

# HTW Berlin

FB1 - Computer Engineering

## Praktikumsbericht

*durchgeführt bei*  
Skateistan gGmbH  
Oppelner Str. 29  
10997 Berlin

*s0536440*  
Manuel Bergler  
Urbanstr. 26  
10967 Berlin

## *Inhaltsverzeichnis*

### [Einleitung](#)

### [Beschreibung der eingerichteten IT-Infrastruktur](#)

#### [Webserver](#)

##### [Systemkonfiguration des Webservers](#)

##### [Remote-Zugriff](#)

##### [Backups](#)

##### [Apache2 Einrichtung](#)

##### [Atlassian Confluence - Kollaborationssoftware als Java Applikation](#)

#### [Lokaler Dateiserver](#)

### [Beschreibung wiederkehrender Tätigkeiten](#)

#### [Administration des Webservers](#)

#### [Confluence Upgrades](#)

### [Beschreibung einmalig durchgeführter Projekte](#)

#### [Upgrade der Webseite zu Drupal 7 und Erstellung eines neuen responsive Theme](#)

##### [Motivation](#)

##### [Analyse der benötigten Module](#)

##### [Upgrade Drupal 6 - Drupal 7](#)

##### [git](#)

##### [Front-End Framework](#)

##### [Aufbau des Drupal Themes](#)

##### [Inhalts-Typen für Drupal](#)

##### [Aufbau der Regionen](#)

##### [Drupal Views](#)

##### [Drupal Panels](#)

##### [SCSS / CSS Struktur](#)

##### [Drupal Contexts](#)

### [Schluss](#)

## Einleitung

Skateistan begann 2007 als Basisprojekt im Bereich 'Sport für Entwicklung' auf den Straßen Kabuls und hat sich im Laufe der Jahre zu einer preisgekrönten, internationalen NGO (Nichtregierungsorganisation) mit Projekten in Afghanistan, Kambodscha und Süd-Afrika entwickelt. Skateistan ist die erste internationale Entwicklungsinitiative, die Skateboarding mit Bildungsprogrammen kombiniert. Skateistan ist politisch unabhängig und unterstützt die Inklusion jeglicher Ethnien, Religionen und sozialen Herkunft. Skateistan strebt stets danach, ein qualitativ hochwertiges und innovativ soziales Projekt zu sein und arbeitet mit Kindern und Jugendlichen im Alter von 5 bis 18 Jahren. Mehr als 50% der Schüler/innen sind auf der Straße arbeitende Kinder und fast 40% unserer Schüler sind Mädchen.

Seit 2012 befindet sich der Hauptsitz der Verwaltung in Berlin. Die Anzahl der Mitarbeiter in Berlin beträgt derzeit 5, wobei die Anzahl aller internationaler Mitarbeiter etwa 60 beträgt. Ich selbst habe hier von April 2012 bis Ende November 2014 die Rolle des IT-Managers übernommen.

Die Wahl für diesen Betrieb fiel mir nicht schwer, da ich auch selbst Skateboard fahre. Außerdem war es interessant in einem kleinen Betrieb zu arbeiten, in dem man die Freiheit hat, selbst seine Ideen in die IT-Infrastruktur mit einzubringen. Zugleich war es ein sehr positiver Aspekt, dadurch Kindern, die in einer schwierigen Situation aufwachsen, indirekt zu helfen. Ein Großteil der Finanzierung für die Organisation stammt aus Spendengeldern, wobei sämtliches Marketing online abläuft. Hierfür muss eine professionelle Webseite die Ziele und Aktivitäten der Organisation widerspiegeln. Und des Weiteren bedarf es wegen der Verbreitung auf mehreren Kontinenten interner Tools, die für die Verwaltung, interne Kommunikation und Austausch von Medien benutzt werden.

# Beschreibung der eingerichteten IT-Infrastruktur

## Webserver

### Systemkonfiguration des Webserver

Die Webseite von Skateistan und das interne Kommunikationstool laufen auf einem Webserver der bei Amazon Web Services gehostet ist. Dieser Server der Teil des EC2 (Elastic Compute Cloud) ist wurde zu Beginn meiner Tätigkeit eingerichtet. Es handelt sich hierbei um einen Server auf dem derzeit die Ubuntu Langzeit-Support-Version 10.04 läuft. Die Wahl für einen Server in der Cloud wurde zum einen aus Kostengründen getroffen, zum Anderen jedoch auch da sich dieser Server per Mausklick upgraden lässt. Es gibt bei EC2 verschiedene sogenannte "instance types" die sich in der Hardware und selbstverständlich auch im Preis groß unterscheiden. Für die Anforderungen bei Skateistan wurde ein Server mit 1 vCPU (virtuellen CPU) und 3.75 GB RAM gewählt. Der Server hat mit 410GB Festplattenspeicher außerdem genügend Platz für sämtliche Medien-Dateien die kurzfristig über FTP ausgetauscht werden können.

Der Server lässt sich über folgende Bedienoberfläche verwalten:

Filter by tags and attributes or search by keyword									
Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP	
kickflip	i-c9983ea9	m1.medium	us-east-1c	running	2/2 checks ...	OK	ec2-50-17-196-178.co...	50.17.196.178	

Instance: i-c9983ea9 (kickflip)		Elastic IP: 50.17.196.178			
Description	Status Checks	Monitoring	Tags		
Instance ID	i-c9983ea9		Public DNS	ec2-50-17-196-178.compute-1.amazonaws.com	
Instance state	running		Public IP	50.17.196.178	
Instance type	m1.medium		Elastic IP	50.17.196.178	
Private DNS	ip-10-158-95-242.ec2.internal		Availability zone	us-east-1c	
Private IPs	10.158.95.242		Security groups	core_services . view rules	
Secondary private IPs	-		Scheduled events	One scheduled event	
VPC ID	-		AMI ID	kickflip 19072013 (ami-a0453cc9)	
Subnet ID	-		Platform	-	
Network interfaces	-		IAM role	-	

Hier ist auch zu sehen, dass der Server mit einer "Elastic IP" versehen wurde, die über das öffentliche Internet erreichbar ist. Der DNS-Eintrag für [www.skateistan.org](http://www.skateistan.org) wurde auf diese IP-Adresse verwiesen.

