msg systems ag

BG ETEM Intranet Präventionswerkzeuge

Systemarchitektur

.consulting .solutions .partnership

BG ETEM
Energie Textil Elektro
Medienerzeugnisse

.msg

BG ETEM Intranet Präventionswerkzeuge

Systemarchitektur

Version	2.01.8
Stand:	05.09.2017
Autor:	Achim.Mueller@msg-systems.com
Ablage:	60_SystemArchitektur.docx
Umfang:	21 Seiten

Versionshistorie

Version	Beschreibung	Autor	Datum
0.9	Ersterstellung	Achim Müller	10.04.2014
1.0	Freigabe zur Abnahme	Stefan Hofmaier	24.04.2014
1.1	Befundung durch BG ETEM	BG ETEM	16.05.2014
1.2	Einarbeitung der Befunde und erneute Freigabe zur Abnahme	Stefan Hofmaier Achim Müller	21.05.2014
1.3	Einarbeitung erneuter Befunde	Stefan Hofmaier	29.05.2014
1.3	Abnahme BG ETEM	BG ETEM	02.06.2014
1.4	Copyright Vermerk angepasst	Stefan Hofmaier	11.07.2014
1.5	Browserunterstützung an Release 1.5 angepasst	Christian Wegert	11.06.2015
1.6	Freigabe zur Abnahme für Rel. 1.5	Stefan Hofmaier	07.07.2015
1.7	Abnahme BG ETEM	Stefan Hofmaier	19.08.2015
1.8	Kapitel 3.1.4 ergänzt: • Aufnahme von Microsoft Edge als unterstützter Browser	Daniel Fiedler	26.10.2016
1.9	Ergänzung Kapitel 3.1.4 • Unterstützung des Internet Explorers	Minh Maria Hoang	31.07.2017

	ab Version 11 Ergänzung Kapitel 3.9 der • Automatischen Installation angepasst • Update der Betriebssoftware eingearbeitet	Minh Maria Hoang & Felix Thiele Felix Thiele	10.08.2017 17.08.2017
2.0	Befunde der BG eingearbeitet	Minh Maria Hoang	23.08.2017

Review

Version	Datum	Teilnehmer
0.9	22.04.2014	Stefan Hofmaier
1.5	03.07.2015	Stefan Hofmaier
1.9	10.08.2017	Stefan Hofmaier

1 Inhaltsverzeichnis

2 Einleitung

2.1 Zweck

Dieses Dokument spezifiziert die System-Architektur und beschreibt die Basis-Architektur für das System. Sie stellt die Sicht auf Technologie und Infrastruktur dar.

Hierzu zählt die Festlegung von eingesetzter Hardware und Systemsoftware, die Verteilung von Systemkomponenten auf verschiedene Knoten, sowie die physische Anbindung von Nachbarsystemen.

2.2 Referenzen

Die Systemarchitektur wurde auf Basis der folgenden von BG ETEM im Rahmen der Ausschreibung zur Verfügung gestellten Dokumente konzipiert:

- Intranet Präventionswerkzeuge - Systemarchitektur

2.3 Abgrenzung

Die Systemarchitektur beschreibt keine fachlichen Anwendungsaspekte. Diese werden in den Dokumenten zum Anwendungsfallmodell, Domänen- und Fachklassenmodell und zum Benutzermodell behandelt.

Die Beschreibung des Software-Designs ist ebenfalls nicht Gegenstand der Systemarchitektur. Das Design wird im Fachgebiet Design erstellt.

3 System-Architektur

Die von der BG ETEM vorgeschlagene System-Architektur wurde in den Ausschreibungsunterlagen im Dokument "Intranet Präventionswerkzeuge-Systemarchitektur.pdf" in der Fassung 1.1 vom 28. März 2013 beschrieben. Sofern notwendig wird in folgendem Text auf das Dokument als "BG ETEM Architekturdokumentation" referenziert.

Die Anwendung wird von msg systems auf der in den folgenden Kapiteln beschriebenen Serverarchitektur entwickelt und für eine Auslieferung auf dieser Serverarchitektur ausgelegt. Um sie mit anderen Architekturkomponenten zu betreiben, insbesondere einem anderen Application Server oder Datenbank, sind unter Umständen kleinere Anpassungen notwendig, die nicht im Projektangebot enthalten sind.

3.1 Unterstützte Web- und Applikations-Server

3.1.1 Java Ablaufumgebung

Die Entwicklung erfolgt auf Basis des Application Servers ~~Glassfish~~Payara.

Eine Architektur mit Java EE Architektur und dem ~~Glassfish~~ Payara Server stellt eine in vielen Projekten bewährte Standardarchitektur dar. msg ~~systems~~ verfügt über sehr gute Erfahrungen in diesem Umfeld. Darüber hinaus erfüllen beide die Bedingung kostenfrei zu sein.

Die Implementierung erfolgt auf Basis der Versionen Java EE ~~6~~7 und Payara 172 ~~Glassfish 3~~ und wird für die Auslieferung auf dieser Serverversion ausgelegt.

3.1.2 Datenbankumgebung

Als Datenbankumgebung wird Postgre~~e~~SQL eingesetzt. msg systems gibt Postgre~~e~~Sql aus folgenden Gründen den Vorzug gegenüber anderen Datenbanklösungen:

PostgreSQL

- orientiert sich am Funktionsumfang der Oracle DB
- ist SQL Standardkonform
- bietet JDBC 4-Treiber
- ist kostenlos
- hat eine große Community

3.1.3 Reverse-Proxy / Web-Server

Als Reverse Proxy / Web-Server wird der Apache Webserver eingesetzt. Dies bringt den Vorteil, dass statische Inhalte wesentlich performanter ausgeliefert werden können und der Application Server entlastet wird. Zusätzlich bietet der Apache Webserver folgende Vorteile:

- Modularer Aufbau
- SSL-Verschlüsselung (mod_ssl)
- Nutzung als Reverse-Proxy (mod_proxy)
- Modifikation von URLs (mod_rewrite, mod_headers)
- Skalierbarkeit
- Unterstützt Ausfallsicherheit

3.1.4 Browser

In der BG ETEM Architekturdokumentation werden als gängige Browser der Microsoft Internet Explorer, Mozilla Firefox, Google Chrome und Apple Safari aufgeführt. Es wird empfohlen sich auf die

wichtigsten Browser in den aktuellen Versionen zu beschränken und je nach Benutzerkreis festzulegen.

