soll auf einer Seite (die im passwortgeschützten internen Bereich untergebracht wird) News eingeben können, die aus einem Titel und einem Text bestehen. HTML-Code soll dabei nicht verwendet werden können respektive keinen Effekt haben. Die Newseinträge sollen in einer Datenbank gespeichert werden. Weiter soll man bereits eingegebene News bearbeiten oder löschen können. Wenn man einen Newseintrag löschen will, soll zuerst noch ein Bestätigungsdialog erscheinen. Zur Übersicht sollen alle diese News auf dieser internen Seite aufgelistet werden. Auf der (öffentlichen) Webseite sollen dann jeweils die neusten vier News angezeigt werden, wobei der letzte Eintrag immer zuoberst stehen soll. Weiter soll auch angezeigt werden, an welchem Datum und zu welcher Uhrzeit der Eintrag erstellt worden ist. Es handelt sich also um eine Applikation, die aus zwei Teilen besteht, einem Bearbeitungs- und einem Anzeigemodul.

Eingaben
Vom Benutzer eingegeben werden nur Titel und Text der News. Datum und Zeit, die zur korrekten Einordnung benötigt werden, sollen von der Applikation generiert werden.

Ausgaben
Im Anzeigemodul sollen die neusten vier Newseinträge untereinander angezeigt werden, jeweils mit einem Titel (fett), hinter dem in kleiner Schrift das Datum und die Uhrzeit stehen sollen. Unter dieser Zeile soll dann gleich der Newstext folgen.

3 – Lösungsansätze

Variante 1 (News in HTML)
Abteilungsleiter senden ihre News per E-Mail an den Internetverantwortlichen, der diese mit dem HTML-Editor in die Seite einfügt

Variante 2 (<Titel>)
Gegebenenfalls Beschreibung weiterer Variante(n)

4 – Realisierung
Systemvoraussetzungen (Hardware) *Neue Hardware / Hardwarekomponenten, falls diese angeschafft werden müssen* ***Funktionalität der Software*** *News Publikationssystem auf Datenbankbasis: Möglichkeit der Erfassung, Bearbeitung und Löschung von Einträgen (Titel, Text, Zeit / Datum, Autor/in) beliebiger Länge. Aussehen soll dem Verwaltungs-Intranet anpassbar sein. Zeitlich begrenzte Anzeige der Einträge: Beiträge sollen nach einstellbarem Zeitintervall nicht mehr angezeigt werden, aber trotzdem gespeichert bleiben. Benutzerbasierte Authentifizierung für Zugang zum Bearbeitungsclient: Eine beliebige Anzahl Benutzer soll sich anmelden können, um News zu verfassen. Beim Bericht soll der Name des Benutzers erscheinen. Benutzer sollen sich mit Benutzername / Passwort anmelden (und auch wieder abmelden) können. Das Passwort soll von den Benutzern geändert werden können. Benutzerbasis einfach bearbeitbar bzw. erweiterbar: Eine Untermenge der Benutzer soll die Möglichkeit haben, Benutzer zu bearbeiten, neu zu erstellen und zu löschen.* ***Zielplattform*** *Newsanzeige: Da die Newsanzeige (auch: News-Output) in die Startseite der Webseite eingebettet wird, muss die der Webseite zugrunde liegende Richtlinie befolgt werden. Dort ist folgende Unterstützung geplant: Ab Internet Explorer™ 5, Opera™ 5, Firefox 10 systemunabhängig.* *News-Bearbeitungsclient: Da in der Verwaltung nur PCs mit Internet Explorer™ eingesetzt werden, muss grundsätzlich nur dieser unterstützt werden. Trotzdem sollen bei der Seitengestaltung so weit als möglich offizielle Standards unterstützt werden.* *Das ganze System soll auch an Bildschirmen mit einer Auflösung von nur 1024 x 768 Pixeln komfortabel bedienbar sein.* ***Zieladressen (URL)*** *Newsanzeige: Auf der Startseite des Webauftritts: http://verwaltung.firma.ch/index.php* *News-Bearbeitungsclient: Auf einer Seite des durch Verzeichnisschutz geschützten internen Bereichs: http://verwaltung.firma.ch/intern/news_admin.php* ***Zeitplan*** *Gemäss separatem Dokument*