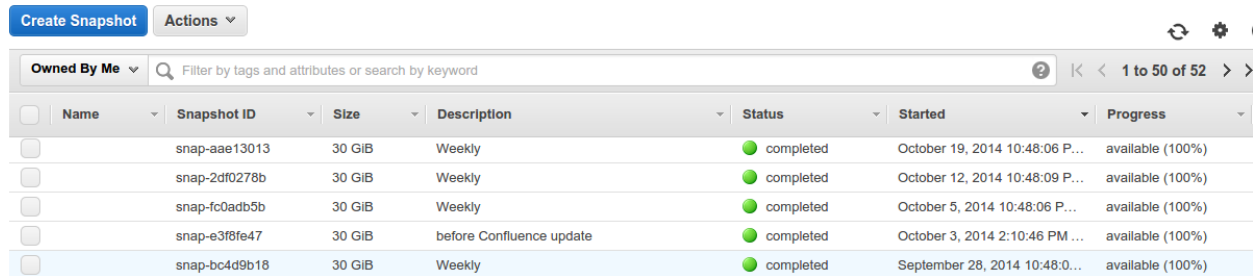
### Remote-Zugriff

Da sich der Server in der Cloud befindet und es keinen physikalischen Zugriff zu ihm gibt, ist es unerlässlich eine sichere Verbindung zum Backend des Servers zu haben. Eine Möglichkeit ist auf die Konsole über die AWS Web-Oberfläche zuzugreifen, was sich jedoch nicht als sehr komfortabel darstellt. Aus diesem Grunde wurde von mir SSH eingerichtet und ausschließlich der Zugriff über die Kommandozeile für die Administration gewählt. Es wurde mein private Key im Homeverzeichnis auf dem Ubuntu Server hinterlegt, was es ermöglicht sich ohne Passwort mit dem Server zu verbinden:

\$ ssh manuel@kickflip.skateistan.org

## Backups

Eine praktische Angelegenheit als Administrator, ist dass Backups vom gesamten Server sehr einfach anzulegen sind. Diese werden Snapshots genannt:



The screenshot shows a web interface for managing snapshots. At the top, there are buttons for 'Create Snapshot' and 'Actions'. Below these is a search bar with the text 'Owned By Me' and a filter option 'Filter by tags and attributes or search by keyword'. The main part of the interface is a table with columns: Name, Snapshot ID, Size, Description, Status, Started, and Progress. There are five rows of snapshots, all with a status of 'completed' and 'available (100%)'.

Name	Snapshot ID	Size	Description	Status	Started	Progress
	snap-aae13013	30 GiB	Weekly	completed	October 19, 2014 10:48:06 P...	available (100%)
	snap-2df0278b	30 GiB	Weekly	completed	October 12, 2014 10:48:09 P...	available (100%)
	snap-fc0adb5b	30 GiB	Weekly	completed	October 5, 2014 10:48:06 P...	available (100%)
	snap-e3f8fe47	30 GiB	before Confluence update	completed	October 3, 2014 2:10:46 PM ...	available (100%)
	snap-bc4d9b18	30 GiB	Weekly	completed	September 28, 2014 10:48:0...	available (100%)

Dieser werden mittels einem Shell Skript täglich erstellt. \*\*\* skript einfügen \*\*\*

## Apache2 Einrichtung

Zu Beginn meiner Tätigkeit wurden auf dem Server die nachfolgenden Dienste installiert und konfiguriert. Als Webserver wurde Apache2 ausgewählt. Die Webseite von Skateistan war zu diesem Stand mit dem Content-Management-System Drupal 6 vorhanden. Da es für die Organisation neben [www.skateistan.org](http://www.skateistan.org) noch weitere Subdomains und andere Seiten gibt, die jedoch alle mit Drupal 6 verwaltet werden, habe ich mich dafür entschieden eine einzige Instanz für fast alle Seiten zu verwenden und dann mittels Apache2 VirtualHosts zu erstellen. Hiermit ist es möglich mehrere Subdomains auf den gleichen Root-Ordner zu verweisen.

## Atlassian Confluence - Kollaborationssoftware als Java Applikation

Auf dem Server wurde desweiteren ein Kollaborationssoftware installiert die der internen Kommunikation und dem Dokumentenaustausch dient. Diese Software wird benutzt um die alle weltweiten Projektmitarbeiter am laufenden zu halten und auch den Austausch durch Email zu vermindern. Hierbei ist es einfach möglich alle Neuigkeiten auf einen Blick zu sehen und gemeinsam an Projekten und Dokumenten zu arbeiten. In Absprache mit allen Abteilungen haben wir uns darauf geeinigt folgende sogenannte "Spaces" einzurichten:

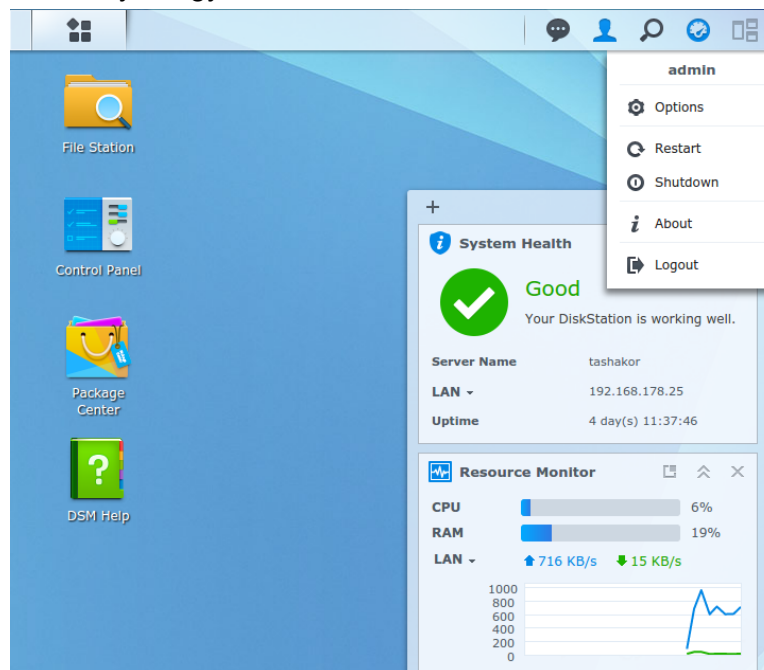
Bei diesen Spaces wird zwischen verschiedenen Layouts und auch Benutzerberechtigungen unterschieden. Je nach Aufgabenbereich können so nur die Dokumente mit Mitarbeitern geteilt werden die wirklich relevant sind.

Atlassian Confluence läuft als Java Applikation unter einem Apache Tomcat Server und greift im Hintergrund auf eine MySQL Datenbank zu.