Ab Release 1.5 werden der Internet Explorer 10, Firefox 31 und Google Chrome 31 und deren spätere Versionen unterstützt.

Ab Release 1.7 wird zusätzlich Microsoft Edge in der Version 38.14393.0.0 (HTML 14.14393) unterstützt.

Ab Release 2.0 wird der Internet Explorer in der Version 10 nicht mehr unterstützt, sondern erst ab Version 11.0.9600.18738.

3.2 Verwendete Programmiersprachen

Die nachfolgende Tabelle listet die im Projekt verwendeten Programmiersprachen, aufgegliedert nach der Ablaufumgebung, für die diese Programmiersprache verwendet wird, auf.

Infrastrukturkomponente	Sprache	Anmerkung
Browser	HTML 5	Wird verwendet für den Aufbau des Gestaltungs- und Navigationsrahmens und der einzelnen fachlichen Module via Templates (Vorlagen).
	CSS3	Zur visuellen Darstellung (Farben, Layout, Typografie usw.). Implementiert wird über eine Stylesheet-Sprache (LESS bzw. SASS)
	JSON	Wird als Datenaustauschformat (via REST) verwendet, als auch für Konfigurationseinstellungen am Client bzw. Browser, z. B. zur Internationalisierung.
	JavaScript (Ecma-Script ~~5~~6)	Dynamisch typisierte, objektorientierte Skriptsprache zur Umsetzung der Applikationslogik und -steuerung am Client (Browser).
Application Server	Java SE 8~~6~~	Java Standard Edition
	Java EE 7~~6~~	Enterprise Edition Unterstützt von ~~Glassfish~~ Payara 172~~3~~
	JSON	Wird als Datenaustauschformat (via REST) verwendet.
	Embedded JavaScript (optional)	Falls der Einsatz von JavaScript am Server Vorteile bringt, z. B. bei der Wiederverwendung von Validierungscode des Frontends, besteht die Möglichkeit innerhalb der JVM auch JavaScript-Code auszuführen.
	JPA	Der Zugriff auf die Datenbank erfolgt mittels

Infrastrukturkomponente	Sprache	Anmerkung
		des in Java EE integrierten Standard JPA (Java Persistence API).
Datenbank	SQL:1999	Sofern direkt im Code native SQL-Abfragen abgesetzt werden, werden SQL:1999 Sprachmittel verwendet. Bei der mittelbaren Verwendung durch die Java Persistence API hat msg systems keinen Einfluss auf die verwendeten SQL-Sprach Konstrukte.

3.3 Verwendete Frameworks

Bei der Entwicklung der Anwendung Intranet Präventionswerkzeuge kommen folgende Frameworks zur Anwendung:

	Framework	Version		Lizenz
GUI	Node	0.10.26	Runtime Engine	MIT & BSD
GUI	npm	1.4.4	Package Manager	MIT
GUI	grunt	0.4.4	Build Tool	MIT
GUI	JSHint	2.4.4	Linting JavaScript	MIT
GUI	UglifyJS	2.4.13	Compression JavaScript	BSD
GUI	Clean-CSS	2.1.6	Compression CSS	MIT
GUI	HTML-Minifier	0.5.6	Compression HTML	MIT
GUI	PhantomJS	1.9.7	Testing (Runtime Engine)	BSD
GUI	Jasmine	2.0	Testing (Framework)	MIT
GUI	Karma	0.0.12	Testing (Framework)	MIT
GUI	DuckyJS	2.0.3	Value Validation	MIT
GUI	i18next	1.7.2	Internationalization	MIT
GUI	JQuery	2.1.0	DOM Manipulation	MIT
GUI	NumeralJS	1.5.3	Localization (Number)	MIT
GUI	MomentJS	2.5.1	Localization (Date)	MIT
GUI	Lodash	2.4.1	Functional Programming	MIT
GUI	Twitter Boostrap	3.1.1	UI Framework	MIT
GUI	Select2	3.4.5	Controls (Selection)	Apache 2.0
GUI	Typeahead	0.10.2	Controls (Autocompletion)	MIT
GUI	LESS	1.7.0	Style Makro Processor	Apache 2.0
GUI	ES5-Shim	2.3.0	Polyfill ECMAScript 5	MIT
GUI	Font-Awesome	4.0.3	Icon Font Library	SIL Open Font License (OFL)
GUI	TypoPRO	1.4.1	Font Library	OpenSans - Apache 2.0
GUI	Google Charts	1.0	Data Visualization	Creative Commons 3.0
GUI	ComponentJS	1.0.2	JavaScript MVC Framework	MPL 2.0
Server	jackson / faster xml	2.3.1	(Un)marshalling Java <-> JSON für die REST-API zum Client	LGPL 2.1
Server	Glassfish	3.1.2.2	Applikations-Container	CDDL
Server	Hibernate	4.3.4	Persistenz-Framework	LGPL 2.1
Server	Apache [optional]	2.4	Reverse Proxy	Apache 2.0
Server	PostgreSQL	9.3.4	Datenbank	BSD
Server	JDBC-Treiber PostgreSQL	9.3	Verbindung GF zu Datenbank	BSD
Server	Jasper Reports	5.5.1	PDF-Erstellung	LGPL
Server	Apache Lucene	4.7.1	Volltextsuche	Apache 2.0
Server	Apache Maven	3.2.1	Build	Apache 2.0
Server	slf4j	1.7.7	Logging Facade	MIT

msg systems behält sich vor, im Lauf der Entwicklung bei Erfordernis einzelne Komponenten auszutauschen oder zu ergänzen. Hierbei werden die Aspekte Kostenfreiheit und Verteilbarkeit berücksichtigt.

Die verwendeten Lizenzen sichern eine kostenlose Verwendung der Frameworks zu, ebenso wie eine Weitergabe an Dritte, sofern der Quellcode der Frameworks durch diese nicht modifiziert wird.

3.4 Verteilung der Software auf verschiedene Rechner und Knoten

Relevante Zielsetzungen für die Verteilung/Verteilbarkeit der Systemkomponenten sind Lastverteilung und Ausfallsicherheit. Diese Aspekte werden in Kapitel 3.7 Skalierbarkeit des Systems näher betrachtet.

