HSR MAS Masterarbeit «AdminTools 2.0»

Abstrakt

Die Fachhochschule Nordwestschweiz (FHNW) betreibt neben ihrem eigentlichen Webauftritt www.fhnw.ch zusätzlich rund 400 weitere Webdienste von diversen Instituten, Projekten, etc.

Im Moment werden all diese Webdienste auf einem einzelnen Server gehostet. Dies ist aus Sicht Performance und Security ungünstig. Ein weiteres Problem stellt das Antragsformular dar; es läuft auf einer veralteten, nicht mehr unterstützen Technologie (Plone 3).

Die Client-Server Applikation «Admintools 2.0» stellt Benutzern und Administratoren neue, effiziente Werkzeuge basierend auf modernen Standards zur Verfügung. Ein Benutzer kann einen neuen Webdienst bestellen, der Administrator oder eine andere Bewilligungsinstanz kann ihn prüfen und freigeben. Für neu freigegebene Webdienste wird durch die Applikation automatisch ein eigenständiger Container erzeugt. Darin enthalten sind Webseiten oder CMS Vorlagen, welche dann durch die Benutzer individuell gestaltet werden können.

Die Frontendkomponente wurde in Angular 5.0 umgesetzt. Das Backend wurde mit Django und dem Django-Restframework implementiert. Neu bestellte Webdienste werden durch einen Bewilligungsworkflow im Frontend freigegeben und danach automatisiert als LXC Container (Virtualisierung) deployt und stehen dann den Benutzern direkt zu Verfügung.

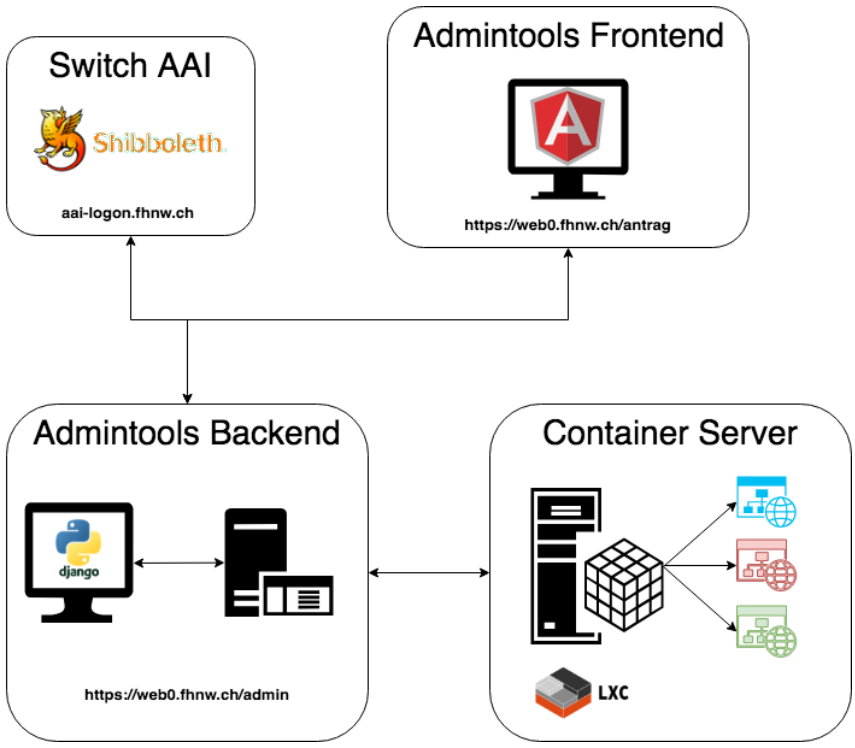


Abbildung 1: AdminTools 2.0 Aufbau

Masterarbeit - MAS Software Engineering
HSR Rapperswil 2016-2018

AdminTools 2.0

Anforderungsspezifikation

Team:
Arjan van Doesburg (vda)
Slavisa Karalic (ska)
Robin Ratcliff (rra)

Betreuer:
Daniel Tobler (dt)

Versionierungsgeschichte

Version	Author	Description	Date
0.1	vda	Initial struct	02.03.2018
0.2	vda	Adding Use cases	09.03.2018
0.3	vda	Update mocks, context, use cases, misc	10.03.2018
0.4	vda,rra, ska	Überarbeitung Feedback von Daniel Tobler	12.03.2018
0.5	vda	Formatierung, Cleanup	16.03.2018
0.6	vda, rra, ska	Einarbeitung Review-Findings	17.03.2018
1.0	vda, rra, ska	Einarbeitung Review-Findings	19.03.2018
1.1	vda, rra, ska	Überarbeitung Spezifikationen	10.05.2018

Inhaltsverzeichnis

1 Einführung

1.1 Ziel dieses Dokuments

Das Dokument soll die Anforderungen der Ziellösung beschreiben und eine Übersicht der involvierten Stellen/Personen und Systeme geben.

1.2 Referenzen

Name	Dokument
[Stakeholder]	MAS_HSR_Stakeholder.docx
[Glossar]	MAS_HSR_Glossar.docx

1.3 Stakeholder

Siehe [Stakeholder]

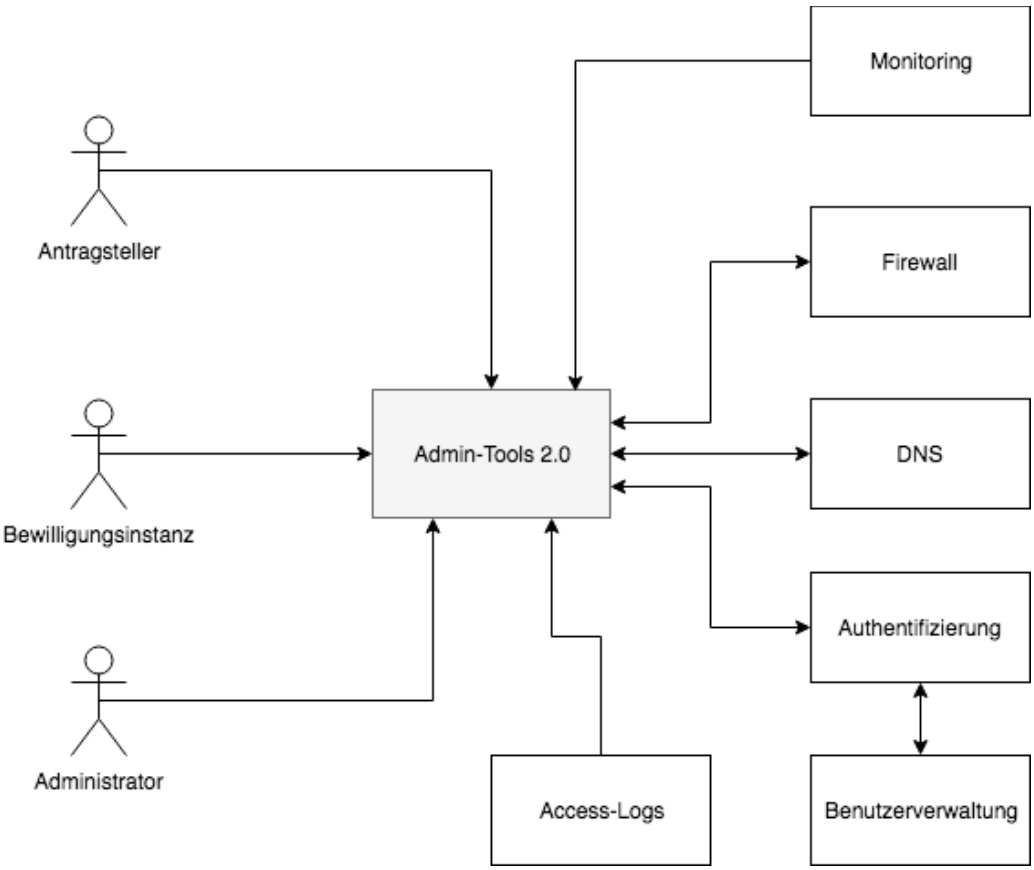
2 Projektbeschreibung

Die FHNW betreibt neben dem eigentlichen Webauftritt www.fhnw.ch auch noch rund 400 sogenannte Webdienste. Diese sind unter web.fhnw.ch/plattformen/ erreichbar.
Hier einige Beispiele solcher Webdienste:

- http://www.digitallernen.ch/
- https://web.fhnw.ch/plattformen/iobusiness/
- https://web.fhnw.ch/plattformen/softskills-sind-lernbar

Im Moment werden all diese Webdienste auf einem einzigen Server gehostet. Dies ist aus Sicht Performance und Security ungünstig. Ein weiteres Problem stellt das Antragsformular dar; es läuft auf einer veralteten, nicht mehr unterstützten Technologie (Plone 3).
Neu sollen die Anträge über ein zeitgemässes Webformular eingegeben werden können. Ein weiteres Anliegen ist die Sicherheit; ein einzelner kompromittierter Webdienst soll unter keinen Umständen einen Einfluss auf alle anderen haben. Auch soll neu gewährleistet werden, dass eine vertikale Skalierung möglich ist (Performance).
Um diesen Problemen zu begegnen, soll ein neuer POC (Proof of Concept) **AdminTools 2.0** erstellt werden, welcher den Benutzern und Administratoren neue, effiziente Werkzeuge (Antragsstrecke), basierend auf modernen Standards, zu Verfügung stellt (Angular / Django). Bei der Applikation handelt es sich aussschliesslich um Webapplikationen welche für den Desktop konzipiert sind.
Parallel zu dieser neuen Plattform wird in der FHNW ein anderes Projekt durchgeführt, welches sich um die Business Anforderungen der betriebenen Webdienste selbst kümmert. Dies beinhaltet primär Corporate Identity, Anwendungszwecke etc. und sind klar vom Lieferumfang der Masterarbeit getrennt.