Nach der Installation kann Confluence einfach als Service behandelt werden und die Administration ist dadurch stark vereinfacht:  
`$sudo /etc/init.d/confluence start`

## Lokaler Dateiserver

Da die Anforderungen für den Dateiserver relativ niedrig waren, es nur ein geringes Budget gab und hauptsächlich darum ging, Daten über das Netzwerk verfügbar zu machen, automatische Backups zu erstellen und zugleich auch einen Remote Zugriff zu ermöglichen fiel meine Wahl auf eine Synology DiskStation.



In der DiskStation wurden 2 x 3 TB Festplatten in einem RAID-1 Verbund eingebaut. Die Festplatten wurden mit ext4 formatiert, da das NAS ein Linux-Betriebssystem hat. Laut Weboberfläche sind somit 2.68 TB belegbar.

Number	Model	Serial number	Firmware ...	Disk Size	Temperat...	Disk Type	4K ...	S.M.A.R.T. Sta...	Status	Volume
Disk 1	WD30EZR...00SP...	WD-WCC4E11643...	80.00A80	2794.52 GB	28 °C / 82 ...	HDD	No	Normal	Normal	Volume 1
Disk 2	WD30EZR...00SP...	WD-WCC4E12461...	80.00A80	2794.52 GB	29 °C / 84 ...	HDD	No	Normal	Normal	Volume 1

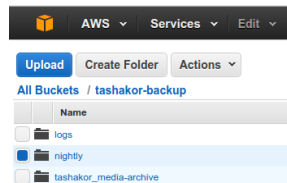
Da nicht alle Daten für jeden verfügbar sein sollen, es jedoch administratorisch zu viel Aufwand ist für jeden einzelnen Benutzer ein einzelnes Konto einzurichten, wurden verschiedene Abteilungskonten angelegt:

- admin
- finance
- guest
- management
- standard

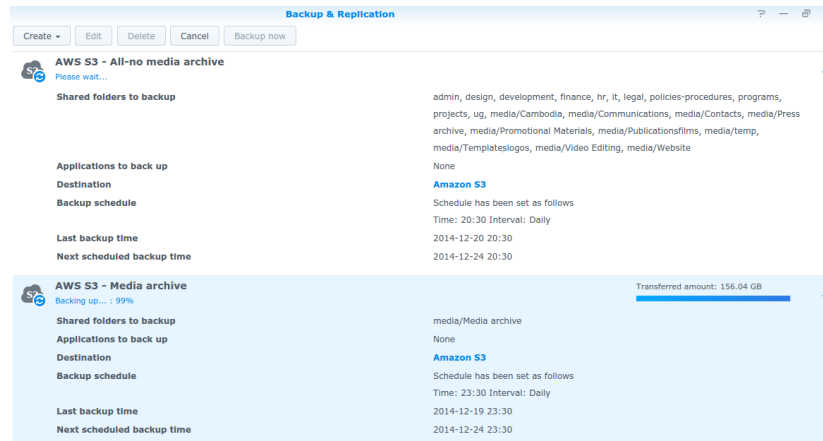
Für den Fileserver wurden dann sogenannte “Shared Folders” angelegt, was einer Freigabe auf einem Windows Fileserver gleichzusetzen ist. Diese können innerhalb des lokalen Netzwerkes als Netzlaufwerk verbunden werden:

Name	Description	Status	Volume
admin		-	Volume 1
design		-	Volume 1
development		-	Volume 1
finance		-	Volume 1
hr		-	Volume 1
it		-	Volume 1
legal		-	Volume 1
media		-	Volume 1
policies-procedures		-	Volume 1
programs		-	Volume 1
projects		-	Volume 1
ug		-	Volume 1

Für alle der Shared Folders wird ein Backup in der AWS Cloud erstellt. Dieser Dienst nennt sich S3 (Simple Storage Service). In S3 wurde ein sogenannter Bucket erstellt, für welchen dann separate Zugangsdaten und ein Access Key generiert werden.



Vor Einrichtung der Backup-Tasks habe ich die vorliegenden Daten analysiert. Der Großteil an Daten befindet sich im Media Archive, in dem von allen Projekt-Standorten alle Video- und Bilddateien archiviert werden. Da der Rest der Dateien verhältnismäßig klein ist, es sich jedoch um wichtige Dokumente wie Buchhaltung, Anträge für Zuschüsse und sonstiges handelt hielt ich es für wichtig, dass das Backup in zwei unterschiedliche Aufträge aufgeteilt wird. So kann sichergestellt werden, dass selbst wenn das Media Archive-Backup einmal fehlschlägt, weil zum Beispiel die Dateigröße überschritten wird, alle Dokumente trotzdem gesichert werden. Wie in nachfolgender Grafik zu sehen ist, gibt es nun zwei Tasks die jede Nacht laufen.



Die Synology DiskStation macht es außerdem sehr einfach Dateien wiederherzustellen, es kann einfach eine Datei und ein Datum ausgewählt werden, da das Backup inkrementell erstellt wird.

Da viele der Mitarbeiter nicht im HQ in Berlin sitzen, jedoch trotzdem Zugriff auf wichtige Dokumente benötigen, habe ich desweiteren einen Fernzugriff eingerichtet. Dieser kann zum Einen über die Weboberfläche erfolgen, da DynDNS eingerichtet wurde. Zum anderen, kann jedoch auch direkt auf einzelne Netzlaufwerke zugegriffen werden und zwar mittels dem SMB oder AFP Protokoll. Hierfür mussten auch Regeln in der Firewall erstellt werden und Port-Forwarding auf dem Router eingerichtet werden.

## Beschreibung wiederkehrender Tätigkeiten

### Administration des Webserver

Für den Server wurden wöchentlich Updates für installierte Pakete installiert.

Dies kann einfach mittels:

```
$ sudo apt-get update && sudo apt-get upgrade
```

durchgeführt werden, da alle Kernel-Updates und Ubuntu Release-Upgrades zurückgehalten werden und sonstige installierte Pakete als unkritisch eingestuft werden.

