Architektur-Dokumentation zu “ShareIt”

mit:

Template

Martin Dörner
Paul Masch
Daniel Goller

23. Juni 2017

Über arc42

arc42, das Template zur Dokumentation von Software- und Systemarchitekturen.

Erstellt von Dr. Gernot Starke, Dr. Peter Hruschka und Mitwirkenden.

Template Revision: 7.0 DE (asciidoc-based), January 2017

Inhaltsverzeichnis

. 1
Einführung und Ziele 3
Aufgabenstellung 3
Qualitätsziele 3
Randbedingungen 4
Kontextabgrenzung 4
Fachlicher Kontext 4
Technischer Kontext 5
Lösungsstrategie 5
Bausteinsicht 6
Whitebox Gesamtsystem 6
Ebene 2 8
Verteilungssicht 10
Querschnittliche Konzepte 11
Persistenz 11
Logging 11
Authentifizierung 11
Fehlermeldungen 11
Risiken und technische Schulden 11
Datenschutz 11
Glossar 12

Einführung und Ziele

Aufgabenstellung

Ziel der Verleihplattform ShareIt ist es, den wechselseitigen Verleih von Lehr- und Lernmaterialien innerhalb einer eingegrenzten Gruppe von Studierenden zu organisieren und zu verwalten. Dieses Gesamtziel lässt sich gliedern in die folgenden drei Hauptaufgabenbereiche:

- Verwaltung von Benutzerdaten
- Verwaltung von Exemplaren
- Ausleihe und Rückgabe der Exemplare

Qualitätsziele

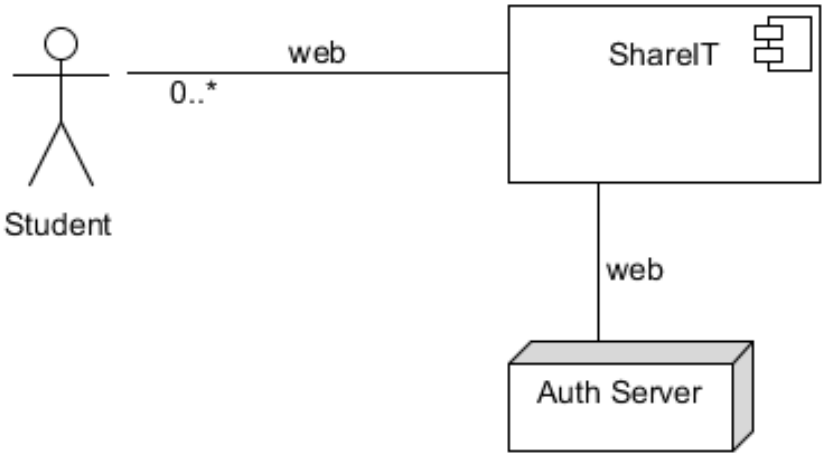
- Ressourcenschonend:
  Das System soll die Basisinformationen zu diesen gleichwertigen Exemplaren so verwalten, dass der Datenbestand möglichst frei von Redundanzen ist und dass inhaltlich gleiche Exemplare auch als gleichwertig behandelt werden.
- Modular:
  Bei der Gestaltung des Systems ist einzuplanen, dass mittelfristig nicht nur Bücher über ShareIt ausgetauscht werden können, sondern je nach Bedarf auch andere Arten von Medien (Beispielsweise DVDs, BlueRay, Audiobooks ...).
- Technische Zugriffbarkeit:
  Die über ShareIt verbundenen Studierenden studieren nicht notwendigerweise alle am selben Ort. Darüber hinaus pendeln viele Studierende zwischen Heimat- und Studienort oder wohnen während eines Praxis- oder Auslandssemesters in einer anderen Stadt. Auch derartig mobile Studierenden sollen ShareIt nutzen
- Authentifizierung:
  Studierende, die ShareIt nutzen wollen, müssen sich vorab als Benutzer registrieren. Geplant sind zwei Arten von registrierten Benutzern, nämlich Administratoren und “normale” Benutzer.

Randbedingungen

- Arbeitsname:
  Das zu erstellende System wird auf den Arbeitsnamen ShareIt getauft.
- Implementierung:
  Das System soll als Webapplikation realisiert werden.