2.1 Systemkontext



2.1.1 Legende

Komponente	Beschreibung
Antragsteller	Ein Kunde welcher einen Webdienst bestellen möchte
Admin-Tools 2.0	Das Ziel dieser Masterarbeit: Ein neuer Service für die FHNW
Firewall	Bestehende Hard und Software-Firewalls der FHNW
DNS	DNS Server der FHNW
Authentifizierung	SwitchAAI Authentication Service (Shibboleth)
Benutzerverwaltung	Microsoft Active Directory Service (LDAP)
Monitoring	Überwachungssoftware (Zabbix)
Access-Logs	Software um Zugriffsstatistiken für Webseiten zu erstellen (AwStats)
Administrator	Die Administratoren der Plattform Admin-Tools 2.0

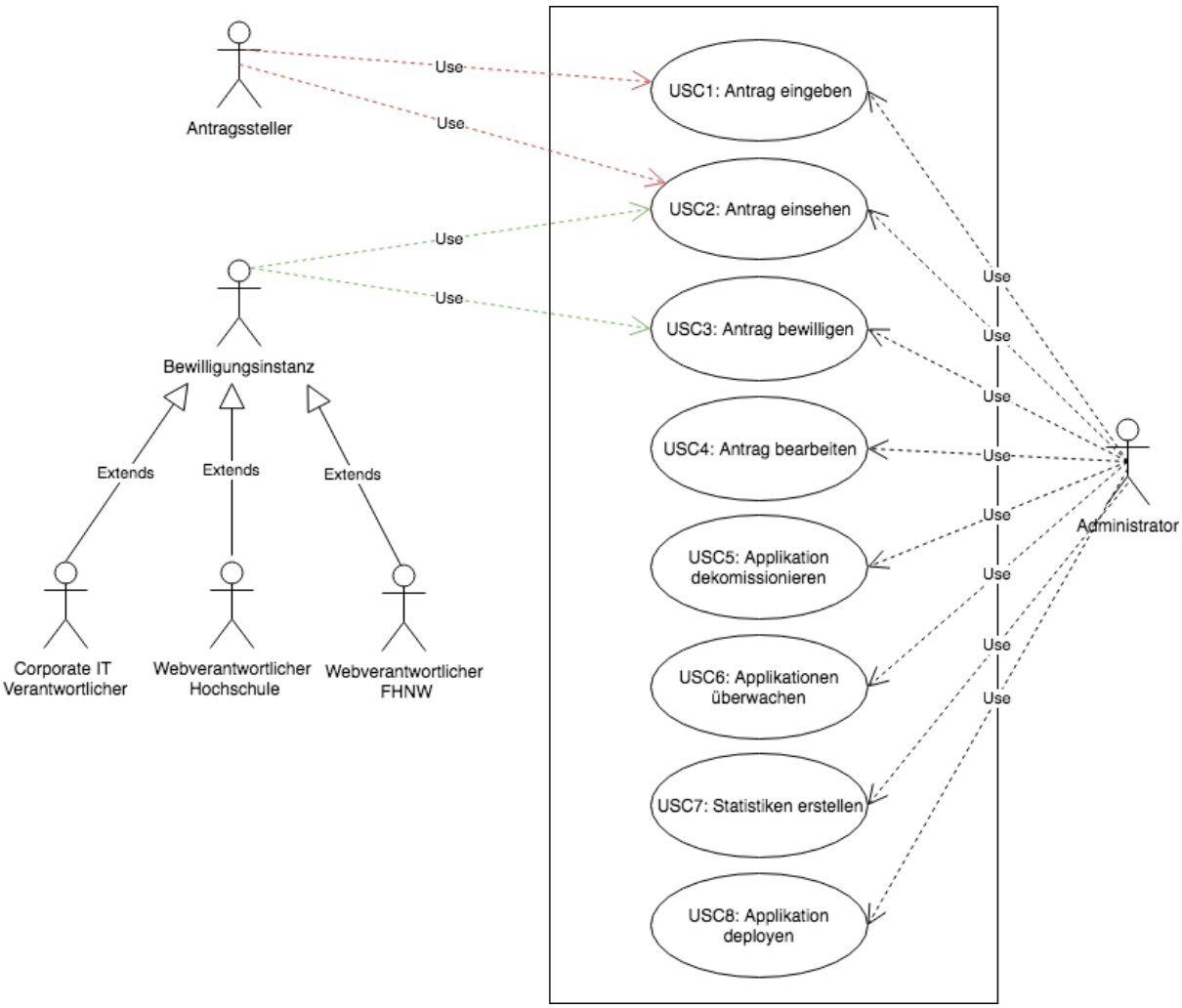
Bewilligungsinstanz	Zusammenfassung der 3 verschiedenen Instanzen welche einen neuen Webdienst bewilligen müssen (Webverantwortlicher, Kommunikationsverantwortlicher, Corporate IT)

3 Anforderungen

3.1 Akteure

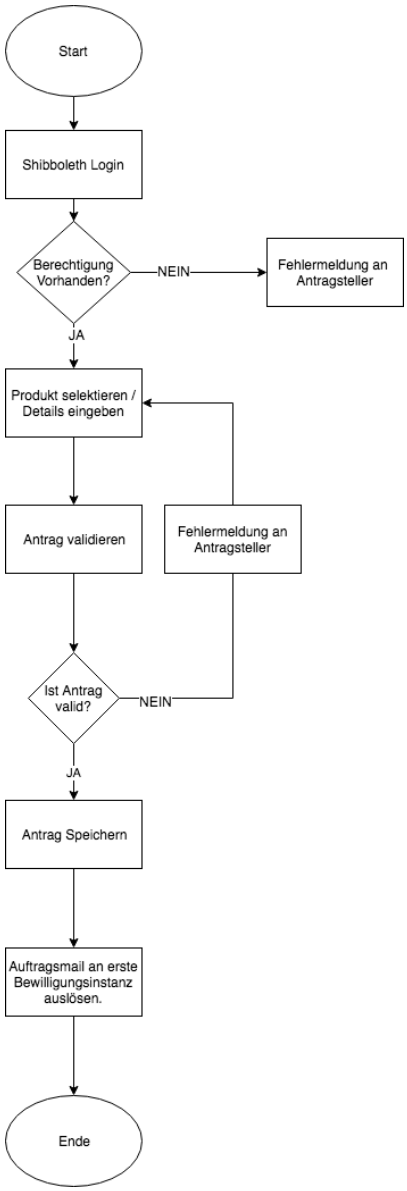
Name	Beschreibung
Antragssteller	Ein Mitarbeiter der FHNW welcher Anträge für Webdienste einreichen und den Antragsverlauf einsehen kann.
Bewilligungsinstanz: Webverantwortlicher der Hochschule	Eine Berechtigte Person der Hochschule/Organisationseinheit. Die erste Bewilligungsinstanz. Jeweils verantwortlich für Anträge aus der eigenen Hochschule (Bsp: Hochschule für Wirtschaft, Technik, Pädagogik usw.)
Bewilligungsinstanz: Kommunikationsverantwortlicher FHNW	Die zweite Bewilligungsinstanz. Verantwortlich für alle Organisationseinheiten der FHNW.
Bewilligungsinstanz: Corporate IT verantwortlicher	Die letzte Bewilligungsinstanz. Bewilligt den Antrag aus technischer Sicht.
Administrator	Löst das Deployment des Antrags/Artefaktes aus. Kann Anträge bearbeiten und die Plattform administrieren.

3.2 Anforderungsbeschreibungen



3.2.1 USC1: Antrag erstellen

Akteure	Antragsteller
Auslöser	Keine
Vorbedingung	Keine
Produkte	Im Rahmen von POC: • Webspace • CMS Webdienst • Dekomissionieren
Ablauf	1. Antragsteller meldet sich mit seinen Shibboleth Credentials auf der Plattform an. 2. Antragsteller selektiert gewünschtes Produkt. 3. Antragsteller gibt relevante Informationen zu selektiertem Produkt ein.

	4. Antragsteller bestätigt Eingaben. 5. Antrag wird Client- und Serverseitig validiert. 6. Antrag wird im System gespeichert und der ersten Bewilligungsinstanz wird ein Mail zugestellt in welchem diese dazu aufgefordert wird den Antrag zu prüfen.
Workflow	

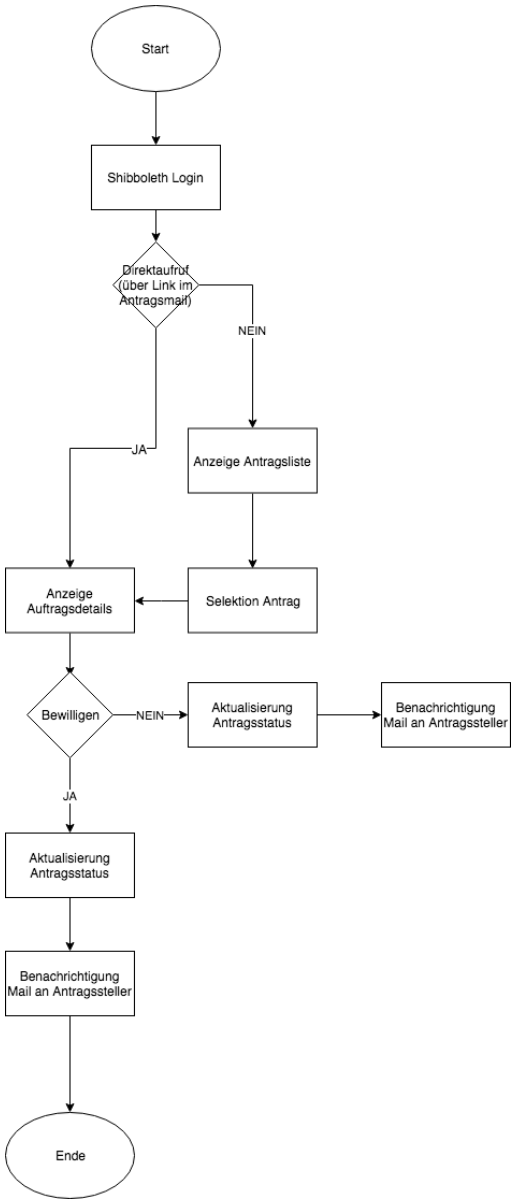
3.2.2 USC2: Antrag einsehen

Akteure	Webverantwortlicher Hochschule, Kommunikationsverantwortlicher der FHNW, Corporate IT Verantwortlicher, Antragssteller, Administrator
Auslöser	Kein
Vorbedingung	USC1

Ablauf	1. Antragsteller meldet sich mit seinen Shibboleth Credentials auf der Plattform an. 2. Antragsteller sieht die vom Antragsteller ausgefüllten Felder, sowie den aktuellen Bewilligungsstatus.