### Confluence Upgrades

Confluence wird rege weiter entwickelt und es gibt alle paar Monate eine neue Version, aus welchem Grunde folgender Ablauf gefolgt werden muss:

1. Backup der MySQL Datenbank



- ```
$ sudo mysqldump --defaults-file=/root/.my.cnf 'confluence' > confluence-db-5.1.3.sql
```
2. Backup der Confluence Verzeichnisse:  
Confluence Installation directory (/var/atlassian/application-data/confluence) & Home directory (/opt/atlassian/confluence)
  3. Download der neuesten Version des Linux 32-bit Installers
  4. Der Installer muss nun vorerst ausführbar gemacht werden:  

```
$ chmod a+x atlassian-confluence-5.4.4-x32.bin
```
  5. Nun wird der Installer ausgeführt:  

```
$ sudo ./atlassian-confluence-5.4.4-x32.bin
```

  
und es wird Option (3) "Upgrade existing Confluence installation" ausgewählt

## Beschreibung einmalig durchgeführter Projekte

### Upgrade der Webseite zu Drupal 7 und Erstellung eines neuen responsive Theme

#### Motivation

Auf Grund zunehmender Webseiten-Besucher mit mobilen Geräten, musste die Seite redesigned werden, da diese sich nicht der Bildschirmbreite des Endgerätes anpasst und somit der Text nicht sonderlich lesbar ist. In den letzten Jahren entstand das sogenannte "Responsive Web Development" bei dem sich der Inhalt je nach Breite des Bildschirms anpasst. Somit müssen viele Gestaltungselemente dynamisch angelegt werden und nicht fixiert in einer bestimmten Pixel-Größe. Da dieses Redesign einige große Veränderungen im Backend herbeiruft und es eine neue Version von Drupal mit höherer Sicherheit und neuen Features gab, habe ich mich entschieden zugleich ein Core Upgrade durchzuführen.

Die Basis für das Redesign war eine Photoshop-Datei von unserem Designer, der in Absprache mit der Communications-Abteilung die Gestaltungselemente und den Aufbau der neuen Seite bestimmt hat.

#### Analyse der benötigten Module

Zu Beginn wurde analysiert welche Drupal Module aktiviert sind und ob es für diese bereits eine neue Version in Drupal 7 gibt, oder ob diese durch vergleichbare Module ersetzt werden können. Einige Module wurden auch in den Drupal Core integriert und konnten somit aussen vor gelassen werden. Nachfolgend ist ein Auszug aus einem Dokument zu sehen, das ich erstellt habe um den Upgrade-Pfad und Abhängigkeiten der Module zu dokumentieren:

|   | A              | B          | C            | D     | E          | F                         | G                                |
|---|----------------|------------|--------------|-------|------------|---------------------------|----------------------------------|
|   | Module         | Compatible | coreIncluded | Use?  | installed? | Comments                  | Migration / solution             |
| 1 | acl            | X          |              | o     |            |                           |                                  |
| 2 | admin_menu     | X          |              | o     |            |                           |                                  |
| 3 | adminrole      | X          | X            | incl. |            |                           |                                  |
| 4 | advanced_forum | X          |              | o     |            | requires views and ctools |                                  |
| 5 | author_pane    | X          |              | o     |            |                           |                                  |
| 6 | auto_username  | no         |              | o     |            |                           | uninstall module, no further use |
| 7 | better_perms   | no         |              | o     |            |                           | uninstall module, no further use |
| 8 | captcha        | X          |              | X     |            | beta available            |                                  |

Anschließend habe ich angefangen eine Test-Umgebung einzurichten. Diese ist über <http://dev.skateistan.org> zu erreichen. Hierfür habe ich den gesamten Ordner aus dem Verzeichnis /var/www/www/\* nach /var/www/dev/ kopiert, einen neuen VirtualHost Eintrag erstellt und die Subdomain in DNS eingetragen.

```
manuel@kickflip:~$ cat /etc/apache2/sites-enabled/dev.skateistan.org
<VirtualHost *:80>
    ServerName      dev.skateistan.org
    DocumentRoot    /var/www/dev

    ErrorLog /var/log/apache2/dev.error.log
    CustomLog /var/log/apache2/dev.access.log combined
    CustomLog /var/log/apache2/access.log.COMBINED combined
</VirtualHost>
```

Außerdem musste die Drupal MySQL Datenbank geclost werden. Dies ist einfach möglich in dem man einen mysqldump in eine Pipeline zur Erstellung einer neuen Datenbank anhängt:

```
$ mysqldump --opt --user='skateistan' --password='dummy' 'skateistan_org' | mysql
--user='skateistan_dev' --password='dummy' 'skateistan_dev'
```

## Upgrade Drupal 6 - Drupal 7

Anschließend wurden alle vorhandenen Module und der Drupal Core auf die letzte Drupal 6-Version aktualisiert. Dann habe ich die Seite in den sogenannten “Wartungsmodus” geschaltet, was verhindert dass Datenbankänderungen gemacht werden und somit zu einer Inkonsistenz nach dem Upgrade führen würden.

Nun wurden alle Module deaktiviert, da diese nach dem Upgrade ungültig sein werden wegen Inkompatibilität zur neuen Version und das Standard-Theme das beim Installieren von Drupal aktiviert ist namens “Garland” aktiviert.

Das Upgrade selbst ist relativ einfach, es müssen nur die neuesten Dateien heruntergeladen und im Root-Ordner verschoben werden. Am einfachsten ist dies mittels:

```
$ git clone http://drupalcode.org/project/drupal.git
```

Da Drupal sämtliche Datei-Uploads in einem Ordner verwaltet benötigt der www-data Benutzer, der von Apache2 benutzt wird, Schreibrechte darin.

```
$ sudo chmod -R 775 sites/default/files
```

Nun können mittels <http://dev.skateistan.org/update.php> die Datenbankinträge aktualisiert werden, so dass diese dem Schema von Drupal 7 entsprechen - dies dauert ca. 30 Minuten.

Jetzt mussten die Drupal 7 Versionen aller noch benötigten Module heruntergeladen und im /var/www/dev/sites/all/modules Ordner entpackt werden. Nach der Dateioperation muss dann wieder das update.php Skript laufen um die Datenbankinhalte der zugehörigen Module auf den neuesten Stand zu bringen.

Nach diesem Schritt befindet sich die Seite im Grundzustand von Drupal 7 und es konnte nun damit begonnen werden, das neue Theme zu erstellen.

Da hierfür viele Änderungen gemacht werden müssen, habe ich mir die letzte Version der dev-Webseite auf meinen Computer kopiert und werde von dort aus programmieren und dann hin und wieder einen Zwischenstand auf den Webserver hochladen.