Kontextabgrenzung

Fachlicher Kontext



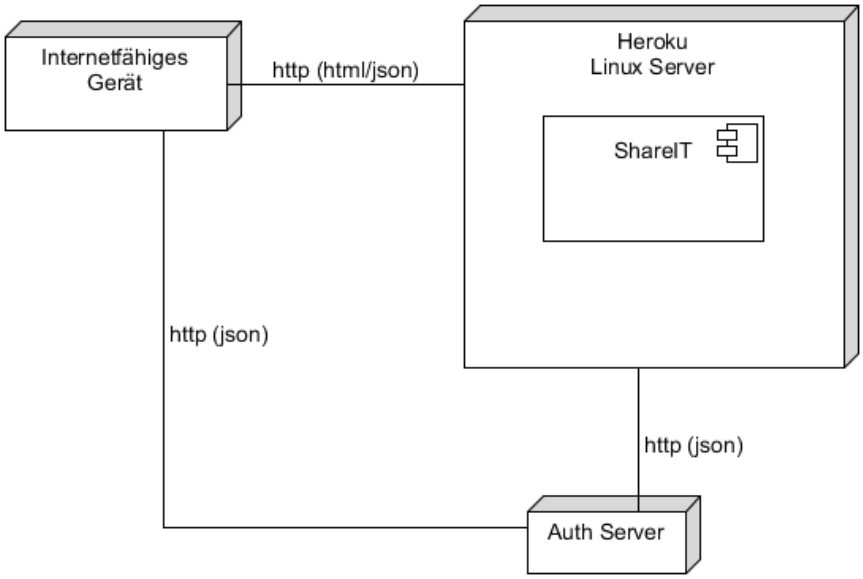
Student (Benutzer)

Um mit dem System interagieren zu baut der Student eine Internetverbindung zu diesem auf mit seinem selbstgewählten Webbrowser und kann nach einer Authentifizierung die Funktionalitäten verwenden.

Auth Server (Fremdsystem)

Das System selber verbindet sich mit einem zuständigen Authentifizierung Server über das Internet oder Intarnet um sich anmeldende Benutzer zu verifizieren.

Technischer Kontext



Alle Übertragungen von Informationen erfolgt mit dem ungesicherten Transport Protokoll HTTP

Beim abrufen der Internetpräsenz des Systems wird einmalig das User Interface mit HTML auf das internetfähige Gerät geladen. Jede weitere Interaktion wird ausschließlich durch das User Interface als JSON Objekt übertragen und empfangen. Dies betrifft auch die Kommunikation mit dem Fremdsystem.

Für eine detaillierte Auskunft der REST Schnittstellen beider Systeme die Readmes der Repositoris lesen:

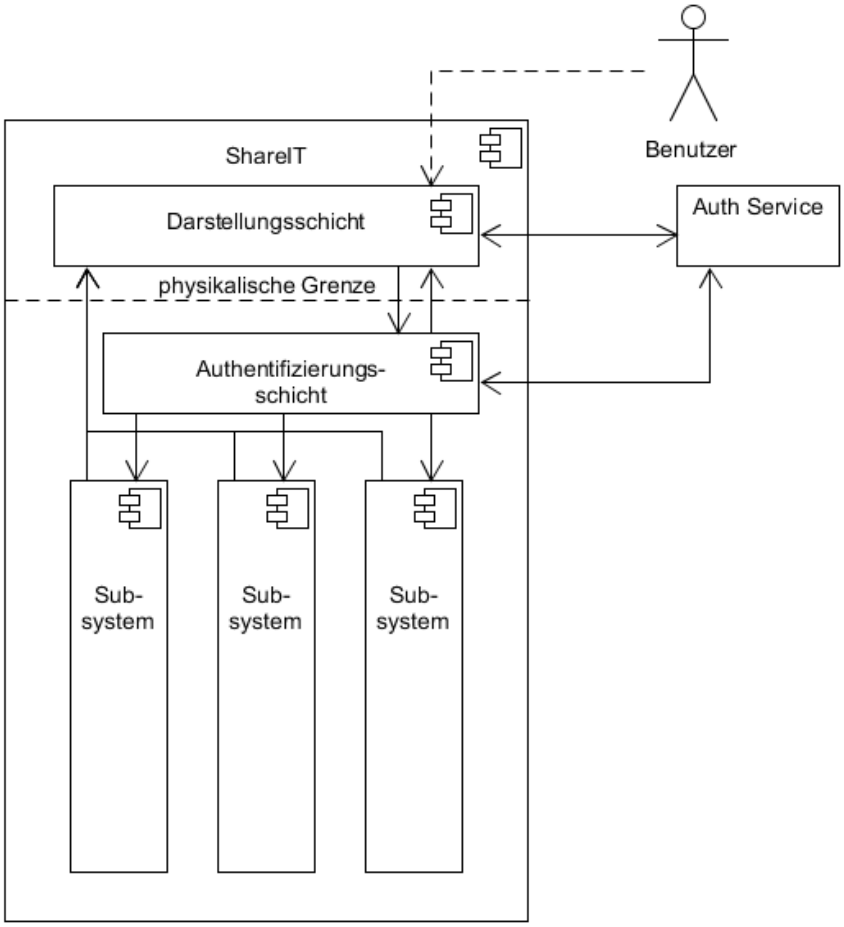
https://github.com/abcshmedu/shareit-summer-2017-ezscheineteam
https://github.com/abcshmedu/a4-auth-server-ezscheineteam