3.2.3 USC3: Antrag bewilligen

Akteure	Webverantwortlicher Hochschule, Kommunikationsverantwortlicher FHNW, Corporate IT Verantwortlicher, Administrator
Auslöser	Mail, Überprüfung via USC2 durch Webverantwortliche/Administratoren
Vorbedingung	USC1
Ablauf	1. Eine Bewilligungsinstanz meldet sich an. 2. Selektion des betroffenen Antrags. 3. Bewilligen des Antrags. 4. Status des Antrags wird aktualisiert (siehe “Antragsprozess Zustandsdiagramm”). 5. Mail an betroffene Instanz. Nächster Bewilliger wird per Mail benachrichtigt.
Alternativer Ablauf	1. Eine Bewilligungsinstanz meldet sich an. 2. Selektion des betroffenen Antrags. 3. Ablehnung des Antrags. 4. Status des Antrags wird aktualisiert (siehe “Antragsprozess Zustandsdiagramm”). 5. Mail an den Antragssteller.

Workflow	

3.2.4 USC4: Antrag bearbeiten

Akteure	Administrator, Antragsteller
Auslöser	Anfrage vom Antragsteller oder einer Bewilligungsinstanz
Vorbedingung	USC1, Antragsteller darf den Antrag nur bearbeiten, solange der Antrag noch von keiner Instanz bewilligt wurde.
Ablauf	1. Administrator meldet sich am System an. 2. Selektion des betroffenen Antrages. 3. Anpassung der Antragsdetails.

3.2.5 USC5: Applikation dekomissionieren

Akteure	Administrator
Auslöser	• Anfrage von Antragsteller via Antragsformular
Vorbedingung	• USC8 • USC1 mit Auswahl “Applikation Löschen” • USC3
Ablauf	1. Selektion der Applikation. 2. Auslösen des Löschprozesses 3. Mail an Antragssteller.

3.2.6 USC6: Applikationen überwachen

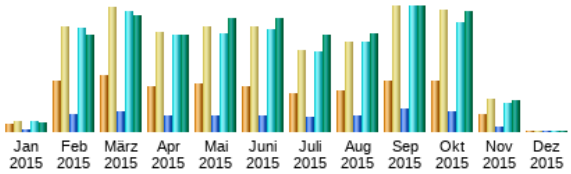
Akteure	Administrator
Auslöser	Applikation wurde deployed
Vorbedingung	USC8
Ablauf	1. Zabbix (externe Monitoring Lösung) prüft im 5 Minuten Intervall ob die Applikation(en) erreichbar ist/sind und misst Performancedaten. 2. Aktualisierung der Zabbix-Datenbank mit den Resultaten. 3. Falls die Applikation länger als 15min nicht erreichbar ist wird ein Mail an die Administratoren verschickt.

3.2.7 USC7: Statistiken erstellen/auswerten

Akteure	System
Auslöser	Applikation wurde deployed
Vorbedingung	USC8
Ablauf	1. AwStats (externes Logfile-Analyse Tool) erstellt anhand von Logs der deployten Applikation Zugriffsstatistiken. 2. AwStats visualisiert Resultate auf eigenem Dashboard.

Monatliche Historie



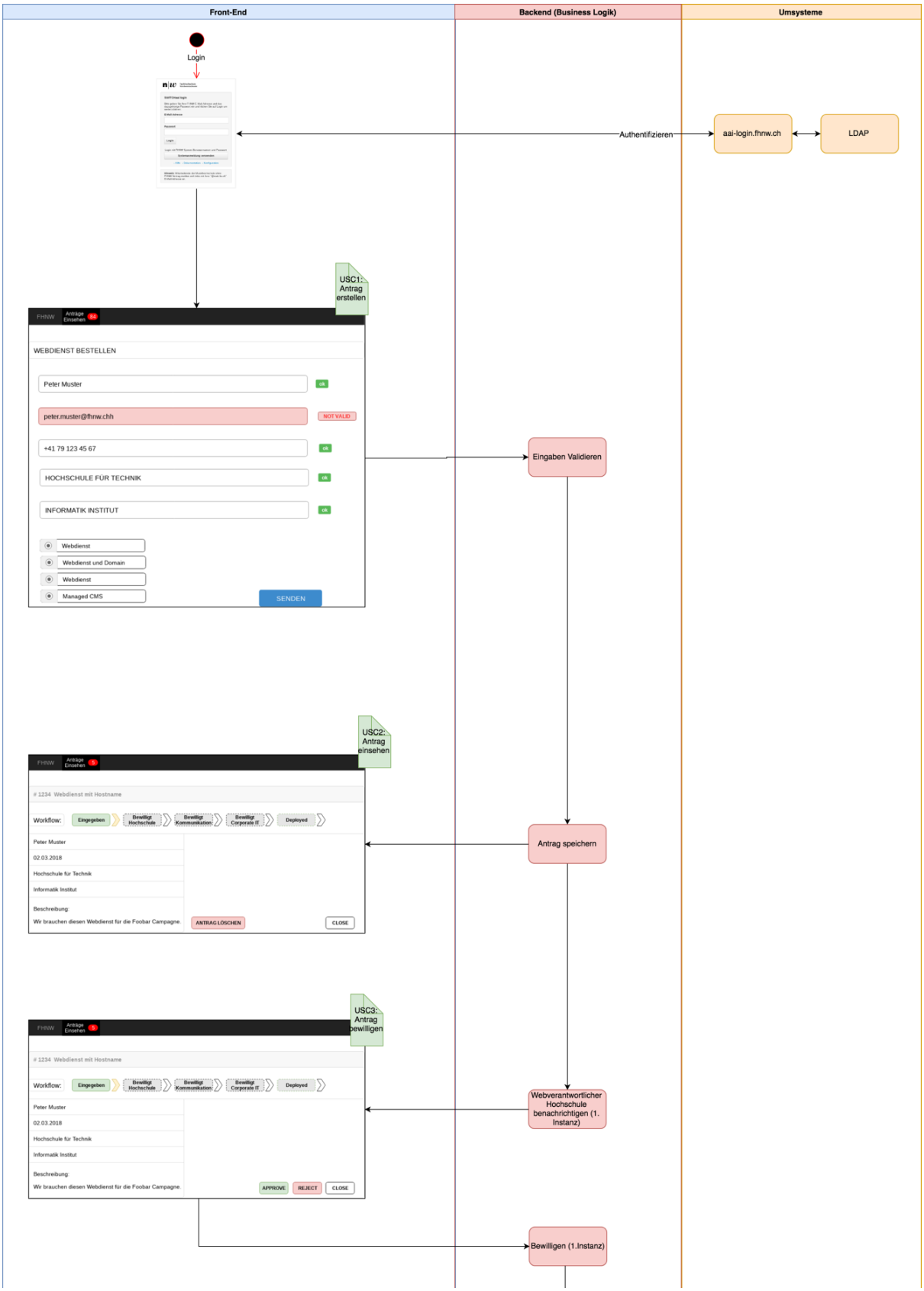
Monat	Unterschiedliche Besucher	Anzahl der Besuche	Seiten	Zugriffe	Bytes
Jan 2015	25,400	35,765	260,970	1,574,873	51.51 GB
Feb 2015	175,315	357,966	2,882,708	16,760,184	537.50 GB
März 2015	191,623	425,966	3,231,048	19,392,774	646.46 GB
Apr 2015	157,216	339,878	2,642,466	15,536,214	537.08 GB
Mai 2015	163,303	359,666	2,673,252	15,843,846	627.79 GB
Juni 2015	156,818	362,576	2,692,309	16,439,967	629.91 GB
Juli 2015	130,659	280,401	2,280,658	12,949,578	535.00 GB
Aug 2015	138,708	309,966	2,612,255	14,600,092	545.33 GB
Sep 2015	174,498	429,458	3,686,548	20,237,038	696.43 GB
Okt 2015	173,164	415,860	3,216,118	17,655,564	670.65 GB
Nov 2015	61,518	110,482	836,672	4,528,492	175.23 GB
Dez 2015	0	0	0	0	0
Total	1,548,222	3,427,984	27,015,004	155,518,622	5652.87 GB