Der Aspekt Verteilbarkeit wird für die folgenden Ausführungseinheiten betrachtet:

1.) Apache Reverse-Proxy
2.) ~~Glassfish 3~~Payara
3.) PostgreSQL

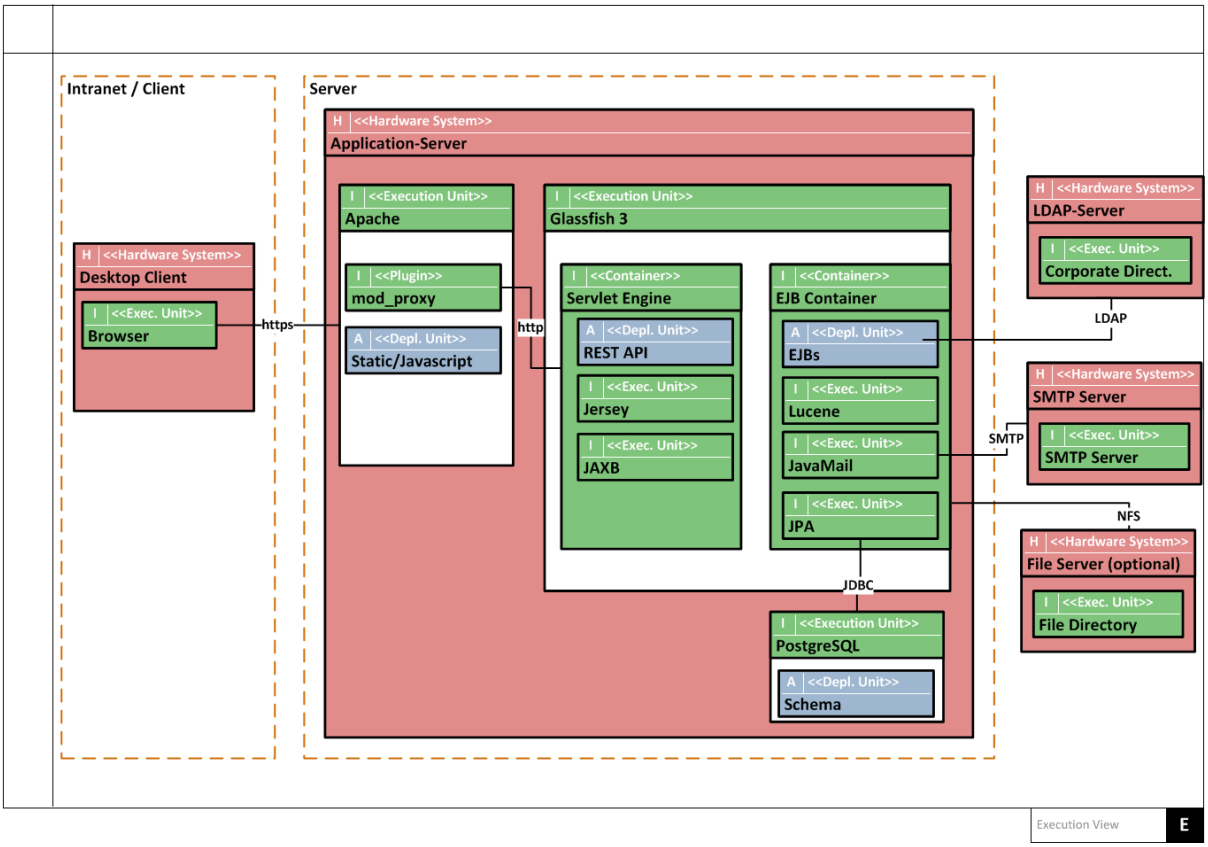
Alle drei Komponenten können ggf. getrennt voneinander auf einzelnen Hardware-Systemen installiert werden.

Damit ergeben sich je nach Anforderung der Mitgliedsunternehmen verschiedene Ausbaustufen der System-Infrastruktur. Wir beschreiben mit der folgenden Tabelle die beiden möglichen Extreme, minimalen und maximalen Ausbau. Tatsächliche Varianten können sich im Spektrum dazwischen befinden.

Verteilung	Beschreibung	Auswirkung
Minimal	Die drei Execution Units werden auf demselben Hardware-System installiert.	• Geringe Hardwarekosten • Einfache Konfiguration und Administrierbarkeit • Keine Lastverteilung und damit nur vertikale Skalierbarkeit • Keine Ausfallsicherheit • Schlechtere Performance
Maximal	Die drei Execution Units werden auf je einem eigenen Hardware-System installiert. Des Weiteren können die Execution-Units auch verdoppelt werden, sodass sich positive Effekte für das Lastverhalten und die Ausfallsicherheit ergeben. Allerdings wird dieser Vorteil mit einem erhöhten administrativen und konfigurativen Aufwand erkauft, speziell im Bereich der Datenbank.	• Höhere Hardwarekosten • Komplexe Konfiguration und Administrierung • Lastverteilung und Skalierbarkeit • Höhere Ausfallsicherheit und Verfügbarkeit

In den allermeisten Fällen sollte aus heutiger Sicht die Minimalvariante ausreichen, da es sich um eine Intranet-Anwendung mit einer begrenzter Anzahl gleichzeitig angemeldeter bzw. agierender Benutzer handelt. Die Maximalvariante findet klassischerweise für Internet-Applikationen mit sehr hohen Benutzerzahlen Anwendung.