5.6 Ressourcen planen

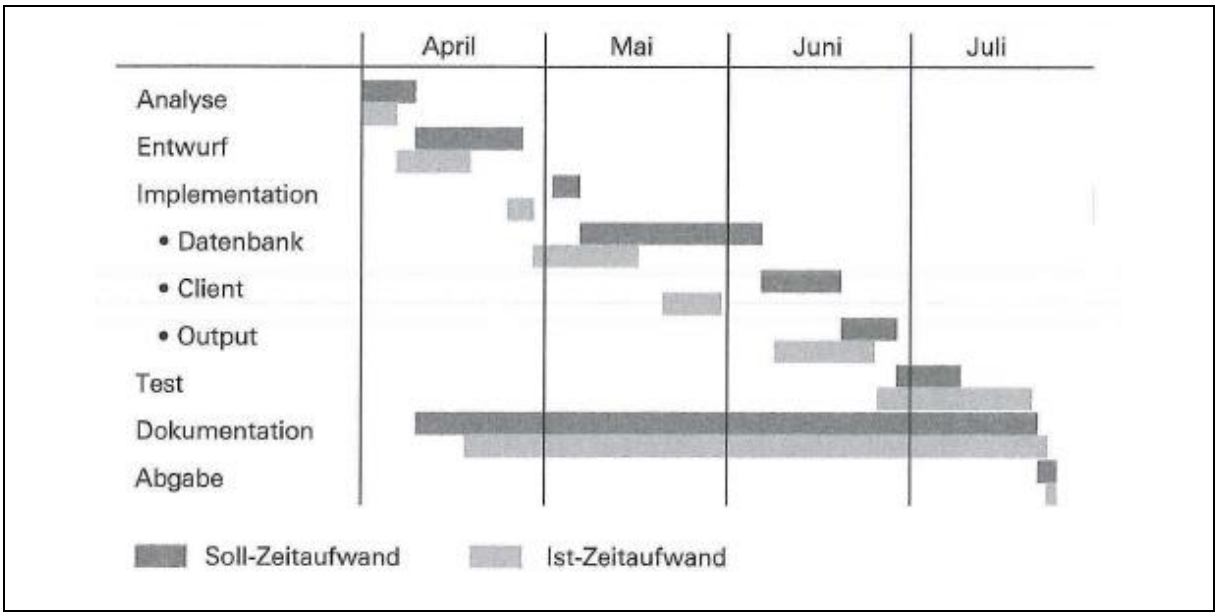
Wir haben nun alle Informationen beisammen, die wir benötigen. Unter anderem auch Anhaltspunkte, wann die einzelnen Dokumente vom Vorgesetzten erwartet werden. Deshalb sollten wir jetzt in der Lage sein, den ungefähren Aufwand einzuschätzen, den wir für die das Projekt erwarten. Erstellen wir nun einen Zeitplan, in dem wir die Zeit beziehungsweise den Aufwand nach Tätigkeiten planen.

Berechnen Sie immer genügend Reserven ein, um auch bei unvorhergesehenen Zwischenfällen noch reagieren zu können!

Je mehr Erfahrung Sie haben, desto genauer können Sie einschätzen, wie viel Zeit Sie für eine bestimmte Aufgabe benötigen. Für den Anfang sollten Sie auf den geschätzten zeitlichen Aufwand noch einmal **50 Prozent dazu rechnen**.

Während des Projekts tragen Sie im Zeitplan laufend ein, wie viel Zeit Sie **effektiv** für die einzelnen Phasen benötigt haben. Am Schluss haben Sie einen **Soll-Ist-Vergleich**, der Ihnen hilft, beim nächsten Projekt besser zu planen.

Projekt-Aufwand (Soll-/Ist-Vergleich)



In diesem Beispiel wurde viel Zeit für die **Analyse** und den **Entwurf** einberechnet, dafür dauerten die Tests um einiges länger als geplant. Beim nächsten vergleichbaren Projekt sollten Sie also weniger Zeit für die Entwurfsphase, aber mehr Zeit für die **Testphase** einplanen (das ist in der Praxis oft der Fall!).

Kontrollfragen

1. Beschreiben Sie in zwei, drei Sätzen, welche Rolle das Realisierungskonzept im Rahmen der Softwareentwicklung spielt!

   Es zeigt auf wie ein Projekt entstehen soll, und am Ende auch ob das eigentliche Ziel eines Projektes erreicht wurde.

   Zeigt auch dem Projektausschuss um was es sich handelt und wie viel Arbeit wo ansteht

   Zukunftssicht: Was wir machen wollen?
   Zurückblickend: Haben wir das erreicht was wir wollten?

2. Nennen Sie die sechs Teile (Kapitel) eines Realisierungskonzepts!

   * Allgemeine Projektangaben
   • Meilensteine des Projekts (Termine)
   • Anforderungen: Eigenschaften und Funktionen der Applikation (Pflichtenheft)
   • Analyse: Ist-Situation
   • Analyse: Zielgruppe(n) [Stakeholders]
   • Technische Rahmenbedingungen

   Auch können Lösungsvarianten und ansätze aufgeschrieben werden

   Ist an der Prüfung individuell, auch Kreativität gefragt...

3. Erläutern Sie in wenigen Sätzen, warum Sie während des Projekts die effektiv benötigte Zeit im Zeitplan nachtragen sollten!

   Da in der Informatik ein Zeitplan eines Projektes nie von Anfang an vorhergesehen werden kann, da immer in Veränderung, neue Features etc.