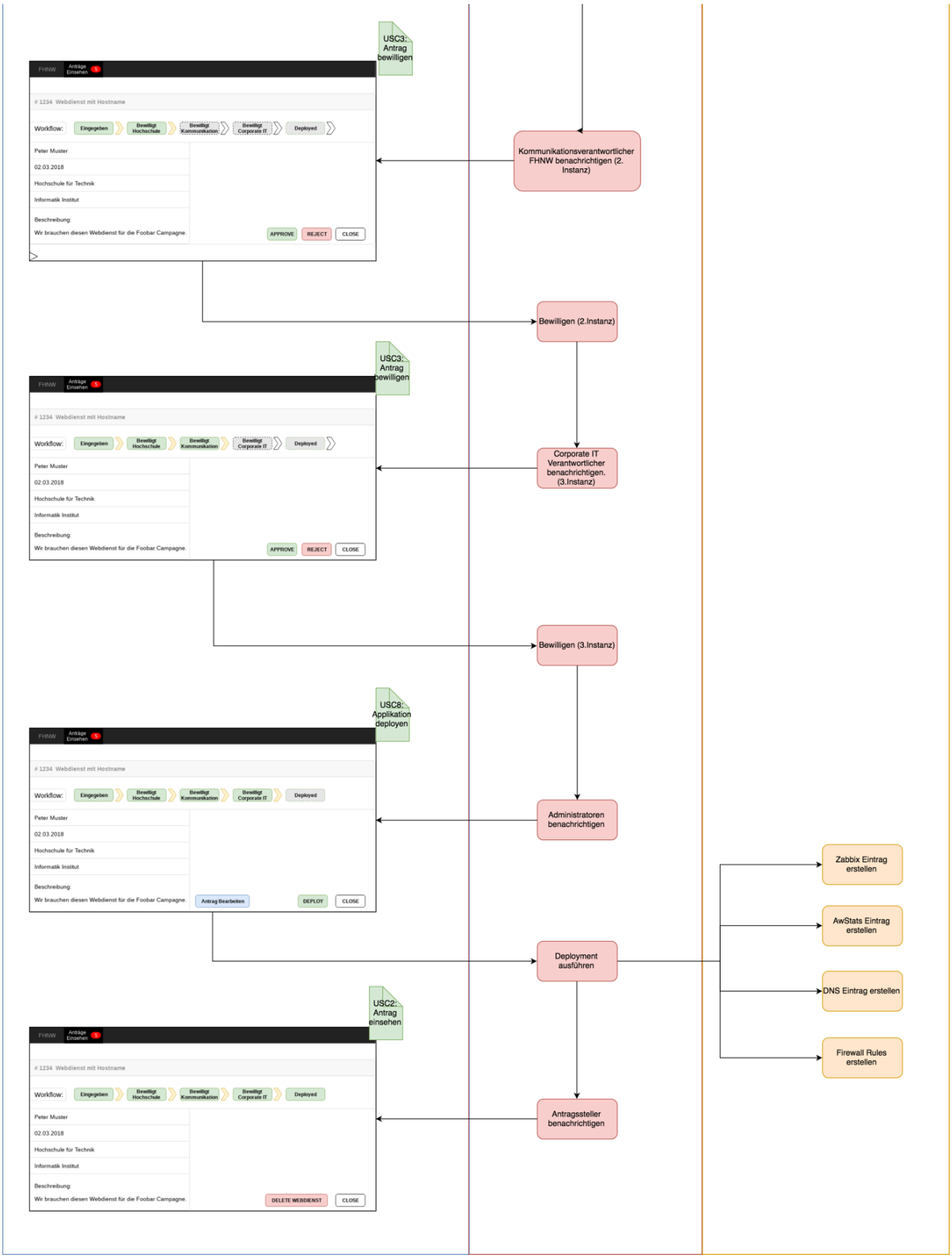
3.2.8 USC8: Applikation deployen

Beschreibung	Löst das effektive deployment aus
Akteure	Administrator
Auslöser	Administrator
Vorbedingung	USC1 und USC3
Ablauf	1. Selektion des bewilligten Antrags. 2. Auslösen des Deployments. 3. Eintrag konfigurieren in Zabbix für neue Applikation 4. Eintrag konfigurieren in AwStats für neue Applikation.

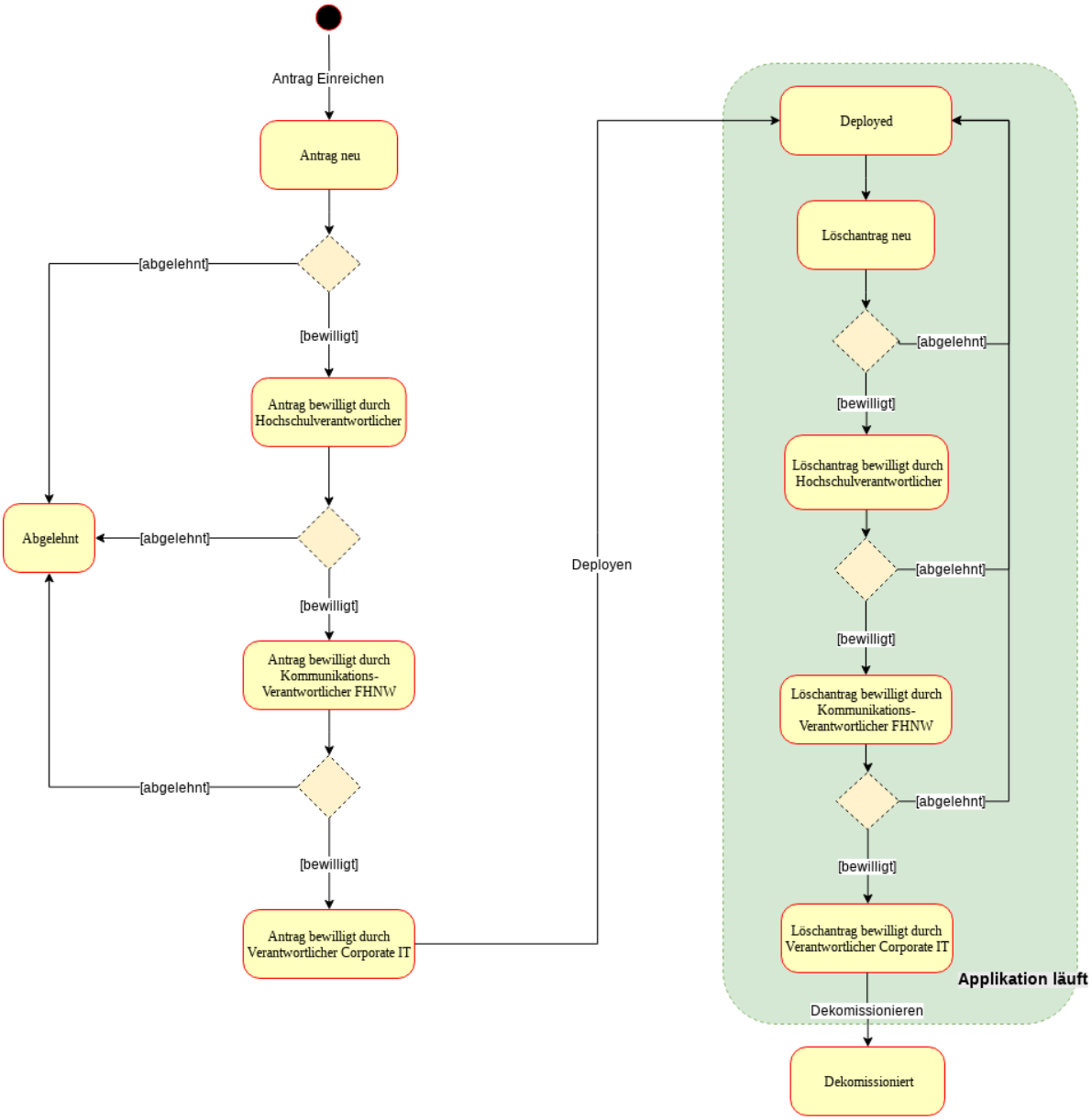
3.3 Hauptanwendungsfall

Eine Visuelles Storyboard welches den Hauptanwendungsfall (USC1) inklusive Mocks der zu entwickelnden Webformulare aufzeichnet.




USC3: Antrag bewilligen
Kommunikationsverantwortlicher FHNW benachrichtigen (2. Instanz)
Bewilligen (2.Instanz)
USC3: Antrag bewilligen
Corporate IT Verantwortlicher benachrichtigen. (3.Instanz)
Bewilligen (3.Instanz)
USC8: Applikation deployen
Administratoren benachrichtigen
Zabbix Eintrag erstellen
AwStats Eintrag erstellen
DNS Eintrag erstellen
Firewall Rules erstellen
Deployment ausführen
USC2: Antrag einsehen
Antragssteller benachrichtigen
FHNW
Anträge Einsehen
1234 Webdienst mit Hostname
Workflow:
Eingegeben
Bewilligt Hochschule
Bewilligt Kommunikation
Bewilligt Corporate IT
Deployed
Peter Muster
02.03.2018
Hochschule für Technik
Informatik Institut
Beschreibung:
Wir brauchen diesen Webdienst für die Foobar Campagne.
APPROVE
REJECT
CLOSE
Antrag Bearbeiten
DEPLOY
DELETE WEBDIENST

3.4 Antragsprozess Zustandsdiagramm



4 Nicht funktionale Anforderungen

4.1 NFA01 Verfügbarkeit

Die Webapplikationen sollen so designt sein, dass sie im Falle einer Produktivsetzung stabil, mit möglichst wenig Wartungsaufwand funktionieren. Es gibt an der FHNW keine SLA. Es gilt “Best Effort”.

4.2 NFA02 Erweiterbarkeit

Die Architektur soll modular und offen gestaltet werden. Hintergrund und Ziel dieser Anforderung ist, dass die Architektur um weitere (heute noch nicht klar definierte) Technologien erweiterbar sein soll, ohne die Architektur von Grund auf neu gestalten zu müssen.

4.3 NFA03 Benutzbarkeit

Trennung Frontend Backend. Hauptziel ist bessere Usability für Antragsteller. Es soll zu jedem Zeitpunkt klar sein, welchen Status der Antrag hat.

4.4 NFA04 Wartbarkeit.

Es müssen Technologien gewählt werden, die im Stack der FHNW oder kompatibel sind (Programmiersprachen, Betriebssysteme, Methoden).

4.5 NFA05 Skalierbarkeit

Verwendung von Containern oder ähnlicher Technologie welche es den Administratoren erlaubt während laufendem Betrieb die Ressourcen (Arbeitsspeicher, Filesystem-Grösse) zu erweitern.

Masterarbeit - MAS Software Engineering
HSR Rapperswil 2016-2018

AdminTools 2.0

Software Development Plan

Team:
Arjan van Doesburg (vda)
Slavisa Karalic (ska)
Robin Ratcliff (rra)

Betreuer:
Daniel Tobler (dt)

Inhaltsverzeichnis

Versionierungsgeschichte

Version	Author	Beschreibung	Datum
0.1	vda	Inital Setup	16.02.2018
0.2	rra, ska	Inhalte und Kapitel	12.03.2018
0.3	vda	Formatierung, Review Inhalt	16.03.2018
0.4	vda, rra, ska	Einarbeitung Review-Findings	17.03.2018
1.0	vda, rra, ska	Einarbeitung Review-Findings	19.03.2018
1.1	vda, rra, ska	Überarbeitung	10.05.2018
1.2	Vda, rra, ska	Überarbeitung für Abgabe (Meilensteine)	23.08.2018

1 Einführung

1.1 Zweck

Dieses Dokument beschreibt die Organisation, Vorgehensweise und die Verwendeten Werkzeuge für die Umsetzung des Projekts "Webdienst Administrationsplattform der FHNW (**Admintools 2.0**)".

1.2 Gültigkeitsbereich

Dieses Dokument ist über die ganze Projektdauer gültig. Änderungen werden fortlaufend ergänzt und im Changelog festgehalten.