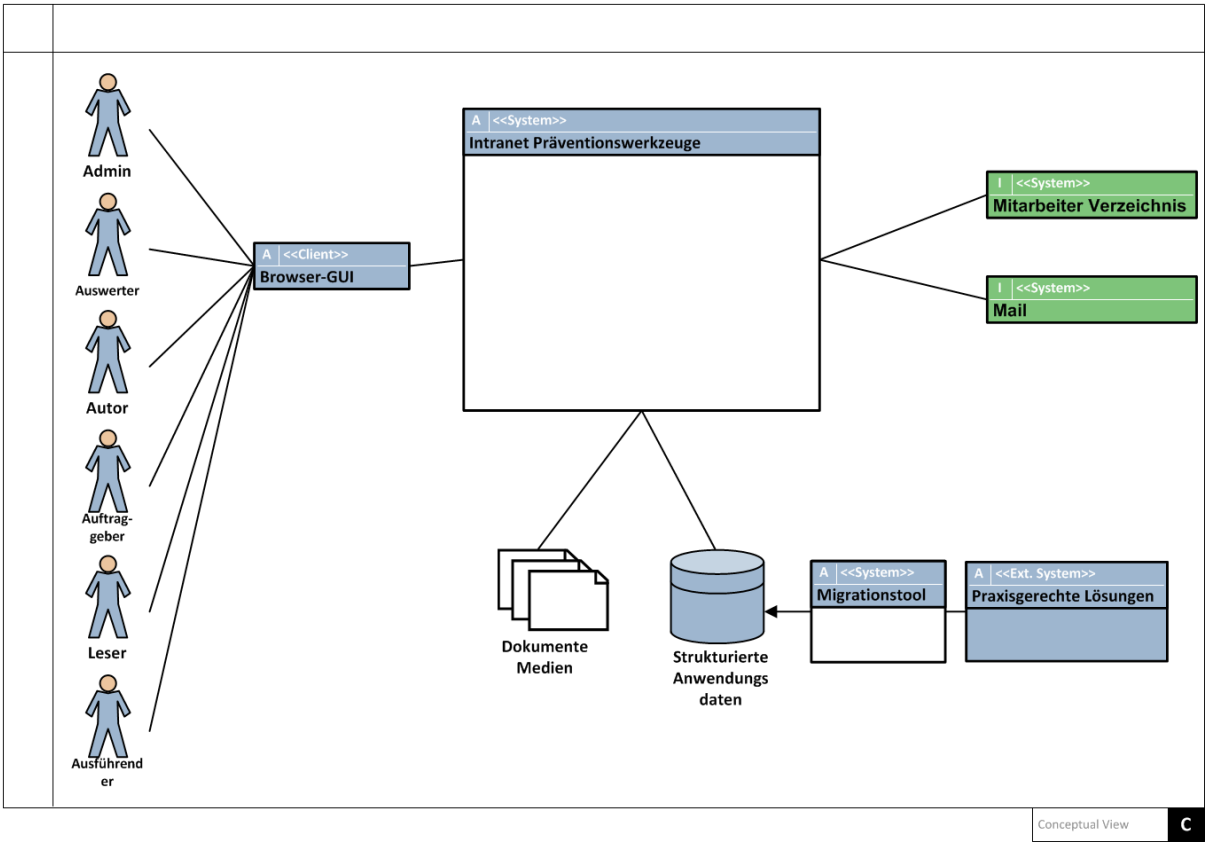
Das für die Anwendung Intranet Präventionswerkzeuge in der ersten Ausbaustufe erwartete Lastprofil spricht für die Minimalvariante. msg ~~systems~~ geht deshalb von dieser Variante aus, so dass sich die Verteilungssicht so darstellt:



Dem ~~Glassfish~~ Payara Application Server vorgeschaltet ist ein Apache Webserver. Dieser liefert den statischen Content wie die JavaScript-Dateien für das Frontend aus und entlastet so den Application Server. Gleichzeitig dient er als Reverse Proxy und ermöglicht eine Verteilung auf mehrere Application Server, falls dies erforderlich wird.

Die tatsächlich zu verwendende Variante ist aber in jedem Fall abhängig vom Lastprofil des Mitgliedsunternehmens.

3.5 Anbindung der Backend-Systeme



Wie aus dem obigen Kontextdiagramm ersichtlich, müssen folgende externe Systeme angebunden werden:

1.) Verzeichnisdienst (LDAP)
2.) Mail/Kalender

Darüber hinaus erfolgt eine Speicherung der Anwendungsdaten in:

3.) Datenbank
4.) Dateisystem

Zum Zugriff auf diese Systeme sind technische Benutzer erforderlich:

1.) LDAP: Technischer Benutzer mit Nur-Lese-Rechten
2.) Mail/Kalender: Technischer Benutzer zum Versenden von Mails auf dem SMTP Server
3.) Datenbank: Technischer Benutzer mit Lese- und Schreibrechten
4.) Dateisystem: Technischer Benutzer, in dessen Kontext der Application Server/Web Server ausgeführt wird, muss Lese- und Schreibrechte besitzen

3.5.1 LDAP

Bei LDAP (Lightweight Directory Access Protocol) handelt es sich um ein Anwendungsprotokoll, um Daten aus einem Verzeichnis-Server (Directory), wie dem Microsoft Active Directory, auszulesen. Ein Verzeichnis-Server wird im Regelfall dazu benutzt, um Benutzerinformationen in einem Unternehmen zu verwalten.

Für die Anwendung Intranet-Präventionswerkzeuge wird davon ausgegangen, dass in dem Unternehmen, in dem die Software eingesetzt wird, ein Verzeichnis-Dienst vorhanden ist, der über eine LDAP-Schnittstelle verfügt und mit dem eine Authentifizierung der Benutzer erfolgen kann. Bei

jedem Login des Benutzers an Intranet Präventionswerkzeuge werden darüber hinaus zusätzliche Daten des Benutzers wie Name, Vorname und E-Mail-Adresse aus dem Verzeichnis-Dienst ausgelesen und in der Datenbank von Intranet Präventionswerkzeuge gespeichert.

Die Erfassung und Pflege der Rollen und Anwendungsberechtigungen erfolgt dagegen ausschließlich in Intranet-Präventionswerkzeuge.

Ein sog. automatischer Single Sign On mit dem Windows Benutzer ist möglich, wenn das Mitgliedsunternehmen über ein Windows Betriebssystem auf Client-Seite verfügt und als LDAP-Server das Microsoft Active Directory eingesetzt wird.

Benutzer, die im LDAP deaktiviert bzw. gesperrt sind, dürfen sich auch nicht in den Intranet Präventionswerkzeugen anmelden können.

3.5.2 Mail/Kalender

An einigen Stellen in der Anwendung ist das Versenden von E-Mails gefordert. Hierbei sind technisch zwei Fälle möglich:

- Versand der Mails vom Browser aus über den installierten E-Mail-Client des Benutzers
- Versand vom Server über eine SMTP-Schnittstelle mittels eines technischen Benutzers

In ersterem Fall wird auf der HTML-Seite ein Link hinterlegt, der mit "mailto:" beginnt, gefolgt von der E-Mail Adresse des Empfängers und parametrisiert mit dem "Betreff" und "Text" der Mail. Beim Klicken auf den Link öffnet sich der standardmäßig konfigurierte E-Mail-Client des Benutzers mit der vorbereiteten E-Mail, die der Benutzer noch modifizieren kann, und explizit durch "Senden" verschicken kann. Dies soll laut Ausschreibung der Standardweg für den Versand von fachlichen E-Mails darstellen.

Im zweiten Fall erfolgt der Versand der E-Mails über den Application Server mittels eines SMTP Servers. Dieser Weg war für technische E-Mails gedacht, z. B. mit Fehlermeldungen des Systems, die an Administratoren gesendet werden. Hierfür ist die Bereitstellung eines SMTP Servers mit einem technischen Benutzer zum Versand der E-Mails erforderlich.