Lösungsstrategie

Um eine iterative Entwicklung des Softwaresystems zu ermöglichen und das System auch mittelfristig wartbar und erweiterbar zu erhalten, soll die logische Architektur der Software modular aufgebaut sein. Grundlage dafür ist eine sukzessive Gliederung in fachliche Domänen bzw. Subsysteme sowie in Schichten. Zusätzlich wird die Anwendung mit einem REST-API zwischen Web-Browser und Server-Infrastruktur implementiert

Bausteinsicht

Whitebox Gesamtsystem



Darstellungsschicht

Die Darstellungsschicht (Presentation Layer) kapselt die Benutzungsschnittstelle des Softwaresystems. Sie bildet für den Benutzer den Einstiegspunkt in die Bedienung des Softwaresystems. Ausgehend von den Aktionen des Benutzers auf dieser Benutzungsschnittstelle wie z.B. der Auswahl eines bestimmten Anwendungsfalles werden die dafür erforderlichen Funktionalitäten der API-Schicht angestoßen. Anfallende Ergebnisdaten werden von der API-Schicht an die Darstellungsschicht

übergeben, von dieser in geeigneter Form aufbereitet und anschließend dargestellt. Das Softwaresystem ShareIt wird ebenfalls gemäß dieser Konvention als Mehrschichtarchitektur realisiert.

Authentifizierungsschicht

Der Authentifizierungsschicht prüft bei jeder eingehenden Anfrage die Authentifizierung des Benutzers und gibt zusätzliche Informationen über diesen an den Subsystemen weiter. Bei ungültiger Authentifizierung wird eine Fehlermeldung zurückgegeben.