6 Tests vorbereiten und planen

Sie können sich zwar einfach einmal hinsetzen und mit Ihrer Applikation zu «spielen» beginnen. Sicher werden Sie so sogar den einen oder anderen Fehler finden und korrigieren können. Allerdings ist so ein Vorgehen nicht sehr effizient. Wie jede andere Tätigkeit bei der Softwareentwicklung sollten Sie auch das **Testen** planen. Hierbei gibt es einige Dinge zu beachten, damit am Schluss nichts Wichtiges vergessen geht!

6.1 Was soll getestet werden?

Folgende Bereiche müssen Sie in Ihren Test berücksichtigen:

- Datenvalidierung
- Bedienbarkeit
- Datenkonsistenz
- Sicherheitsaspekte

Vergessen Sie nicht, dass Sie alles sowohl im **Frontend** wie auch im **Backend** testen!

6.2 Welche Testarten gibt es?

Es gibt ganz unterschiedliche Arten, wie ein Programm getestet werden kann.

6.2.1 Black-Box-Test

Bei einem Black-Box-Test betrachten Sie die Applikation als **schwarzen Kasten**, dessen Innenleben Sie nicht kennen. Sie testen quasi anhand der Bedienungsanleitung. Das ist so, wie ein Benutzer die Arbeit mit der Applikation erlebt.

6.2.2 White-Box-Test

Beim White-Box-Test gehen Sie davon aus, dass Sie den **inneren Aufbau** und den **Code der Applikation** kennen (der «Kasten» ist weiss / durchsichtig). Sie sind daher in der Lage, ganz andere und viel detailliertere Testschritte durchzuführen. Sie können z. B. ganz gezielt einen Test machen, bei dem eine spezielle Fehlerbehandlung notwendig wird, und so überprüfen, ob das auch wirklich passiert. Ein Entwickler mit Zugriff auf den Code kann solche Tests machen.

6.2.3 Konstruktiver Test

Wenn Sie Tests machen, um z.B. Ihrem Kunden zu zeigen, dass **alles korrekt funktioniert**, spricht man von konstruktivem Testen. Das Ziel eines solchen Tests besteht darin, zu zeigen, dass der Kunde alles erhält, was er von der Applikation erwartet.

6.2.4 Destruktiver Test

Im Gegensatz zum konstruktiven Test versucht man beim destruktiven Testen gezielt die **Applikation in Bedrängnis zu bringen**. Hier geht es nicht darum, dass etwas korrekt ist, sondern man sucht bewusst Probleme. Destruktive Tests sind sehr wichtig, aber nicht immer ganz einfach. Wer soll denn solche Tests machen? Der Programmierer einer Funktion weiss ja genau, was er programmiert hat; für ihn ist es nicht einfach, seinen eigenen Code «kaputt» zu testen. Wenn Sie Tests anschauen, die ein Entwickler macht, finden Sie meistens konstruktive Tests. Der Entwickler will doch zeigen, was er gut und richtig gemacht hat und nicht, welche Fehler im Code sind.

6.3 Testplan erstellen

Bevor Sie mit dem Programmieren beginnen, planen Sie die Applikationstests mithilfe der Angaben aus Ihrem Realisierungskonzept. Dazu erstellen Sie einen Testplan (manchmal auch Testprogramm genannt), mit dem folgende Testsituationen jeweils anhand mehrerer Testfälle getestet werden sollen:

- Normale Benutzung der Applikation mit gültigen Eingaben
- Normale Benutzung der Applikation mit Varianten gültiger Eingaben
- Normale Benutzung der Applikation mit ungültigen Eingaben
- Unvorhergesehene Benutzung der Applikation
- Gesamteindruck

Sie erstellen dazu eine **Testtabelle**, in der Sie die einzelnen **Testsituationen** darstellen. Beschreiben Sie ebenfalls, welche **Reaktion** Sie von der Applikation erwarten.

Ihr Testplan sollte Tests für die folgenden Bereiche beinhalten:

- Korrekte Eingaben
- Spezialfälle korrekter Eingaben
- Ungültige Eingaben
- Falsche Bedienung
- Usability

Zu jedem Testschritt notieren Sie die notwendigen Voraussetzungen, eine genaue Beschreibung der Durchführung sowie das erwartete Ergebnis.

Kontrollfragen

1. Erläutern Sie in drei bis vier Sätzen den Unterschied zwischen einem Black-Box- und einem White-Box-Test!

2. Machen Sie zwei Beispiele für Tests aus dem Bereich «Spezialfälle gültiger Eingaben» und erklären Sie, weshalb Sie diese gewählt haben!

Historie

Dokument erstellt durch Rolf H. Hirschi	Version 1.0 / 17.08.2015
Erweitert/Angepasst durch Roman Thommen	Version 1.1 / 09.08.2017