Der Eintrag von Terminen lässt sich mittels Versand einer E-Mail mit einer .ics Datei als Anhang bewerkstelligen. Das .ics Dateiformat wird von den gängigen E-Mail-Programmen insbesondere auch von Lotus Notes unterstützt und bietet die Möglichkeit Serientermine einzustellen.

E-Mails werden aus folgenden Gründen ausschließlich über den Server verschickt:

- Kalendereinträge (.ics-Dateien) können nicht vom Browser verschickt werden
- Der Serverversand ist zuverlässiger und kann besser überwacht werden
- Einheitliches Verfahren aus Gründen der Wartbarkeit

3.5.3 Datenbank

Die Datenbank ist für das Funktionieren der Anwendung essentiell. Dort werden insbesondere Benutzer mit ihren Rollen, Aufgaben und fachlichen Informationen zu den Dokumenten gespeichert. Die Ablage der fertig erstellten Dokumente im PDF-Format erfolgt nicht in der Datenbank, sondern im Dateisystem (siehe nachfolgendes Kapitel).

Für den Zugriff der Anwendung auf die Datenbank ist dort ein entsprechender technischer Benutzer mit Lese- und Schreibrechten einzurichten.

Der Datenzugriff erfolgt von der Anwendung mit JPA.

3.5.4 Dateisystem für Dokumentenablage

Die Ablage von generierten Dokumenten oder importierten Dokumenten (primär im PDF-Format) erfolgt wie in der BG ETEM Architekturdokumentation vorgeschlagen im Dateisystem. Dort wird allerdings noch zusätzlich das Aufsetzen eines separaten Apache Web Servers als dedizierter Fileserver empfohlen. msg systems hält dies aufgrund des zu erwartenden Lastprofils nicht für

erforderlich und sieht eine Auslieferung der Dokumente über den Reverse Proxy Apache Server für ausreichend.

Auf das Filesystem für die Dokumentenablage haben sowohl der Reverse Proxy Apache Web Server als auch der ~~Glassfish~~ Payara Application Server Zugriff. Das Filesystem kann entweder direkt auf dem Server liegen, auf dem die Anwendung läuft, oder auch per NFS gemountet werden. Aus Sicht von Intranet Präventionswerkzeuge ist es lediglich ein Verzeichnis, welches in der Anwendung bei der Erstauslieferung konfiguriert wird.

3.6 Verwendete Kommunikationsprotokolle

Protokoll	Grundlage/ Version	Von	Nach	Bemerkung
HTTPS	HTTP SSL/TLS	Browser	Application Server	HTTPS ist für eine sichere Übertragung dringend erforderlich, insbesondere beim Login, da das Passwort ansonsten im Klartext übertragen wird.
REST/ JSON	HTTP(S)	Browser	Application Server	Anwendungsprotokoll zwischen der Browser-GUI und dem Server-Backend
LDAP	RFC 4510/4511	Application Server	Directory Server	Authentifizierung und Auslesen Benutzerdaten
SMTP	RFC 821	Application Server	Mail Server	Serverseitige Mails versenden
JDBC	4.0	Application Server	Datenbank	Zugriff über den JDBC-Treiber für PostgreSQL

3.7 Skalierbarkeit des Systems

Unter Skalierbarkeit versteht man die Fähigkeit eines Systems sich an gestiegene Last- oder Performanceanforderungen anzupassen, z. B. um bei einer höheren Zahl gleichzeitiger Benutzer oder größeren Datenmengen Anfragen mit gleichbleibenden Antwortzeiten verarbeiten zu können.

Grundsätzlich unterscheidet man zwischen vertikaler Skalierbarkeit (scale up) und horizontaler Skalierbarkeit (scale out). Bei vertikaler Skalierung wird die Leistung einer einzelnen Instanz bzw. eines Knotens der Software verbessert, indem dem Knoten mehr Hardware-Ressourcen wie z. B. mehr CPU-Leistung oder Hauptspeicher zur Verfügung gestellt wird. Bei horizontaler Skalierung verteilt man die Software auf mehrere Instanzen bzw. Knoten (vgl. auch 3.4 Verteilung der Software auf verschiedene Rechner und Knoten).

Während eine vertikale Skalierung immer möglich ist, sofern noch leistungsfähigere Hardware-Komponenten verfügbar sind, muss die Software bei der horizontalen Skalierung bestimmte Voraussetzungen erfüllen. msg systems entwirft die Anwendung Intranet Präventionswerkzeuge so,

dass eine horizontale Skalierung möglich ist, sofern das Lastprofil dies erforderlich macht. Hierzu gehören:

- Verwendung von zustandslosen Benutzersitzungen
  Mit der Java Enterprise Edition ist es möglich zustandsbehaftete oder zustandslose Benutzersitzungen zu entwickeln. Da sich eine Anwendung mit zustandslosen Benutzersitzungen wesentlich besser skalieren lässt, wird diese Variante gewählt.
- Keine langlaufenden Transaktionen

  Langlaufende Transaktion blockieren Ressourcen und können ein System deutlich langsamer machen, sofern mehrere Benutzer gleichzeitig solche Transaktionen auslösen. Langlaufende Transaktionen werden deshalb soweit möglich vermieden und stattdessen z. B. asynchrone Kommunikationsformen eingesetzt.
- Optimistisches Sperrverfahren auf Datenbank-Ebene

  Der schreibende Zugriff auf Daten sperrt diese so kurz wie möglich, lesender Zugriff ist weiterhin möglich.
- Feingranulare Schnittstelle zwischen Client und Server

  Die REST-Schnittstelle zwischen Client und Server wird möglichst feingranular gestaltet. D.h. der Client kann nur die Daten anfordern, die er wirklich gerade benötigt, unnötiger Overhead auf Serverseite, was den Ressourcenverbrauch angeht, und auch die übermittelte Datenmenge wird dadurch reduziert.

3.8 Software-Architektur

Die Software-Architektur folgt der Referenz-Architektur einer klassischen 3-Schichten Architektur. Aufgrund der Ausprägung als Rich Intranet Application (RIA) befindet sich die Präsentationsschicht größtenteils im Browser in Form von HTML5 und JavaScript-Komponenten. Das Serversystem wird über REST-Services angebunden. Aus diesem Grund wird die klassische 3-Schichten Architektur dadurch erweitert sodass sich letztlich 5 Schichten ergeben: Client, Interaktionsschicht, Business Schicht, Integrationsschicht und Ressourcen Schicht.