## git

Für die Programmierung habe ich mich dafür entschieden, die Versionsverwaltungssoftware git zu verwenden um alle Änderungen nicht nur nachvollziehen, sondern bei Bedarf auch rückgängig machen zu können. Hierfür habe ich lokal einen "dev" Branch erstellt, auf welchem die Entwicklung stattfindet. Während der Programmierung wird dann der Code in ein privates Repository bei GitHub gepushed und kann dann von dort aus auf den Server in das /var/www/dev Verzeichnis gecloned werden. Der "master" Branch kann dann später in das /var/www/www gecloned werden, wenn die Seite fertig gestellt wurde.

## Front-End Framework

Für die Entwicklung des Themes habe ich mich für ein responsive front-end framework namens Zurb Foundation entschieden, was einem viel Arbeit abnimmt und zugleich die Verwendung von standardisierten CSS-Elementen vereinfacht. Des weiteren habe ich mich dafür entschieden die Stylesheets, in SCSS (Sassy CSS) zu schreiben, da man hiermit ein Stylesheet mit Variablen und Verschachtelungen bauen kann welches dann zu CSS kompiliert wird. Hierfür musste Ruby und das Compass Gem installiert werden. Außerdem wird Bower dafür verwendet um Foundation herunterladen und aktuell zu halten, und auch um andere Libraries herunterladen die für Foundation benötigt werden.

```
$ npm install -g bower grunt-cli
```

```
$ sudo gem update --system && gem install compass foundation
```

## Aufbau des Drupal Themes

Nun kann im sites/all/themes Ordner die Grundstruktur des neuen Themes mittels:

```
$ foundation new skateistan_theme
```

erstellt werden. Um diesen Ordner für Drupal erkennbar zu machen, müssen noch folgende Dateien hinzugefügt werden:

- skateistan\_theme.info

Hier werden die unterschiedlichen Regionen des Layouts definiert, festgelegt in welcher Datei sich die kompilierten CSS Dateien befinden und die Version definiert.

```

name = Skateistan Theme
package = Core
version = 1.0
core = 7.x

engine = phptemplate

stylesheets[all][] = stylesheet/style.css

regions[header] = Header
regions[content_top] = Content Top
regions[content] = Content
regions[content_bottom] = Content Bottom
regions[sidebar] = Sidebar
regions[footer] = Footer

```

- templates Ordner mit folgenden Dateien in denen dann die zuvor definierten Regionen als CSS Klassen verwendet werden:
  - node.tpl.php: Das HTML/PHP Gerüst für Blog-Einträge
  - page.tpl.php: Das HTML/PHP Gerüst für alle anderen Seiten
- template.php Datei in der zusätzlich definiert werden kann, ob zu jeder Seite automatisch bestimmte JavaScript Dateien hinzugefügt werden sollen oder zusätzlicher HTML Code

```

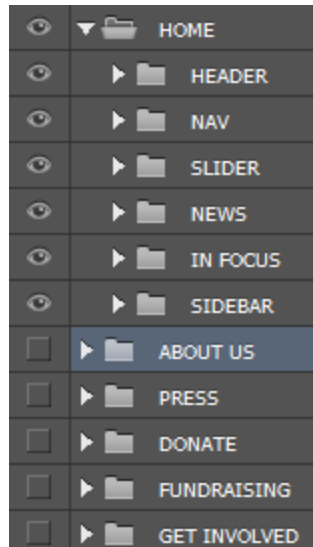
1 <?php
2 function skateistan_theme_preprocess_page(&$vars) {
3
4     drupal_add_js(drupal_get_path("theme", "skateistan_theme")."/javascript/foundation/foundation.js");
5     drupal_add_js(drupal_get_path("theme", "skateistan_theme")."/javascript/foundation/foundation.forms.js");
6     drupal_add_js(drupal_get_path("theme", "skateistan_theme")."/javascript/vendor/jquery.touchSwipe.min.js");
7     drupal_add_js(drupal_get_path("theme", "skateistan_theme")."/javascript/global.js");
8
9     // viewport
10    $element = array(
11        '#tag' => 'meta',
12        '#attributes' => array(
13            'name' => 'viewport',
14            'content' => 'width=device-width, initial-scale=1, minimum-scale=1, maximum-scale=2, user-scalable=1'
15        ),
16    );
17
18    drupal_add_html_head($element, 'mobile_view');
19
20 }

```

Die hier aufzufindende PHP Funktion fügt die JavaScript Bibliotheken die für das Foundation Framework benötigt werden zu jeder Seite hinzu und zugleich wird ein Viewport als Meta Tag hinzugefügt. Dieser informiert den Browser des Webseiten-Besuchers über Informationen wie welche Bildschirmbreite zu verwenden ist und ob die Seite automatisch skaliert werden soll oder nicht. Der interpretierte Tag sieht dann in HTML wie folgt aus:

```
<meta name="viewport" content="width=device-width, initial-scale=1, minimum-scale=1, maximum-scale=2, user-scalable=1" />
```

Auf Basis der folgenden Photoshop-Datei habe ich nun begonnen nach und nach das Layout aufzubauen. Es ist zu vermerken, dass die Grundstruktur für die PHP Gerüste von einem Startertheme übernommen wurde (Zen Starter-Theme), da diese generell immer gleich ist.



Der Aufbau der Seite ist links in den einzelnen Ordnern zu sehen. Home enthält wie auch alle anderen Seiten einen Header, die Navigation eine Sidebar und einen Footer. Außerdem soll auf dieser Startseite unter der Navigation eine Slideshow und dann darunter in zwei nebeneinander angeordneten Spalten zum einen die zuletzt hinzugefügten Blog-Einträge und ein Raum für hervorgehobene Inhalte, wie z. B. neueste Videos, aktuelle Fundraiser oder sonstiges vorhanden sein.