1.3 Referenzen

Name	Dokument
Stakeholder	MAS_HSR_Stakeholder.docx
Glossar	MAS_HSR_Glossar.docx

2 Projektübersicht

Die FHNW betreibt neben dem eigentlichen Webauftritt www.fhnw.ch auch noch rund 400 sogenannte Webdienste. Diese sind unter web.fhnw.ch/plattformen/ erreichbar.
Hier einige Beispiele solcher Webdienste:

- http://www.digitallernen.ch/
- https://web.fhnw.ch/plattformen/iobusiness/
- https://web.fhnw.ch/plattformen/softskills-sind-lernbar/

Im Moment werden all diese Webdienste auf einem einzigen Server gehostet. Dies ist aus Sicht Performance und Security ungünstig. Ein weiteres Problem stellt das Antragsformular dar; es läuft auf einer veralteten, nicht mehr unterstützten Technologie (Plone 3).
Neu sollen die Anträge über ein zeitgemässes Webformular eingegeben werden können. Ein weiteres Anliegen ist die Sicherheit; ein einzelner kompromittierter Webdienst soll unter keinen Umständen einen Einfluss auf alle anderen haben. Auch soll neu gewährleistet werden, dass eine vertikale Skalierung möglich ist (Performance).
Um diesen Problemen zu begegnen, soll ein neuer POC (Proof of Concept) **AdminTools 2.0** erstellt werden, welcher den Benutzern und Administratoren neue, effiziente Werkzeuge (Antragsstrecke), basierend auf modernen Standards, zu Verfügung stellt (Angular / Django). Bei der Applikation handelt es sich aussschliesslich um Webapplikationen welche für den Desktop konzipiert sind.
Parallel zu dieser neuen Plattform wird in der FHNW ein anderes Projekt durchgeführt, welches sich um die Business Anforderungen der betriebenen Webdienste selbst kümmert. Dies beinhaltet primär Corporate Identity, Anwendungszwecke etc. und sind klar vom Lieferumfang der Masterarbeit getrennt.

2.1 Lieferumfang

Ziel ist es einen lauffähigen POC (proof of concept) zu erstellen. Auf diesem soll es möglich sein, mittels einer Antragsstrecke, einen einzelnen neuen Webdienst zu bestellen.
Der Webdienst soll dann auch automatisch deployed und gestartet werden.
Ein weiterer Teil der Arbeit ist die Evaluation der zu verwenden Technologien (Docker, LXC, etc.). Im Rahmen dieser Arbeit wird eine Technologie gewählt und mit Hilfe dieser der Prototyp erstellt.

3 Projektorganisation

3.1 Projektmitglieder

Name	Rolle	Kontakt	Abwesenheiten
Arjan van Doesburg	Entwickler		
Robin Ratcliff	Entwickler		
Slavisa Karalic	Entwickler		
Daniel Tobler	Betreuer		14.7 - 9.8.2018
Tom Schneider	Productowner		

3.2 Zeitmanagement / Kollaboration

- Das Projekt arbeitet in einem agilen Prozess, angelehnt an Scrum und RUP.
- Die Entwickler arbeiten jede Woche Montag/Samstag und nach Bedarf an zusätzlichen Randstunden/Tagen.
- Die Arbeiten finden vor Ort im FHNW Campus Brugg/Windisch statt und nach Bedarf auch via Remote-Zugriff von zuhause aus. Zur Kommunikation werden Tools wie Slack, OneDrive etc. verwendet. Siehe Kapitel: 4.2 Entwicklungsumgebung, Tools
- Es werden an jedem Arbeitstag morgens um 09:30 Stand-Ups zu 15min durchgeführt um die aktuelle Projektlage/Arbeitsleistung zu besprechen.
- Der Productowner wird jeweils am Montag am Stand-Up teilnehmen (ausdrücklich so erwünscht worden). Die Resultate der Stand-Ups werden in Gitlab entsprechend nachgeführt. Zusätzliche Protokolle sind nicht vorgesehen.
- Das Treffen mit dem Betreuer findet im 3 Wochen-Rhythmus statt. Zusätzlich sind vereinzelte Treffen auf dem Campus Brugg-Windisch der FHNW geplant.

3.3 Sprintplanung

- Die individuellen Sprints haben eine Länge von 3 Wochen (à je 2 Arbeitstage/Woche)
- Es ergeben sich ein Total aus 8 Sprints beginnend am 19.03.2018 bis zum 27.08.2018
- Sprint-Review findet jeweils am Ende jedes Sprints zusammen mit dem Betreuer dem Productowner, Projektleiter sowie Webmaster statt.

3.4 Meilensteine

Datum	Meilenstein	Lieferergebnisse
06.01.2018	Einreichung Antrag Masterarbeit	Unterzeichnetes Antragsdokument
23.02.2018	Erstes Treffen Betreuer	Terminvereinbarungen, Review Anforderungen, Initiale Projektplanung
19.03.2018	Einreichen Dokumente für Projektreview	Anforderungsspezifikation, Software Development Plan
Voraussichtlich Sprint	Technischer Durchstich	Erster funktionaler Prototyp mit: • Bestellformular • Dummy-Container (Webdienst) • Deploy-Prozess • Infrastruktur Es soll möglich sein via das Bestellformular ein Webdienst zu deployen
03.04.2018	Review Masterarbeiten	Projektreview durch ein anderes Projektteam und Dozenten
04.04.2018	Sprint 1 (Technischer Durchstich 1)	Vorstellung FHNW mit Productowner, Vorgesetzten. Review Projektziele.
24.04.2018	Sprint 2 (Technischer Durchstich 2)	Vorstellung Prototyp
22.05.2018	Sprint 3	Authentifizierung, Domainmodell, Dokumentation überarbeiten.
15.06.2018	Sprint 4	OpenShift Switch.ch Container & Angular Forms
10.07.2018	Sprint 5	Security, Webforms, Container-Execution

12.08.2018	Sprint 6 (Endspurt)	Abgabe Abstrakt an Betreuer, Neue Container, Finalisierung Applikation
27.08.2018	Finale Abgabe Masterarbeit	Abgabe Masterarbeit
03.09.2018	Präsentation Masterarbeit	Demo und Q&A Masterarbeit vor Publikum

4 Infrastruktur

4.1 Server Infrastruktur

Die im Projekt benutzten Server sind von der FHNW zur Verfügung gestellt worden. Die Server werden für die Entwicklung benutzt und gehen später direkt in produktiven Betrieb über. Es handelt sich um virtuelle Server. Betriebssystem ist Red Hat Enterprise Linux 7

Domain	Betriebssystem	Funktion
web0.fhnw.ch	RHEL 7	Proxy Server
v000153.adm.ds.fhnw.ch	RHEL 7	Applikationsserver
v000154.adm.ds.fhnw.ch	RHEL 7	SQL DB Server
v000155.adm.ds.fhnw.ch	Ubuntu 16.4 LTS	Container Server
Gitlab-runner-admin-tools	CentOS 7.	Git-Lab Runner (Testing)

4.2 Entwicklungsumgebung, Tools

Bereich	Werkzeug	Version
Git Versionierung	gitlab.fhnw.ch	10.4
CI/CD	Jenkins	2.109
Backlog/Task Management	gitlab.fhnw.ch	10.4
Kommunikation	Slack https://hsrmas2016.slack.com/, Skype (Conference Calls)	
IDE	Intellij, PyCharm	2017.3.5
Fileablage	OneDrive https://onedrive.live.com/	-
Dokumentation/Grafiken	MS Word, Draw.io, MarkDown	-

5 Qualitätsmassnahmen

Beschreibt die Vorgehensweisen und technischen Mittel um über die gesamte Projektlauzeit eine hohe Qualität sicherzustellen.

5.1 Dokumentation

Technische Dokumentation wird in Form von README Dateien direkt in Git versioniert.
Die Formatierung der README Dateien entspricht MarkDown. Dieses Vorgehen ist in der FHNW Standard und wird in Gitlab benutzerfreundlich unterstützt.

Eine Userdokumentation (Betriebshandbuch) wird in der offiziellen Dokumentationsplattform der FHNW (http://help.fhnw.ch) erstellt.

Ein separates Wiki innerhalb von Gitlab.
https://gitlab.fhnw.ch/hsrmas/tasks/wikis/home

5.2 Entwicklung

5.2.1 Versionsverwaltung

Als Versionsverwaltungstool kommt bei uns Git zum Einsatz.

5.2.2 Verwendete Technologien

Name	Version	Beschreibung
Angular	5.0	Angular.io
RxJs	5.5.6	reactivex.io
Django	2.0	Djangoproject.org
Django-Rest	3.7.7	Django-rest-framework.org
Nginx	1.12	Nginx.com
Apache httpd	2.4	Httpd.apache.org
Shibboleth	2	Shibboleth.net, switch.ch
LXC		
LXC Rest		
LXC Web		
OpenShift Origin		Redhat.com

5.2.3 Aufgaben-Tracking

Zur Nachverfolgung der Aufgaben wird das Gitlab Dashboard verwendet.

5.2.4 Status Definitionen

Definition oft Done (DOD)
Eine Aufgabe (Task) gilt als abgeschlossen, wenn folgende Kriterien erfüllt sind:

- Task ist in der Versionsverwaltung eingecheckt.
- Merge-Request ist abgeschlossen im 4 Augen Prinzip.

- Es gibt keine fehlschlagenden Unit-Tests.
- Changelog Eintrag nachgeführt.

5.2.5 Code Style Guidelines

- Python - PEP 8: https://www.python.org/dev/peps/pep-0008/
- TypeScript: https://www.npmjs.com/package/tslint
- Bei allen anderen eingesetzten Frameworks, soweit als möglich, die jeweiligen Style Guidelines berücksichtigen.

5.4 Testen / Code Reviews / Code Coverage

- Code Coverage in Prozent darf nur konstant bleiben oder zunehmen. Die Gitlab Plattform wird so eingestellt, dass bei einem Rückgang der Coverage dem Entwicklungsteam eine Warnung per Mail verschickt wird.
- Gitlab Runner wird bei jedem jeden Commit ausgeführt und prüft auf fehlerfreie Unit-Tests und die Einhaltung der Coding Styleguides.
- Als CI/CD Umgebung wird ebenfalls GitLab-Runner verwendet. Builds werden in der Entwicklungsumgebung automatisch getriggert bei commits.
- Ein 4 Augen Prinzip bei den Merge-Requests
- Bei komplexeren Aufgaben werden wir falls möglich Pair-Programming einsetzen.