Das nachfolgende Diagramm stellt mehrere strukturelle Aspekte des Systems dar. Diese Aspekte sind

- das Schichten-Modell
- die Aufteilung in Client und Server
- eine Abbildung fachlicher Komponenten auf technische Komponenten

Im Folgenden werden diese Aspekte und die einzelnen Schichten näher erläutert.

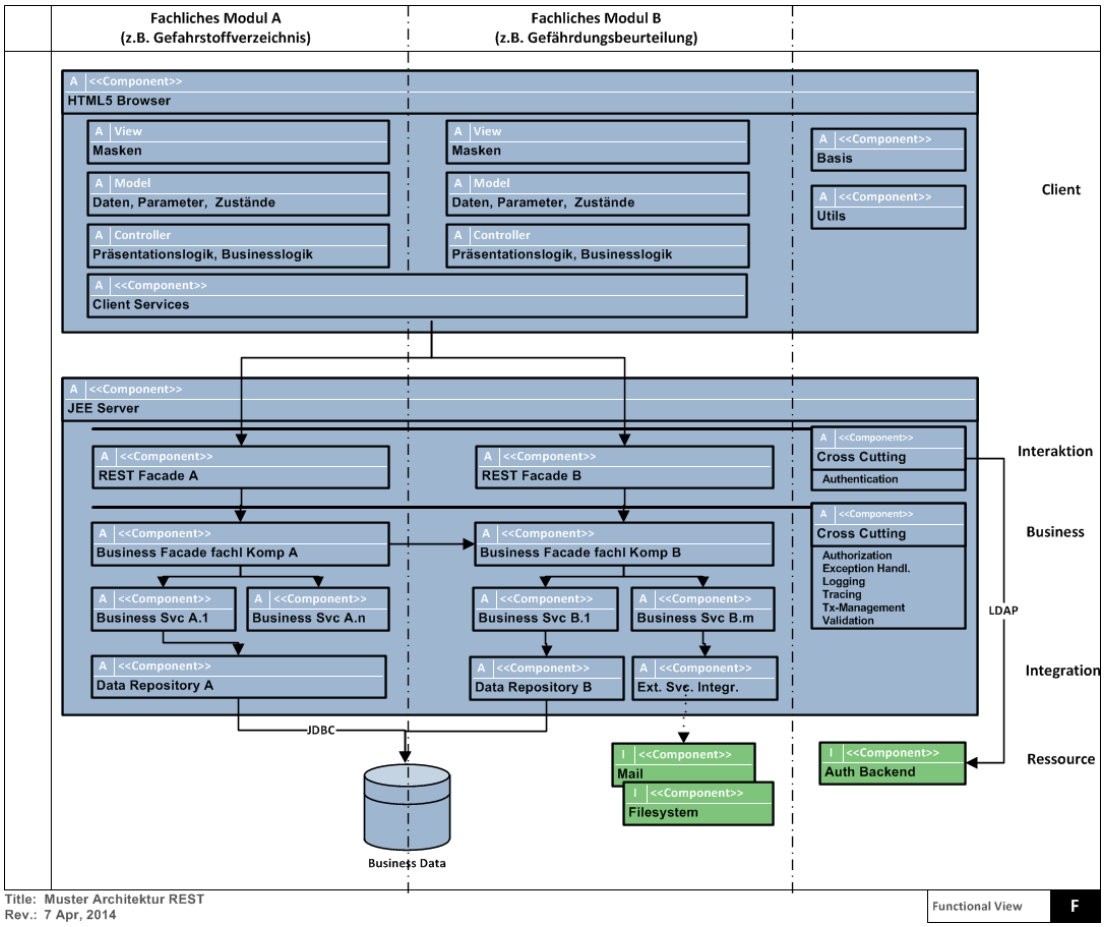


Abbildung: Schichten und Komponenten

Der Client wird, wie bei Rich Intranet Applikationen üblich, als sog. Einzelseiten Webanwendung realisiert.

Um eine hohe Interaktivität zu ermöglichen, werden die Präsentationsschicht bzw. die einzelnen fachlichen Module vollständig browserseitig mit HTML5 und JavaScript realisiert.

Über ein MVC-Modell (Model-View-Controller) wird ein modularer Aufbau der entsprechenden Anwendungen sichergestellt.

Im Server wird eine Java EE Enterprise Anwendung bereitgestellt. Die Kommunikation zwischen Präsentationsschicht und dem Backend erfolgt über Web-Services auf Basis des REST-Protokolls.

Diese Services enthalten die Geschäftslogik und binden externe Ressourcen mit JSON Objektstrukturen als Datenaustauschformat ein. Services entsprechen damit dem Modell im MVC-Modell (Model-View-Controller).

Der Komponenten-Schnitt, am Beispiel der Komponenten „Gefahrstoffverzeichnis“ und „Gefährdungsbeurteilung“ dargestellt, dient als Muster für die Umsetzung der konkreten fachlichen Komponenten auf ein technisches Modell. Eine Komponente besteht immer aus dem kompletten vertikalen Durchstich, d.h. von den HTML5/JavaScript Masken bis zu den Daten-Repositories, die auf bestimmte Tabellen in der Datenbank zugreifen. Untereinander kommunizieren die Komponenten über interne Schnittstellen.

Die folgenden Unterkapitel beschreiben die einzelnen Schichten.

3.8.1 Client- Schicht

In dieser Schicht wird der Client als Einzelseiten Webanwendung mit HTML5, CSS und JavaScript implementiert. Die Vorteile dieser Technologie sind folgende:

- Mit der Verwendung von Webstandards ergibt sich eine hohe Kompatibilität und nachhaltiger Investitionsschutz
- Es werden keine zusätzlichen Installationen bzw. Plugins auf Seite des Clients benötigt.
- Die Anwendung wird direkt im Web-Browser ausgeführt und gewährleistet damit eine hohe Portabilität
- Der Browser ist verantwortlich für die Erstellung, Präsentation und Kontrolle der Bedienoberfläche. Dialogsteuerung und Validierung von Benutzereingaben werden von ihm übernommen.
- Erst wenn es diese Prüfungen durchgeführt wurden, verantwortet das Backend die weiteren abschließenden Validierungen
- Die Kommunikation zwischen Client und Server wird erheblich reduziert. Das Datenmodell wird auf dem Client gehalten. Ein Datenaustausch mittels schlanker Objektstrukturen (REST/JSON) findet nur bei wenigen Benutzerinteraktionen statt.