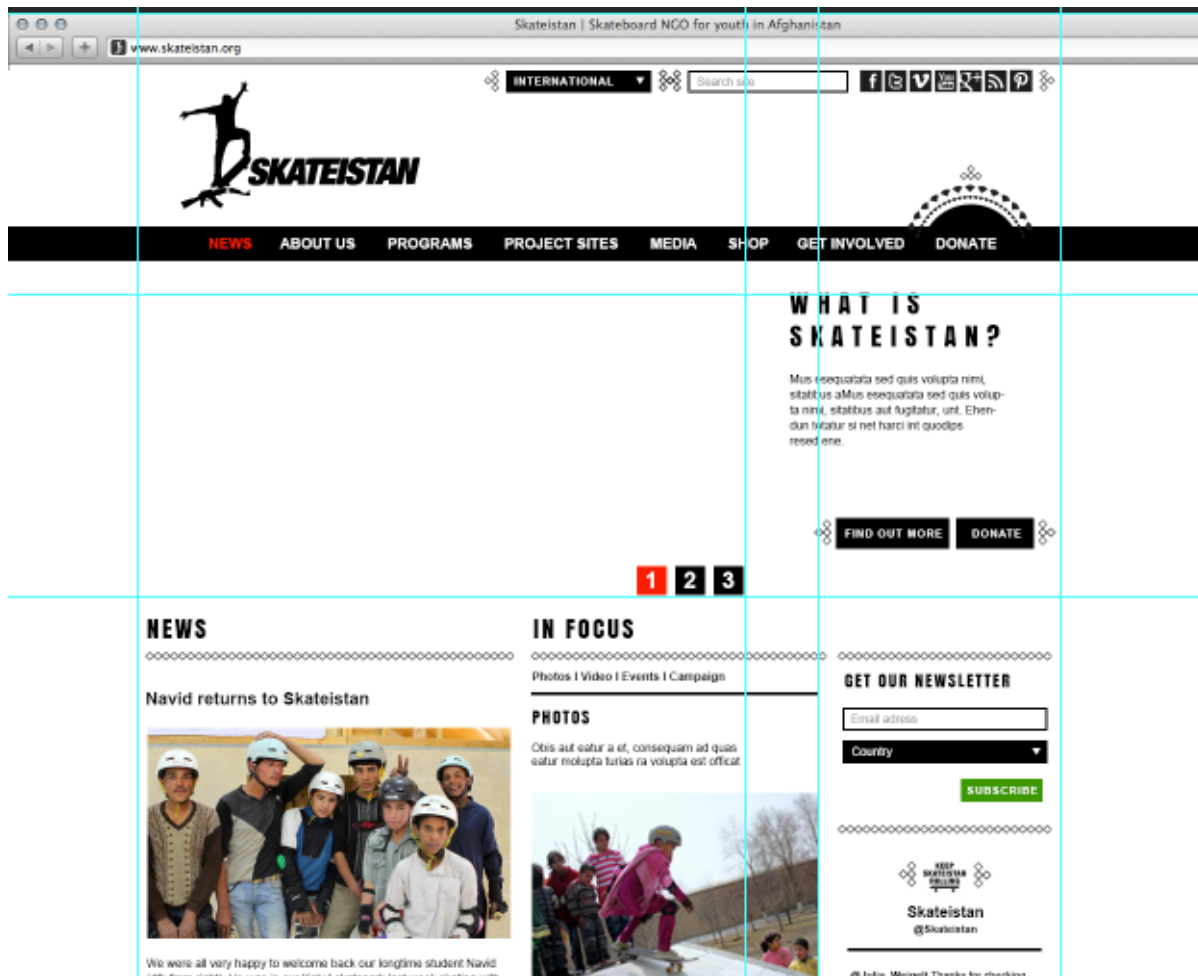
Auf Basis hiervon habe ich recherchiert welche Drupal Module vorhanden sind und dieses Layout ermöglichen. Um Inhalte in Drupal anzulegen gibt es sogenannte Content-Types (Inhalts-Kategorien) womit man unterscheiden kann ob es sich um einen Blog-Eintrag, eine Seite, eine Bildergalerie oder sonstigen Inhalt handelt.

## Inhalts-Typen für Drupal

So wurden in Zusammenarbeit mit der Media / Communications-Abteilung die für die Pflege des Inhalts und die Erstellung von Blog-Einträgen zuständig ist folgende Inhalts-Typen definiert:

- Dynamic Slideshow item: hiermit kann eine neue Slide in der Slideshow definiert werden, außerdem soll hier auch der Ort an welcher sich die Slideshow befinden soll mehr oder weniger dynamisch angegeben werden
- Image Slideshow: hiermit kann eine neue Seite erstellt werden die eine Slideshow von Fotos enthält.
- In Focus - Photo slide item: hier können neue Bilder zu der Slideshow die sich auf der Startseite im hervorgehobenen Bereich befindet hinzugefügt werden
- Keep Skateistan Rolling post: ein spezieller Blog-Eintrag für eine Fundraising-Kampagne die 2x im Jahr statt findet. Diese wird nur unter einer speziellen URL verfügbar sein.
- Page: dies ist der Standard-Typ für Seiten die sich weniger regelmäßig ändern. Der Großteil des Seiten-Inhaltes abgesehen von den Blog-Einträgen wird von diesem Typ sein.
- Skateistan Blog entry: hiermit werden neue Blog-Einträge erstellt, diese können auch mit Kategorien versehen werden.
- Sponsor: im Footer der Seite sollen alle aktuellen Sponsoren von Skateistan angezeigt werden. Diese sollen einfach als Inhalt hinzugefügt werden
- Student Blog entry: ähnlich wie ein normaler Blog-Eintrag, jedoch werden diese Einträge nicht auf der Startseite angezeigt, sondern unter einer anderen URL

## Aufbau der Regionen



Aufbau der Startseite - Designvorgabe

Das page.tpl.php Template ist wie folgt unterteilt:

```
<div class="container">
  <div class="page_header"> ...
</div>
<div class="page_content"> ...
</div>
<div class="page_footer"> ...
</div>
</div>
```

Die hier sichtbaren CSS Klassen unterteilen die Seite in 3 unterschiedliche Teile, den Header, Content und Footer. Die Content Ebene ist in folgende Unterebenen gegliedert, welche dann den Inhalt der jeweiligen Regionen rendern:

```
<?php print render($page['content_top']); ?>
<?php print render($page['content']); ?>
```

```
<?php print render($page['sidebar']); ?>  
<?php print render($page['content_bottom']); ?>
```

Da der Inhalt der Seite dynamisch sein soll und so viel wie möglich mittels der Drupal CMS Administrations-Oberfläche bearbeitet und hinzugefügt werden soll, wollte ich so viel wie möglich auf vorhandene Drupal Module zurückgreifen.