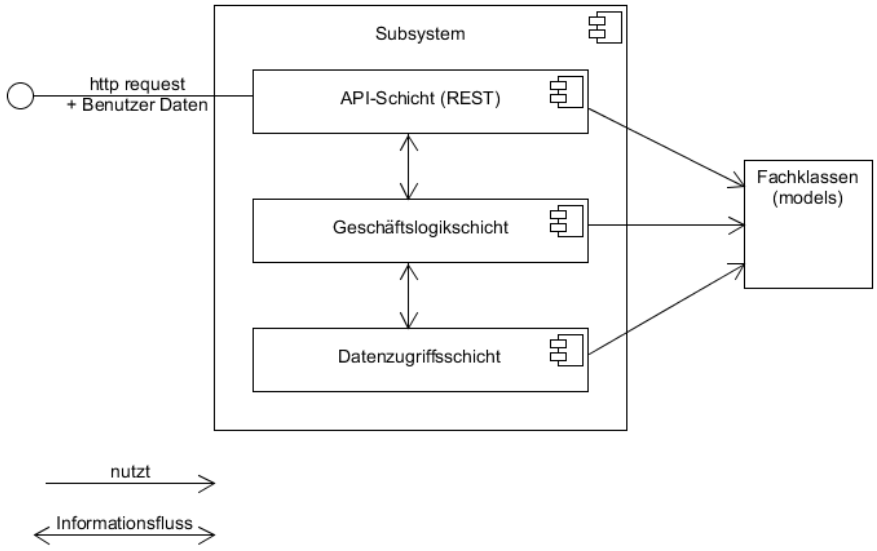
Subsysteme

Subsysteme gliedern ein Softwaresystem vertikal nach fachlichen Gesichtspunkten. Dazu werden diejenigen Anwendungsfälle, die thematisch eng miteinander gekoppelt sind (in einer fachlichen Domäne liegen), in ein Subsystem gebündelt. Alternativ wäre es auch denkbar, das Softwaresystem auf der Grundlage des fachlichen Klassenmodells in Subsysteme zu gliedern.

Die vertikale Gliederung von ShareIt nach thematisch zusammenhängenden Anwendungsfällen ergibt zunächst die folgenden drei fachlichen Domänen:

- Benutzerverwaltung
- Verwaltung von Medien und Exemplaren
- Ausleihe und Rückgabe

Ebene 2



Schichten

Durch eine Einteilung in Schichten wird das Softwaresystem horizontal gegliedert in verschiedene Abstraktionsebenen. Jede Schicht ist dabei Dienstanbieter für die darüber liegende Schicht und nutzt selbst nur diejenigen Dienste der unmittelbar darunter liegenden Schicht. Die Grenze zwischen den Schichten ist so zu ziehen, dass m¨oglichst wenige Abhängigkeiten über die Schichtgrenzen hinweg bestehen. Ein wesentliches Ziel der Gliederung in Schichten besteht darin, die Implementierung einer einzelnen Schicht auszutauschen. Die Schnittstellen dieser Schicht bleiben dabei unverändert. Die Abhängigkeiten zwischen den Schichten werden über Schnittstellen gekapselt, sodass die Implementierung jeder einzelnen Schicht austauschbar ist. Modelklassen (Fachklassen) dürfen von allen Schichten, also auch über Schichtengrenzen hinweg, verwendet werden.

API-Schicht

Die API-Schicht stellt die Kommunikations-Endpunkte zur Verfügung für Mensch-Maschine (Browser-basiert) und Maschine-Maschine-Interaktionen. Hier werden insbesondere Objekte serialisiert und deserialisiert.

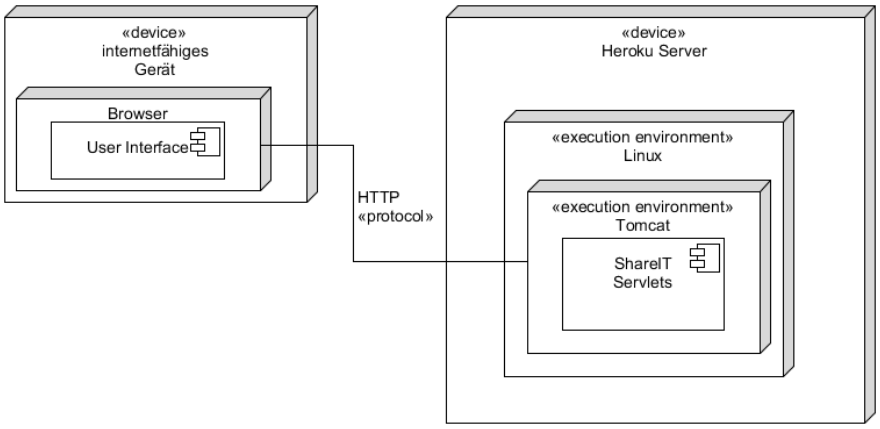
Geschäftslogikschicht

Auf der nächsten Abstraktionsebene liegt die Geschäftslogikschicht (Business Layer), die sich auf der Datenzugriffsschicht abstützt. Die Geschäftslogikschicht beinhaltet die grundlegende Geschäftslogik zur Verarbeitung der Fachklassen sowie eine Implementierung der Workflows bzw. Arbeitsabläufe des Systems.