Masterarbeit - MAS Software Engineering
HSR Rapperswil 2016-2018

Architektur & Design

Team:
Arjan van Doesburg (vda)
Slavisa Karalic (ska)
Robin Ratcliff (rra)

Betreuer:
Daniel Tobler (dt)

Inhaltsverzeichnis

1 Projektbeschreibung

Die FHNW betreibt neben dem eigentlichen Webauftritt www.fhnw.ch auch noch rund 400 sogenannte Webdienste. Diese sind unter web.fhnw.ch/plattformen/ erreichbar.
Hier einige Beispiele solcher Webdienste:

- http://www.digitallernen.ch/
- https://web.fhnw.ch/plattformen/iobusiness/
- https://web.fhnw.ch/plattformen/softskills-sind-lernbar

Im Moment werden all diese Webdienste auf einem einzigen Server gehostet. Dies ist aus Sicht Performance und Security ungünstig. Ein weiteres Problem stellt das Antragsformular dar; es läuft auf einer veralteten, nicht mehr unterstützten Technologie (Plone 3).

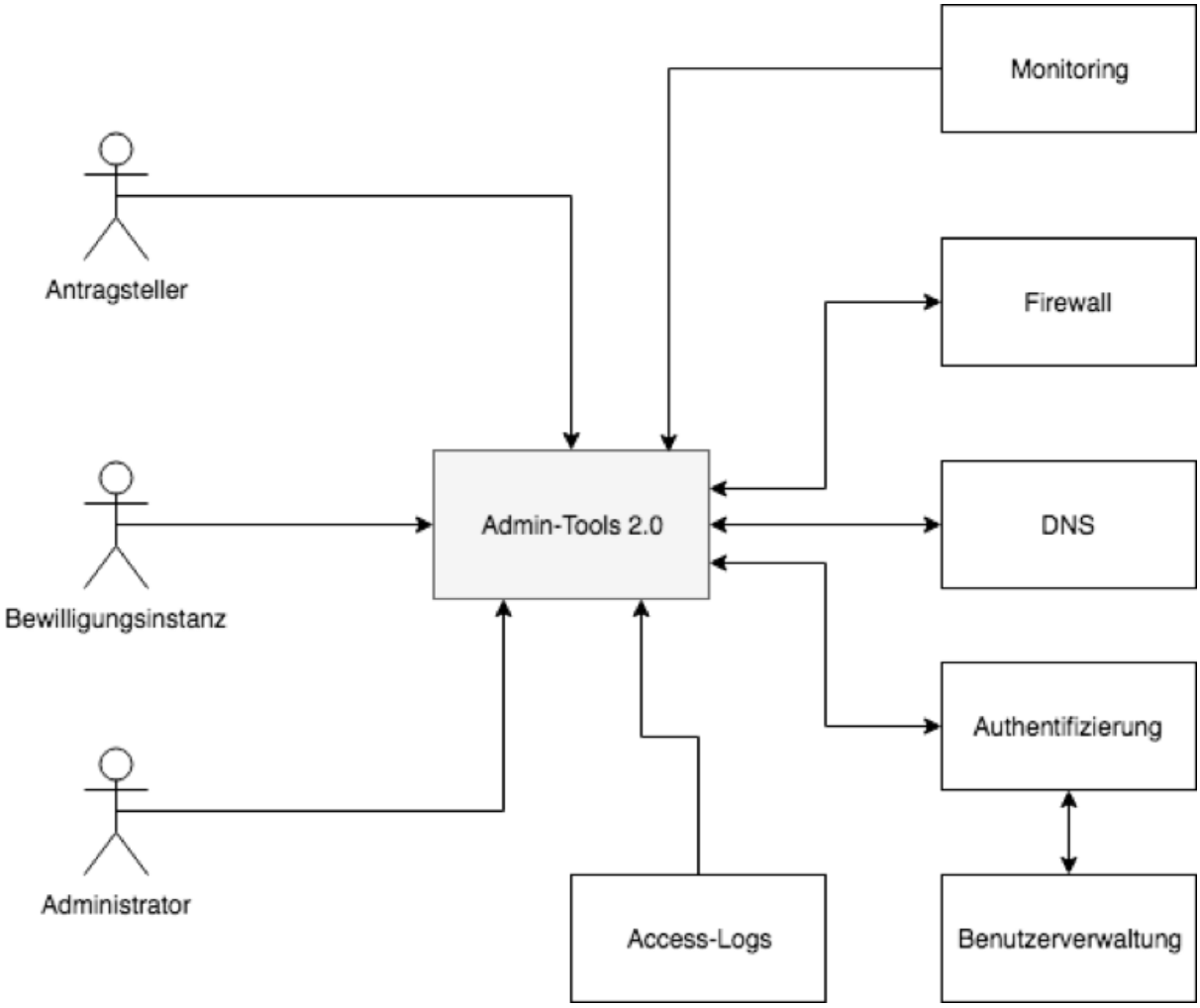
Neu sollen die Anträge über ein zeitgemässes Webformular eingegeben werden können. Ein weiteres Anliegen ist die Sicherheit; ein einzelner kompromittierter Webdienst soll unter keinen Umständen einen Einfluss auf alle anderen haben. Auch soll neu gewährleistet werden, dass eine vertikale Skalierung möglich ist (Performance).

Um diesen Problemen zu begegnen, wurde ein neuer POC (Proof of Concept) AdminTools 2.0 erstellt, welcher den Benutzern und Administratoren neue, effiziente Werkzeuge (Antragsstrecke), basierend auf modernen Standards, zu Verfügung stellt (Angular / Django). Bei der Applikation handelt es sich aussschliesslich um Webapplikationen welche für den Desktop konzipiert sind.
Parallel zu dieser neuen Plattform wird in der FHNW ein anderes Projekt durchgeführt, welches sich um die Business Anforderungen der betriebenen Webdienste selbst kümmert. Dies beinhaltet primär Corporate Identity, Anwendungszwecke etc. und sind klar vom Lieferumfang der Masterarbeit getrennt.

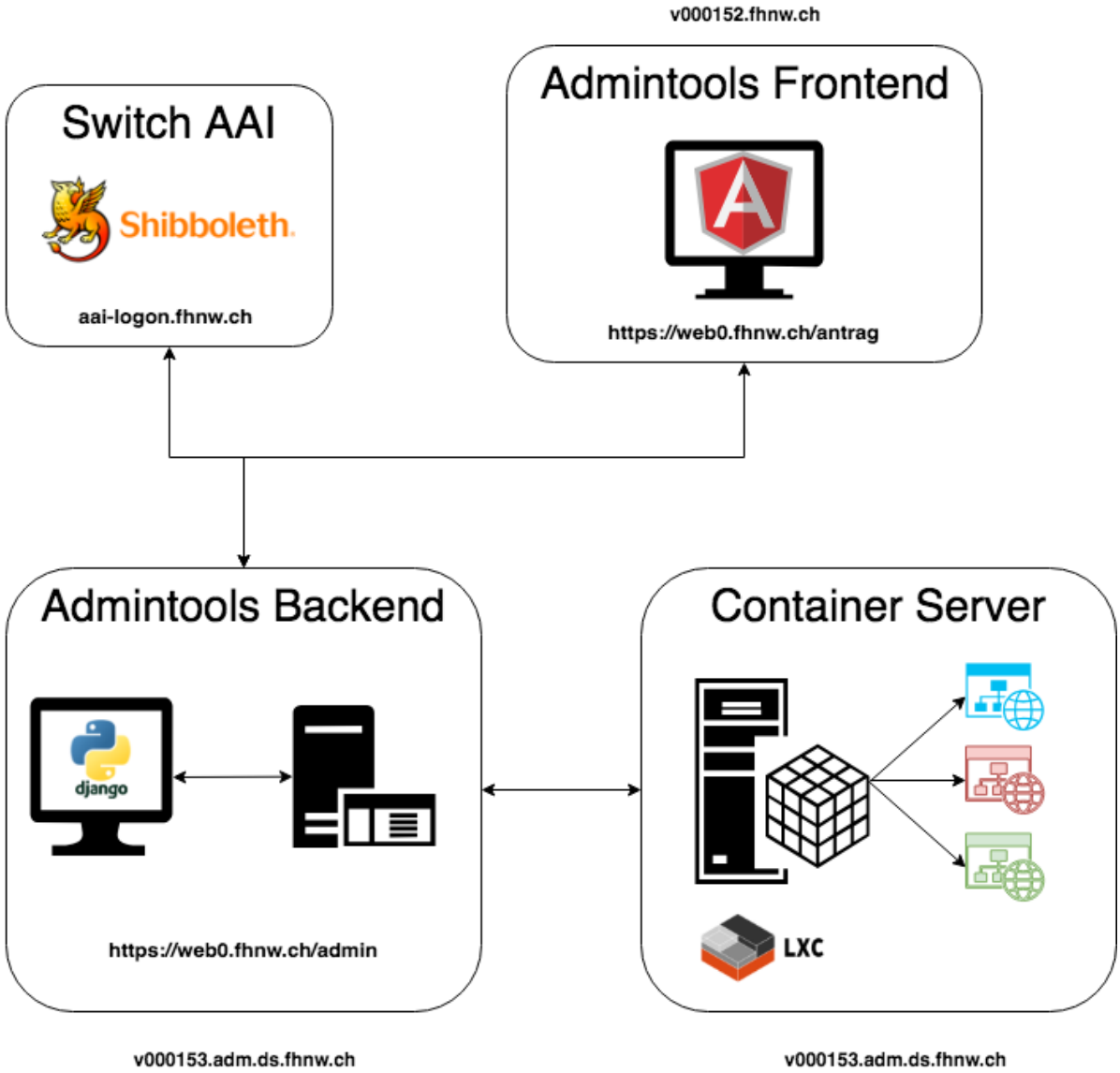
1.1 Stakeholder

Name	Organisation	Rolle
Daniel Tobler	HSR	Betreuer
Luc Bläser	HSR	MAS Studiengangsleiter
Robin Ratcliff	FHNW	Entwickler (Team)
Slavisa Karalic	FHNW	Entwickler (Team)
Arjan van Doesburg	Allianz	Entwickler (Team)
Tom Schneider	FHNW	Teamleiter Webteam FHNW/ Productowner
Roland von Aesch	FHNW	Webmaster FHNW
Jan Rothenberger	FHNW	Projektleiter Webdienste Relaunch