## Drupal Views

Um die Inhalte die in unterschiedlichen Inhalts-Typen erstellt werden zu unterscheiden und anzeigen zu lassen, benötigt man eine Drupal View. In dieser kann nicht nur der anzuzeigende Typ ausgewählt werden, sondern auch eine URL definiert werden durch die diese Seite zu erreichen ist. Für die Skateistan Seite wurden folgende Views erstellt, die alle unterschiedlich konfiguriert wurden:

- Elite partners / Major partners / Partners:  
diese 3 Views sollen den Inhalt der Sponsoren rendern. Da es eine Unterscheidung je nach Finanzierungshöhe gibt, sind diese in 3 Kategorien aufgeteilt. Sie sollen in einem 3-spaltigen Layout dargestellt werden, von daher gibt es 3 unterschiedliche Views hierfür.
- Front Slider:  
diese View ist für die Slideshow die auf der Startseite dargestellt wird gedacht
- InFocus Photo Slideshow:  
diese View rendert alle Bilder dieses Typs in eine Slideshow. Dies ist möglich durch ein zusätzliches Plugin für Drupal Views, dass einen "Slideshow" als Anzeigetyp auswählen lässt.
- Newsfeed:  
wie auf nachfolgender Grafik ersichtlich ist, werden hier die Teaser alle veröffentlichten Blog-Einträge aus der Datenbank geladen und jeweils 5 pro Seite angezeigt

#### ▼ Master details

<b>TITLE</b> Title: <a href="#">None</a>	Access: <a href="#">Permission</a>   <a href="#">View published content</a>
<b>FORMAT</b> Format: <a href="#">Unformatted list</a>   <a href="#">Settings</a> Show: <a href="#">Content</a>   <a href="#">Teaser</a>	<b>HEADER</b> <a href="#">Add</a>
<b>FIELDS</b> The selected style or row format does not utilize fields.	<b>FOOTER</b> <a href="#">Add</a>
<b>FILTER CRITERIA</b> <a href="#">Add</a> ▼ <a href="#">Content: Published (Yes)</a> <a href="#">Content: Promoted to front page (Yes)</a> <a href="#">Content: Type (in Skateistan Blog entry, ...)</a>	<b>PAGER</b> Use pager: <a href="#">Full</a>   <a href="#">Paged, 5 items</a> More link: <a href="#">No</a>
<b>SORT CRITERIA</b> <a href="#">Add</a> ▼ <a href="#">Content: Post date (desc)</a>	

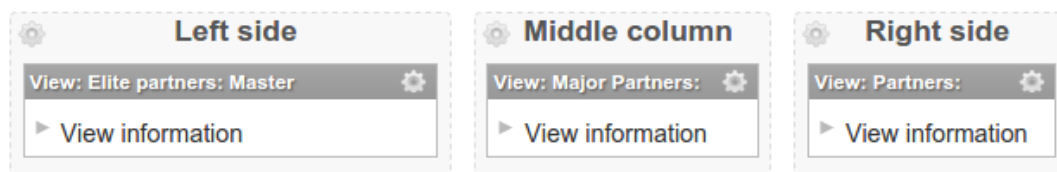
- Programs: für alle bei Skateistan angebotenen Programme für die Schüler, werden hier jeweils Blog-Einträge angezeigt die mit Kategorien versehen wurden, die mit den jeweiligen Programmen zusammen hängen. Außerdem wurde ein statischer HTML Header eingefügt um etwas mehr Information darüber anzubieten.
- Project sites: ähnlich wie bei den Programmen werden hier alle Blog-Einträge angezeigt die mit den jeweiligen Projekt-Standorten verbunden sind.

## Drupal Panels

Da diese Views auf manchen Seiten in einem mehrspaltigen Layout angeordnet werden sollen, musste außerdem noch von einem weiteren Drupal Modul namens Drupal Panels Gebrauch gemacht werden.

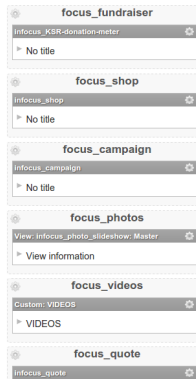
Es wurden hiermit sogenannte Mini-Panels, die später als Drupal Block einer Region zugeordnet werden können, erstellt:

- Ein Panel für den Footer, der die unterschiedlichen Sponsoren-Kategorien in 3 Spalten unterteilen soll. In nachfolgender Grafik ist zu sehen, wie die einzelnen Views den Spalten zugeordnet wurden:



- Das In Focus Layout, das sich auf der Startseite befindet und eine Spalte in mehrere Reihen aufteilt beinhaltet unterschiedliche Blöcke in denen entweder ein Drupal Block oder eine Drupal View gerendert werden kann.





Die Startseite wird dann aus dem In Focus Mini-Panel und der Newsfeed View zusammengesetzt. Die geschieht mittels einer Panel Page im Layout: zweispaltig, 45 % / 55 %.



Alle zuvor erstellten Views, bei denen als Anzeigeformat: Block ausgewählt wurde, sind nun auch als Drupal Blocks verfügbar. Es würde die Möglichkeit bestehen diese manuell einzeln für eine Region zu aktivieren, jedoch verliert man hierbei schnell die Übersicht.

Aus diesem Grunde habe ich mich dafür entschieden das Context Modul zu installieren. Hiermit können Kontexte erstellt werden für die bestimmte Bedingungen zutreffen müssen, wie zum Beispiel ein bestimmter URL-Pfad und dann können dort spezielle Aktionen ausgeführt werden, wie Blocks gerendert, CSS hinzugefügt, JavaScript hinzugefügt uvm.

## SCSS / CSS Struktur

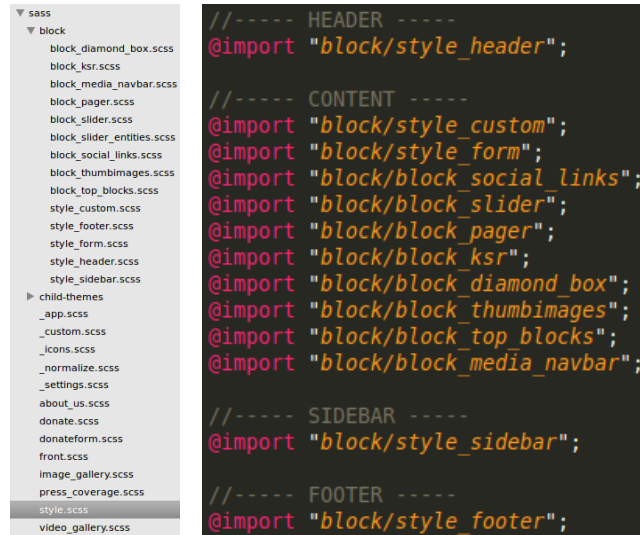
Vorerst werde ich kurz auf die Struktur des erstellten SCSS / CSS Codes eingehen.