Für eine modularisierte Architekturgestaltung wird auf das MVC-Entwurfsmuster (Model-View-Controller) zurückgegriffen. Hierbei wird nicht der komplette Client in Model, View und Controller aufgeteilt, sondern für jede einzelne Komponente wird diese Gliederung umgesetzt.

Der Client besteht nicht nur aus den spezifischen fachlichen Komponenten für die jeweiligen Module, sondern es existieren auch allgemeine Komponenten. Beispielsweise der Anwendungs- und Navigationsrahmen sowie die Administrationsoberfläche.

3.8.2 Interaktionsschicht

Die Interaktionsschicht bietet dem Client eine Sicht auf die Business-Funktionalität im Server. Sie stellt diese Sicht in Form von HTTP REST-Services zur Verfügung. REST ist ein Architektur-Ansatz, den wir auf Grund seiner signifikanten Vorteile gegenüber anderen Lösungsmöglichkeiten gewählt haben. Der große Vorteil des REST-Ansatzes liegt in folgenden Aspekten:

- Die Services sind für alle Clients über das HTTP-Protokoll anbindbar. Dies ermöglicht eine einfache Anbindung des HTML5/JavaScript Clients und in einer späteren Ausbaustufe ggf. einer mobilen Lösung, z. B. auf einem Tablet-PC. Diese Funktionalität wird nicht über Java EE-native Protokolle oder für JavaScript nur umständlich handhabbares XML nach außen zur Verfügung gestellt.
- Die Anbindung über HTTP-basierte Services erlaubt es, die Anwendung ohne Freischaltung eines proprietären Ports in der Firewall zu betreiben.
- Zur Authentifizierung kann auf vorhandene HTTP-basierte und von Applikation-Servern unterstützte Ansätze zurückgegriffen werden (Baustein).
- Der REST-Ansatz ist ein zustandsloser Architektur-Ansatz, d.h. auf den Applikations-Servern wird kein Zustand in Form von Präsentations-Sessions gehalten – im Gegensatz zu beispielsweise JSF-Anwendungen. Der Zustand wird zum überwiegenden Teil im Client verwaltet. Dieser Ansatz bietet Vorteile in Bezug auf die Verfügbarkeit und Skalierbarkeit der Lösung.

3.8.3 Business Schicht

Die Business Schicht stellt Geschäftslogik in Form von zustandslosen Geschäftsfunktionen optional Geschäftsdiensten zur Verfügung. Hier kommen zustandslose Objekte der Java Enterprise Edition zum Einsatz, die nicht nach außen über das EJB-Protokoll sichtbar sein werden, sondern nur von den REST-Services genutzt werden.

Geschäftsfunktionen sind der zentrale Einstieg in die Geschäftslogik. Sie stellen grobgranulare, anwendungsfall-orientierte Funktionen bereit und steuern und delegieren an wiederverwendbare Business-Komponenten, die feingranulare, atomare Fach-Operationen darstellen.

Auf der Ebene der Geschäftsfunktionen sind auch wichtige Querschnittsaspekte verortet – einige Beispiele sind:

- *Transaktions-Management (Tx-Mgmt):* Transaktionen werden hier gestartet und beendet. Aufgerufene Komponenten wie Geschäftsfunktionen und Datenobjekte nehmen nur an diesen

Transaktionen teil, steuern diese aber nicht. Die Lösung wird konsequent ohne verteilte bzw. langlaufende Transaktionen implementiert, was einen Vorteil für die Skalierbarkeit und die Verfügbarkeit der Anwendung bedeutet.
- *Datenvalidierung*: Vor der Verarbeitung werden fachliche und sicherheitsrelevante Validierungen an den eingehenden Daten durchgeführt.
- *Fehlerbehandlung und Protokollierung:* Als zentraler Einstiegspunkt in die Geschäftsfunktionalität sind die Einstiegsfunktionen prädestiniert um dort ein strukturiertes, einheitliches Management dieser beiden Aspekte anzusiedeln.
- *Laufzeitmessung*: ist ein Unteraspekt der Protokollierung. Messwerte werden in einer eigenen Protokolldatei zur Verfügung gestellt, welche von Monitoring-Tools wie z. B. Nagios, ausgewertet werden kann.

3.8.4 Integrationsschicht

Die hier angesiedelten Daten-Objekte, techn. ebenfalls als zustandslose EJB-Objekte ausgelegt, kapseln den Zugriff auf je eine Untermenge der Geschäftsobjekte in der Datenbank. Die Daten-Repositories enthalten keine Geschäftsfunktionalität, sondern stellen nur CRUD-Operationen (CRUD = Create/Read/Update/Delete) bereit.

Die Integration-Services sind ebenfalls zustandslose EJBs und kapseln die Kommunikation mit den angeschlossenen Fremdsystemen, insbesondere dem Mail-System und LDAP. Darüber hinaus wird es auch Dienste für den Zugriff auf das Dateisystem geben.

3.8.5 Ressourcen Schicht

Die Ressourcen Schicht besteht bei Intranet Präventionswerkzeuge primär aus der zugehörigen Datenbank und dem Filesystem zur Dokumentenablage.

Darüber hinaus gehören hierzu auch die angeschlossenen Systeme:

- Verzeichnisdienst (LDAP)
- SMTP-Server für E-Mail-Versand

3.9 Auslieferung der Software

Bei der Auslieferung der Software ist zwischen der Erstinstallation und der Installation von Updates zu unterscheiden.

Bei der Erstinstallation geht msg ~~systems~~ davon aus, dass die Mitgliedsunternehmen einen Server mit dem Betriebssystem Linux (idealerweise ~~Suse Enterprise Edition~~ Ubuntu ~~11~~16.04.3 LTS Xenial Xerus) oder Windows Server 2008 R2 oder höher zur Verfügung stellen. msg ~~systems~~ liefert ein vollständiges vorkonfiguriertes Softwarepaket bestehend aus einem Apache Webserver, dem Application Server ~~Glassfish~~ Payara und der Datenbank mit dem bereits angelegten Datenmodell. Eine Anpassung an die Zielumgebung erfolgt automatisiert mittels ~~einer oder mehrerer separater Konfigurationsdateien, in denen unter anderem folgende Daten enthalten sind:~~eines Installationsskripts, welches u. A. die folgenden Daten abfragt:

- LDAP-Server Adresse mit technischem Benutzer zum lesenden Zugriff (sofern notwendig), Einstiegsadresse im Verzeichnis (BaseDN) und Suchfilter zum Auffinden von Benutzern im LDAP-Verzeichnis
- SMTP Server Adresse mit techn. Benutzer zum Versenden von E-Mails
- Verzeichnis, in dem die Dokumente abgelegt werden

Die Bekanntgabe der o.g. Angaben muss vom Mitgliedsunternehmen erfolgen, ebenso wie die Einrichtung von technischen Benutzern zum Zugriff auf die angeschlossenen Systeme und das Anlegen des Verzeichnisses zur Ablage der Dokumente mit Lese- und Schreib-Zugriffsrechten.