2 System Kontext



3 Systemübersicht



3.1 Übersicht Server und Betriebssysteme

- web0.fhnw.ch (v000152.fhnw.ch) DMZ
  - Server Virtuell - FHNW
  - RHEL 7
  - 4 CPU
  - 8 GB RAM
  - 100 GB

- v000153.adm.fhnw.ch Django Applikationsserver
  - Server Virtuell - FHNW
  - RHEL 7
  - 2 CPU
  - 4 GB RAM
  - 100 GB

- v000154.adm.fhnw.ch - DB Server out of scope

- v000155.adm.fhnw.ch - LXC Server
  - Server Virtuell - FHNW
  - Ubuntu 16.04 LTS
  - 4 CPU
  - 16 GB RAM
  - 100 GB

3.2 Übersicht Dienste und Applikationen

web0.fhnw.ch

Proxy Server, befindet sich im DMZ (von Aussen erreichbar)

- Angular Frontend Applikation Deployment; erreichbar unter https://web0.fhnw.ch/antrag
- Hookreceiver - Python Applikation, die Commit Hooks von Gitlab empfängt und ein automatisches Deployment der Frontend Applikation durchführt.
- Nginx - Haupt Webserver. Liefert die Frontend Applikation direkt aus; ausserdem fungiert der Nginx als Proxy Server: Proxy zur admintools applikation und allen bereits bereitgestellten Webdiensten: Bsp: web0.fhnw.ch/webdienst
- Apache - Webserver wird benutzt um hinter nginx eine Shibboleth (Switch AAI) Integration zu gewährleisten

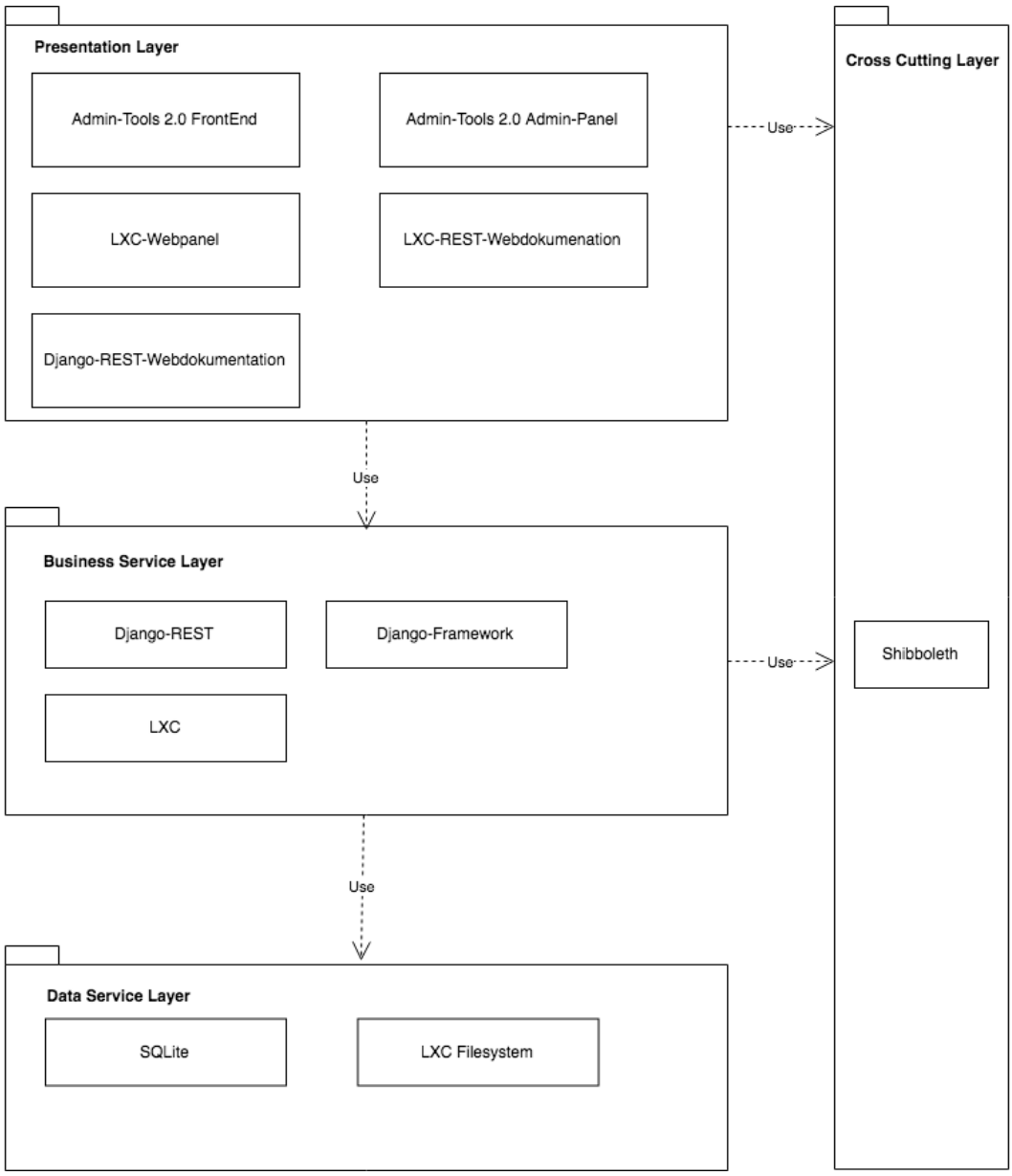
v000153.adm.ds.fhnw.ch

- Admintools 2.0 - Die Haupt Applikation implementiert mit Django Framework; erreichbar unter https://web0.fhnw.ch/admin
- nginx - Liefert statische Inhalte der Applikation, dienst als Proxy für die dynamischen Inhalte, die vom Gunicorn Daemon geliefert werden.
- Gunicorn - Python WSGI HTTP Server: liefert die dynamischen Inhalte der Django Applikation (2 Worker)

v000155.adm.ds.fhnw.ch

- LXC Daemon : Steuert alle benötigten Prozesse, die für LXC notwendig sind (lxc-containers, lxc-net, lxc-apparmor etc.)
- LXC Webpanel : eine Python Flask Webapplikation, die ein rudimentäres UI für viele LXC Tools zur Verfügung stellt. Abgesehen von der Steuerung der Container, kann sie auch als ein simples Monitoring verwendet werden. Erreichbar unter http://10.51.6.78/ im internen FHNW Netz.
- LXC REST : Rest Schnittstelle zum LXC. Erreichbar unter http://v000155.adm.ds.fhnw.ch/doc/im internen FHNW Netz
- Apache : Läuft auf Port 80 (http) Liefert die bieden LXC Applicationen (Webpanel & REST) aus.
- Nginx : Läuft auf Port 443 (https) Dienst als Zwischenproxy. Anfragen von web0.fhnw.ch leitet dieser nginx weiter auf die laufenden Webserver innerhalb der Container.

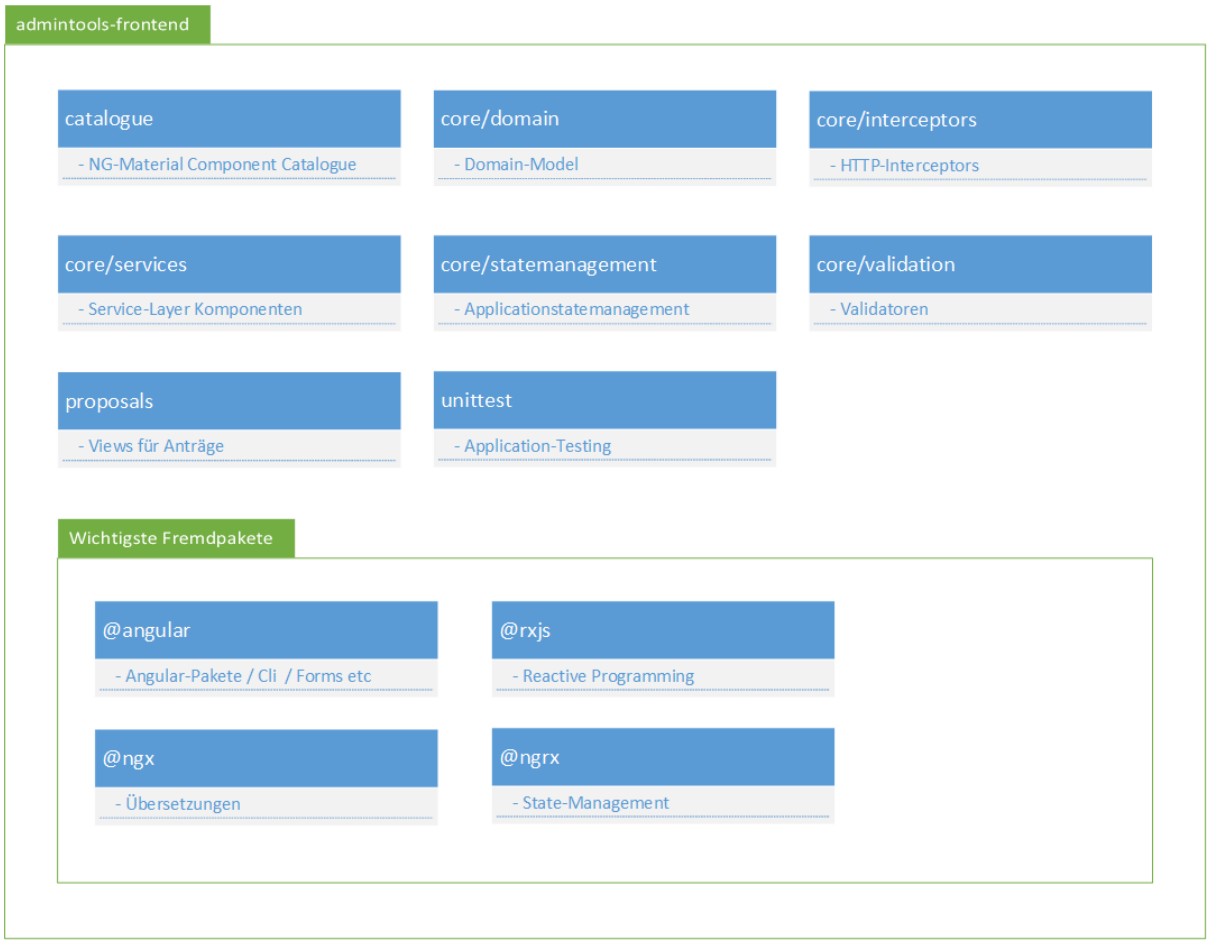
4 Logische Architektur



Layer	Beschreibung
Presentationlayer	Enthält alle Komponenten der Präsentationsschicht. Die Komponenten “Admin-Tools 2.0 FrontEnd” sowie “Admin-Tools 2.0 Admin-Panel” und “LXC-Webpanel” dienen der nutzung und Administration der Applikation. Die Django-REST-Webdokumentation sowie LXC-REST-Webdokumentation sind autogenerierte Seiten welche mit Swagger kreiert wurden.
Business Service Layer	Enthält die Fachliche Logik in Form von Django Modulen welche das Verhalten der Applikation steuern.
Data Service Layer	Die Persistenzschicht der Admin-Tools 2.0 Applikation welche sich um das speichern der Applikationsdaten sowie der erstellen Container kümmert.
Cross Cutting Layer	Alle Komponenten sind mittels der “Shibboleth” Single Sign On Applikation gesichert.