Es ist sehr modular angelegt um eine gute Übersicht behalten zu können. Die Inhalte des block Unterordners werden jedoch in das style.scss importiert, da dieses für alle Seiten gilt. Alle anderen scss-Dateien im root Ordner (außer \*.scss) werden zu CSS kompiliert und können dann über Contexts auf vereinzelt Seiten eingebunden werden um den Overhead an CSS Definitionen geringer zu halten.

Um die SCSS Dateien zu kompilieren, habe ich während der Entwicklung ständig

```
$ compass watch
```

im drupal\_theme Ordner in einer Konsole laufen lassen, um die Kompilierung bei jeder Änderung neu anzustoßen und somit immer die aktuellen Stylesheets zu verwenden.



Da es sich um eine responsive Seite handelt, wurden vorher sogenannte Breakpoints festgelegt. Das sind Bildschirmbreiten bei denen sich das Layout ändern soll. Mittels CSS können so je nach Gerät und Ausrichtung unterschiedliche CSS-Styles angewendet werden.

Ich habe mich für folgende Größen entschieden:

```
$breakpointTiny: 400px;  
$breakpointSmall: 550px;  
$breakpointMedium: 768px;  
$breakpointLarge: 960px;
```

Ein großer Unterschied zur klassischen Web-Entwicklung ist, dass in den Stylesheets die meisten Werte in Prozent ausgedrückt werden, da sie sich somit dynamisch der Breite des Browser-Fensters anpassen. Somit sieht der Inhalt der Seite auf sämtlichen Gerät gut aus und die Texte sind leserlich.

Außerdem habe ich den sogenannten “mobile-first” Ansatz gewählt. Das heißt alle CSS Definitionen sind grundlegend für mobile Geräte, also kleiner als der oben festgelegte \$breakpointTiny (400px) gedacht und für alle größeren Bildschirme müssen dann Änderungen vorgenommen werden. Hierzu nachfolgend ein Beispiel der “About Us” Seite. Links ist der SCSS Code zu sehen und die unterschiedlichen Definitionen für die Breakpoints.

```

1  $include-html-classes: false !default;
2  @import "app";
3
4  // What next
5  .what-next-first,
6  .what-next-second,
7  .what-next-third {
8      float: left;
9      display: inline-block;
10     vertical-align: top;
11 }
12
13 .what-next-first,
14 .what-next-third {
15     width: 74%;
16     margin-left: 26%;
17     padding-top: 40px;
18 }
19
20 .what-next-second {
21     width: 100%;
22     margin-top: 20px;
23     padding: 0 30px;
24 }
25
26
27
28 @media all and (min-width: $breakpointSmall) {
29     .what-next-first,
30     .what-next-third {
31         width: auto;
32         margin-left: 8%;
33     }
34
35     .what-next-second {
36         float: right;
37     }
38 }
39
40 @media all and (min-width: $breakpointLarge) {
41     .what-next-first,
42     .what-next-second,
43     .what-next-third {
44         float: none;
45     }
46 }

```

Links ist das mobile Layout zu sehen, rechts mit den Styles die zutreffen ab der Bildschirmbreite \$breakpointLarge (960px).



## Drupal Contexts

Alle zuvor erstellten Views, bei denen als Anzeigeformat: Block ausgewählt wurde, sind nun auch als Drupal Blocks verfügbar. Es würde die Möglichkeit bestehen diese manuell einzeln für eine Region zu aktivieren, jedoch verliert man hierbei schnell die Übersicht.

Aus diesem Grunde habe ich mich dafür entschieden das Context Modul zu installieren. Hiermit können Kontexte erstellt werden für die bestimmte Bedingungen zutreffen müssen, wie zum Beispiel ein bestimmter URL-Pfad und dann können dort spezielle Aktionen ausgeführt werden, wie Blocks gerendert, CSS hinzugefügt, JavaScript hinzugefügt uvm.

Für die folgenden Seiten wurden Contexts erstellt, in welchen die darzustellenden Blöcke und CSS Definitionen festgelegt wurden:

- About Us
  - Condition: Pfad muss /about-us sein
  - Reactions: die gewünschten Blöcke wurden in der richtigen Reihenfolge in die Regionen hinzugefügt und zusätzlich wurde folgende CSS Datei hinzugefügt: `sites/all/themes/skateistan_theme/stylesheet/about_us.css`

The screenshot shows the 'Reactions' configuration for a context. On the left, there's a sidebar with 'Blocks' and 'CSS from Themes'. The main area is divided into regions: Header, Content Top, Content, Content Bottom, and Sidebar. Each region has a list of blocks with a weight and a checkbox to enable them. The 'Content' region has four blocks: about\_us\_0\_header, about\_us\_1\_infographics, about\_us\_2\_why skateboarding, and about\_us\_3\_whatnext. The 'Content Bottom' region has one block: about\_us\_s1\_miniramp. The 'Sidebar' region has two blocks: about\_us\_s2\_mission and about\_us\_s3\_quote.

Region	Block Name	Weight	Enabled
Header			
Content Top			
Content	about_us_0_header	X	
	about_us_1_infographics	X	
	about_us_2_why skateboarding	X	
	about_us_3_whatnext	X	
Content Bottom			
Sidebar	about_us_s1_miniramp	X	
	about_us_s2_mission	X	
	about_us_s3_quote	X	

Die Definitionen der anderen Seiten ist ähnlich wie About Us, nur mit anderen Pfaden / CS Dateien und Blöcken.

Nach Fertigstellung aller CSS / PHP Dateien wurde der gesamte Drupal Ordner und die Datenbank auf den Server übertragen.

## Schluss

Das Praktikum bei Skateistan war sehr lehrreich, gerade deshalb weil es etwas ganz anderes als im Studium war. Es war interessant im Arbeitsleben einer international agierenden Organisation mitzuwirken. Die Atmosphäre war sehr positiv und ich hatte immer einen Ansprechpartner. Ich dachte anfangs ich würde nicht so viel lernen, da es ein neuer Bereich für mich war und ich nicht sicher war ob alles so funktionieren würde - aber allen in allem konnte ich alles zufriedenstellend fertigstellen.

Das Praktikum fand komplett in englischer Sprache statt, was es noch interessanter machte. Außerdem waren nicht alle Personen mit denen ich Kontakt stand in einem Büro und ich musste von daher Gebrauch von unterschiedlichen Tools machen um mich auszutauschen. Diesen Prozess empfand ich als sehr positiv. Der Mix aus administrativen Tätigkeiten und Programmierung war gut.