Datenzugriffsschicht

Die Datenzugriffsschicht (Persistence Layer) realisiert die Fachklassen sowie den Zugriff auf diejenige Technologie, die zur persistenten Datenhaltung für diese Fachklassen verwendet wird, beispielsweise eine relationale Datenbank. (Die Technologie selbst ist dabei nicht Bestandteil der Datenzugriffsschicht.) Ebenfalls in der Datenzugriffsschicht angesiedelt ist die Abbildung von Objekten auf Strukturen einer Datenbank. Das umfasst beispielsweise das OR-Mapping, also die Abbildung der Objektstruktur der Fachklassen auf relationale Datenbanktabellen, sofern eine relationale Datenbank eingesetzt wird.

Verteilungssicht



Das User Interface des Systems wird auf dem Gerät des Benutzers innerhalb eines Browsers laufen. Da diese in JavaScript geschrieben sein soll wird keine Abhängigkeit erzeugt. Einzigste Voraussetzung an das Gerät ist eine bestehende Internetverbindung.

Die restlichen Bausteine des Systems werden auf eine Webservice (PaaS) mit dem Namen Heroku (https://www.heroku.com/) deployed um von verschiedenen Orten online abrufbar zu sein. Für Testzwecke ist ein lokales ausführen auch denkbar.

Querschnittliche Konzepte

Persistenz

Zum dauerhaften speichern von Informationen werden nur die Fachklassen mittels Nutzung von Hibernate als Object/Relational Mapper auf einer Datenbank gesichert. Ausschließlich die Datenzugriffsschicht kümmert sich um die Verwaltung der Datenbank. Als Datenbank verwenden wir HSQL welche im lokalen Ordner db gesichert wird.

Logging

Um eine hilfsreiche Informationsquelle für Entwickler und Administratoren zum Verhalten des Systems zu liefern wird das Logging Framework log4j verwendet

Authentifizierung

Jeder Benutzer der das System verwenden will muss sich gegenüber diesem sich authentifizieren. Dafür benötigt der Benutzer ein Token welches er von einem Fremdsystem, den Authentifizierung Server, bei Angabe von Benutzername und Passwort erhält. Das Token muss bei jeder Anfrage an das System im HTTP Feld UserToken befinden und gültig sein da sonst die Anfrage mit einer Fehlermeldung abgewiesen wird.

Fehlermeldungen

Jede falsche oder ungültige Verwendung des Systems muss mit einer Fehlermeldung als JSON-Objekt beantwortet werden. Dabei besitzt das JSON-Objekt ein Feld status mit einem HTTP Fehlercode und ein Feld detail mit einer Nachricht die dem durchschnittlichen Benutzer den Fehler verständlich mitteilt. Beide Statuscodes von HTTP Response und JSON-Objekt müssen übereinstimmen

Risiken und technische Schulden

Datenschutz

Da bei Informationsaustausch auf das verschlüsselnde Protokoll HTTPS verzichtet wird und stattdessen das unverschlüsselte Protokoll HTTP verwendet wird können sehr leicht diese Daten von Dritten abgegriffen werden und sogar dann dazu missbraucht werden sich als eine andere Person gegenüber dem System zu identifizieren.

Glossar

Begriff	Definition
HTML	Eine textbasierte Auszeichnungssprache zur Strukturierung digitaler Dokumente wie Texte mit Hyperlinks, Bildern und anderen Inhalten.eine textbasierte Auszeichnungssprache zur Strukturierung digitaler Dokumente wie Texte mit Hyperlinks, Bildern und anderen Inhalten.
HTTP	HyperText Transfer Protocol. Ein zustandsloses Protokoll zur Übertragung von Daten auf der Anwendungsschicht über ein Rechnernetz.
HTTPS	HyperText Transfer Protocol Secure. Ein Kommunikationsprotokoll im World Wide Web, um Daten abhörsicher zu übertragen.
JSON	JavaScript Object Notation. Ein kompaktes Datenformat in einer einfach lesbaren Textform zum Zweck des Datenaustauschs zwischen Anwendungen.
Servlet	Java-Klassen, deren Instanzen innerhalb eines Webservers Anfragen von Clients entgegennehmen und beantworten.
PaaS	Platform as a Service. Eine Dienstleistung, die in der Cloud eine Computer-Plattform für Entwickler von Webanwendungen zur Verfügung stellt.