4.1 Architektur Frontend

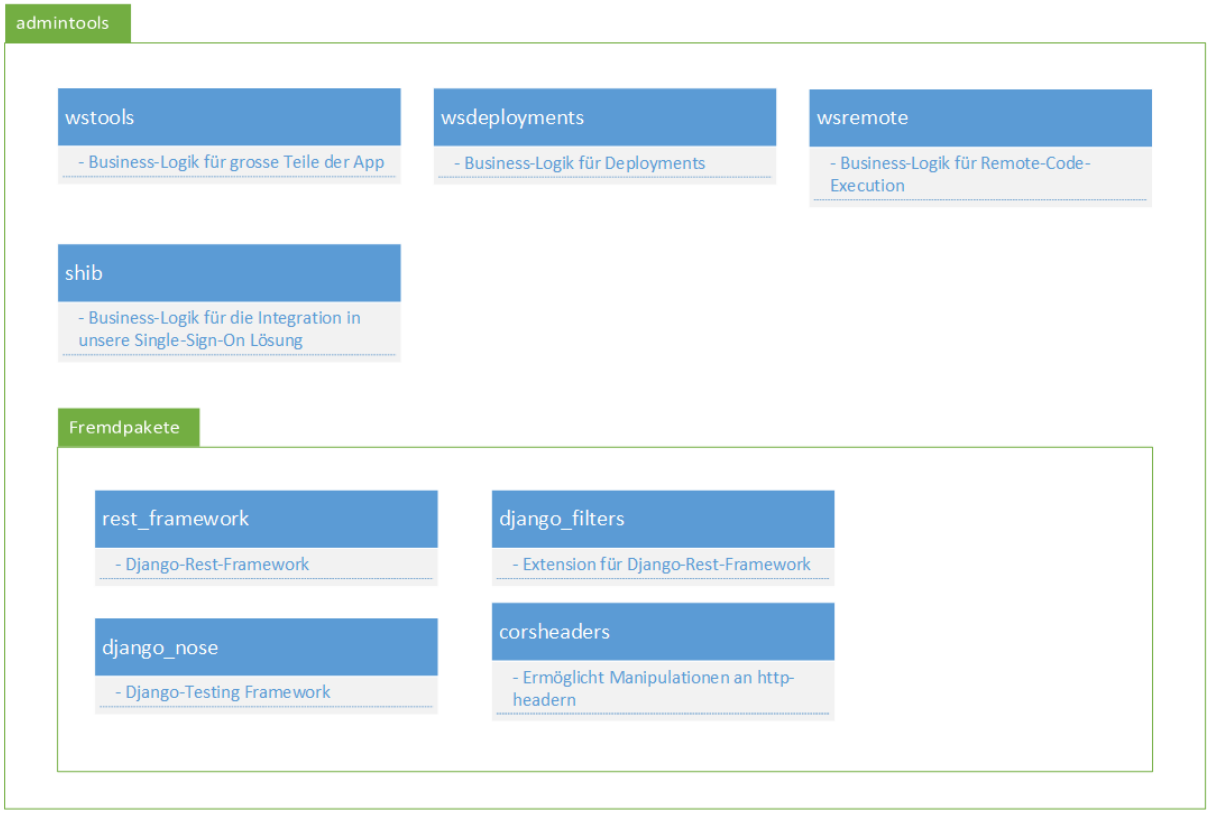
Die zentrale Front-End Komponente der Admin-Tools 2.0 Applikation ist geschrieben in Typescript und Angular 5.0. Die “Admin-Tools 2.0 Admin-Panel” Komponente ist Teil der Business-Service-Layer Schicht und wird durch Django automatisch als Paket mit installiert und als Front-End Komponente exponiert.



Entwickelte Module	Beschreibung
catalogue	Enthält Referenzen auf den von Angular Material erstellten Komponenten-Katalog. Die Material Komponenten enthalten Bausteine der Webapplikation wie Eingabefelder, Buttons, Modale etc.
core/domain	Klassendefinitionen des Applikationsmodells zum Austausch mit dem Backend.
core/interceptors	Klassen zur Filterung des HTTP Datentransfer zwischen Front-End und Backend. Hier werden Fehler und Berechtigungsdaten abgefangen und interpretiert.
core/services	Alle internen Dienste welche zur Steuerung und Regelung der Fachlogik mit dem Backend benötigt werden. Hier wird vor allem auf moderne Dependency Injection Techniken zurückgegriffen.
core/statemanagement	Alle notwendigen Klassen um ein prädiktives Speichermodell zu unterstützen. Die “Redux” Architektur wird angewendet um ein stetig konsistentes Verhalten der Webapplikation zur erzwingen. Zusätzlich ermöglicht diese Architektur eine elegante Weise um unabhängige Tests zu schreiben.

core/validation	Spezialanfertigungen von Validatoren Klassen um die Fachlogik zu unterstützen. Speziell elegant sind die Validatoren welche modular mit Services sowie dem Statemanagment interagieren.
proposals	Enthält alle HTML und Typescript-Komponenten welche dem Benutzer visuell angezeigt werden. Hier sind die Layout und Verhaltensregelungen der Webapplikation abgelegt.
unittest	Enthält Mock-Klassen und Testdaten welche für das Unit-Testing der Applikation verwendet werden.

4.2 Architektur Backend



Entwickelte Module	Beschreibung
wstools	Hauptmodul der Admintools Applikation. Definiert das Backend UI und alle Datenbank Modelle der Applikation. Auch sind hier alle REST-Views und Serializers definiert.
wsdeployments	Stellt die Verbindung zum Containerserver. Definiert alle Scripts, die in diesem Context benutzt werden.

wsremote	Definiert Threadpools die in der Applikation als Python Dekoratoren benutzt werden können. Auf diese Weise können Funktionen und Methoden, die längere Laufzeiten benötigen, in parallele Prozesse ausgelagert werden.
shib	Ersatz für die Django User Komponente. Stellt die Verbindung zum Shibboleth Authentifizierungsdienst zur Verfügung. Erstellt eine Django Session und Django Systemuser anhand der gelieferten Shibboleth Attributen.
Installierte Module	**Beschreibung**
rest_framework	Rest-Framework für Django http://www.django-rest-framework.org/
django_filters	Erweiterung für Rest-Framework https://django-filter.readthedocs.io/en/master/
django_nose	Testing-Extensions für Django https://github.com/django-nose/django-nose
corsheaders	Cross-Origin Resource Sharing https://github.com/ottoyiu/django-cors-headers

Masterarbeit - MAS Software Engineering
HSR Rapperswil 2016-2018

Stakeholder

Team:
Arjan van Doesburg (vda)
Slavisa Karalic (ska)
Robin Ratcliff (rra)

Betreuer:
Daniel Tobler (dt)

Name	Organisation	Rolle
Daniel Tobler	HSR	Betreuer
Luc Bläser	HSR	MAS Studiengangsleiter
Robin Ratcliff	FHNW	Entwickler (Team)
Slavisa Karalic	FHNW	Entwickler (Team)
Arjan van Doesburg	Allianz	Entwickler (Team)
Tom Schneider	FHNW	Teamleiter Webteam FHNW/ Productowner
Roland von Aesch	FHNW	Webmaster FHNW
Jan Rothenberger	FHNW	Projektleiter Webdienste Relaunch

Masterarbeit - MAS Software Engineering
HSR Rapperswil 2016-2018

Glossar

Team:
Arjan van Doesburg (vda)
Slavisa Karalic (ska)
Robin Ratcliff (rra)

Betreuer:
Daniel Tobler (dt)

Name	Beschreibung
Antrag	Gesuch für einen Webdienst, Domäne oder ähnliches
Applikation	Webdienst (Php, Python, WordPress, usw.)
FHNW	Fachhochschule Nordwestschweiz
Webdienst	Eigenständige Webapplikation welche auf Basis eines Frameworks (Wordpress, Mezzanine usw.) von Kunden der FHNW gebraucht werden kann.
RUP	Rational Unified Process (Projekt Methodologie)
SCRUM	Vorgehensmodell in der agilen Software Entwicklung
Product Owner (PO)	Der Auftraggeber welche die fachliche Sicht einer Applikation/ eines Projekts vertritt.
Stand-Up	Tägliches Meeting in der Agile Entwicklungs-Methodologie an welchem die aktuellen Entwicklungsaufgaben besprochen werden im Team.
CI/CD	Continuous Integration, Continuous Deployment. Ein Intergrationskonzept in der Software Entwicklung für das Bauen und Verteilen von Softwarekomponenten.
Task	Kleinste mögliche Aufgabe welche eine Funktionalität / Feature beschreibt welches einem Entwickler zur Umsetzung zugewiesen wird.
Pair Programming	Ein Entwicklungsprozess bei welchem ein Entwickler Code schreibt und ein zweiter Entwickler diesen kontrolliert/überwacht.
Changelog	Eine Datei im Sourcecode im Markdown Format welches die Änderungen die gemacht wurden festhaltet.