~~Die Auslieferung der Anwendung für die Erstinstallation und insbesondere für Updates erfolgt in Form einer EAR-Datei (Enterprise Application Archive), dies ist das Standardformat für die Verteilung einer~~

~~mit der Java Enterprise Edition erstellten Applikation. Diese kann auf dem Glassfish Application Server mittels der integrierten Administrator Konsole eingespielt werden.~~

Die Auslieferung erfolgt in zwei ZIP Archiven für das Betriebssystem Linux und das Betriebssystem Windows.

Die beiden Pakete können ebenfalls zum Upgrade einer bestehenden Umgebung der Intranet Präventionswerkzeuge verwendet werden. Hierbei werden die schon getätigten Angaben nicht erneut abgefragt, sondern es sind lediglich die Werte neuer Konfigurationsparameter zu bestimmen. Um eine unautorisierte Installation zu verhindern muss bei der erstmaligen Installation ein fest konfiguriertes Passwort angegeben werden. Weitere Upgrades benötigen dann kein Passwort. Das Upgrade-Paket bietet die Möglichkeit ~~Versionen~~Programmausgaben ab der Version 2.0 direkt auf den aktuellen Stand der Intranet Präventionswerkzeuge zu aktualisieren. Es ist damit für Mitgliedsunternehmen nicht nötig, Upgrades einzeln zu durchlaufen.

Die JSON-Dateien (Client-Ressourcen), die bei der manuellen Erfassung einer neuen Sprache über die Sprachverwaltung hinzugefügt worden sind, werden bei einem Upgrade automatisiert angepasst. Die Anwendung aktualisiert die JSON-Dateien und fügt für neu hinzugefügte Einträge entsprechende Platzhalter ein.

~~Falls ein Update eine Änderung der Datenstruktur beinhaltet, wird ein Datenbankskript mitgeliefert, welches die Datenbankstruktur an die auszuliefernde Version anpasst. Dieses muss mittels einer Datenbank Administrator Konsole eingespielt werden.~~

Bestandteil der Auslieferung ist auch ein Betriebshandbuch, ~~in dem die Erstinstallation und Updates beschrieben werden. Die Installation und Updates werden vom Mitgliedsunternehmen anhand dieses Betriebshandbuchs ausgeführt.~~in dem der Betrieb der Software beschrieben wird.

3.10 Entwicklungsrichtlinien

Zur Sicherstellung der Wartbarkeit und Erweiterbarkeit des Programmcodes wird ein Entwicklerhandbuch erstellt, die Entwicklungsrichtlinien beinhalten, nach denen sich jeder Entwickler zur richten hat. Die Einhaltung der Richtlinien wird durch Werkzeuge zur statischen Codeanalyse sowie über regelmäßige Code-Reviews des Chef-Entwicklers oder Software-Architekten sichergestellt.

Exemplarische Inhalte des Entwicklerhandbuchs sind:

Coding Conventions

- Namen
- Formatierung
- Kommentare
- Komplexität (Länge Klassen/Methoden/Parameter)
- Struktur einer Klasse (private/public, Reihenfolge der Methoden)
- Datenobjekte mit toString, equals, hashCode

Projektstruktur in der Entwicklungsumgebung

- Eclipse Projekte
- Schichten

Wiederverwendbare Komponenten

- E-Mail schicken
- Konfiguration der Anwendung
- PDF Generator
- Benutzerverwaltung

Protokollierung

- Was ist wie zu protokollieren und mit welchem Log-Level?
- Welches Logging-Framework wird verwendet?

Fehlerbehandlung

- Welche Fehlerkategorien gibt es?
- Wie ist mit Fehlern umzugehen?
- Formatbeschreibung wie das Backend Fehler an das Frontend meldet

Testing

- Wo werden Tests abgelegt?
- Welche Tests sind zu schreiben, welche Ebene?
- Welches Test-Framework?

Zusammenarbeit

- Wie oft ist Quellcode im Source Repository zu speichern?
- Commit-Kommentare
- Bauen eines Releases
- Branching und Merging

3.11 Softwareentwicklungsprozess

Der Softwareentwicklungsprozess erfolgt nach dem modernen Prinzip der Kontinuierlichen Integration (Continuous Integration), um die maximal mögliche Softwarequalität zu erreichen.

Er ist gekennzeichnet durch:

- Gemeinsame Codebasis

  Der Quellcode aller Entwickler wird in einem zentralen Repository mit Versionsverwaltung abgelegt. In der msg systems kommt hier der Marktführer „Subversion“ zum Einsatz.

- Automatisierter Build und Test

  Der Quellcode wird in regelmäßigen Abständen automatisiert übersetzt (nächtlich oder nach jeder Änderung des Quellcodes), um mögliche Fehler schnellstmöglich zu entdecken. Dabei werden auch von den Entwicklern zu erstellende automatisierte Testfälle durchlaufen. Als Build- und Konfigurationswerkzeug kommt „MAVEN“ und als Umgebung für den Standbau „Jenkins“ zum Einsatz.

- Häufige Integration

  Jeder Entwickler integriert seine Änderungen so oft wie möglich in die zentrale Codebasis. Mit kurzen Integrations-Intervallen reduziert man das Risiko von Fehlern im Integrationsprozess und sichert gleichzeitig die Stabilität der Anwendung.

- Automatische Verteilung

  Jede Softwareversion kann auf Knopfdruck automatisch in die Testumgebungen verteilt werden, um kurze Testzyklen zu ermöglichen.

- Produktionsnahe Testumgebung

  Die Änderungen können in einem Abbild der realen Produktionsumgebung getestet werden. Es existieren mehrere Testumgebungen je nach Zielgruppe. So steht den Entwicklern eine separate Testumgebung zur Verfügung, ebenso wie dem Kunden zur fachlichen Einsichtnahme.

4 Offene Punkte

Nr.	Beschreibung	Verantwortlich