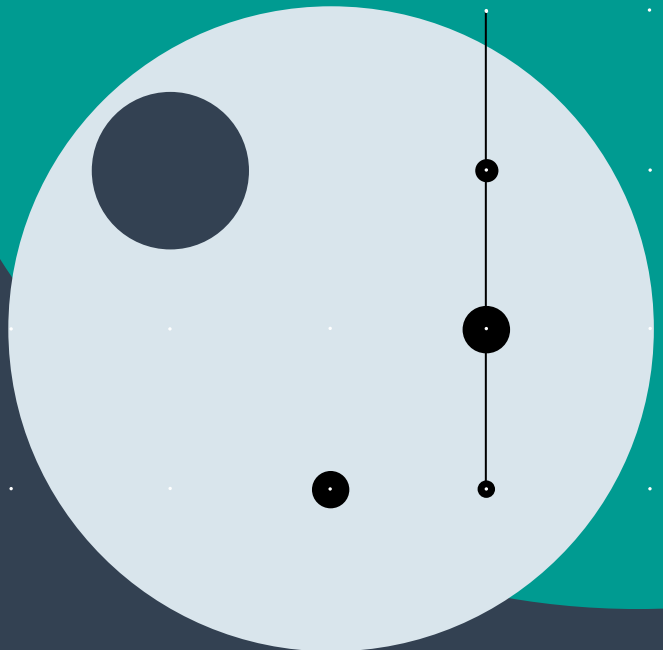
Hochschule Konstanz
Fakultät Informatik

Teamprojekt SS 20 - WS 20/21

Stefan Schmunk, 295387
Patrick Sterk, 294849
Pascal Ulmer, 301108
Lucas Schmidt, 294457
Philip Benischke, 287519



Inhalt

Ziele des Teamprojekts

Stand etwa Anfang Mai 2020:

- Concept-Editor auf yFiles umstellen
- Model-Editor auf yFiles umstellen
- Code-Generator anpassen
- Code-Editor anpassen

Stand vor 12.05.2020

Kein funktionierendes Backend => kein lauffähiges Projekt

Stand vor 12.05.2020

Wiki:

- Veraltete Beschreibung.
  - Nicht mehr enthaltene Features.
- Fehlerhafte Beispiele.
  - DSL stimmt nicht.
  - Beispiele funktionieren nicht.

Stand vor 12.05.2020

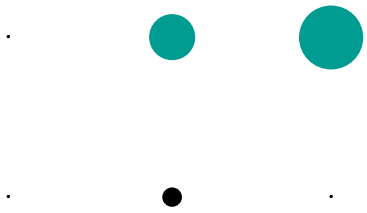
Backend:

- Java auf Version 8 limitiert.
  - Debian 10.7 stellt min. Java 11 bereit.
- Extra Installation von JavaFX.
  - Nicht Teil des OpenJDK.
  - Wird nicht von sbt abgedeckt.
  - Schwierig für Windows zu finden.
  - Muss vom Benutzer selbständig installiert werden.
- Kaputte Abhängigkeiten durch.
  - Namensänderung der Pakete.
  - Verschoben in andere Repositories.
  - Deprecated (Ersetzt durch andere Pakete).
  - Bereits in verwendeten Paketen integriert.

Stand vor 12.05.2020

Backend:

- Abweichung der internen Abhängigkeit der Pakete.
  - Bsp.: Akka verwendet Scala 2.11 und ScalaTests 2.12.
    - Dependency Tree Conflict.
- Tests haben nicht mehr funktioniert.
  - Module konnten nicht mehr gebaut werden.
- Versionsstand auf 2017.
- Kaputter Container.
  - Verweise auf Versionen im Image nicht mehr verfügbar.
  - Zur Laufzeit abstürze, durch enorme Speichernutzung.



Stand vor 12.05.2020

Frontend:

- Veraltete Abhängigkeiten.
  - Kein Browser-Support.
  - Deprecated.
  - Sicherheitsprobleme.
- Kein funktionierendes Interface.
  - Bootstrap fehlerhaft.
  - Buttons/Dropdown usw. nicht verwendbar.
- Fehlerhaftes Bundle.
  - Keine Bereitstellung der Artefakte.
  - Falsches Bundlen der Artefakte.
- Frontend im Frontend.
  - Unabhängiges Vue Projekt im Frontend.
  - Bruch der Architektur.
- Kaputte Komponenten:
  - Texteditor, Grafikeditor, Code-Editor

Stand vor 12.05.2020

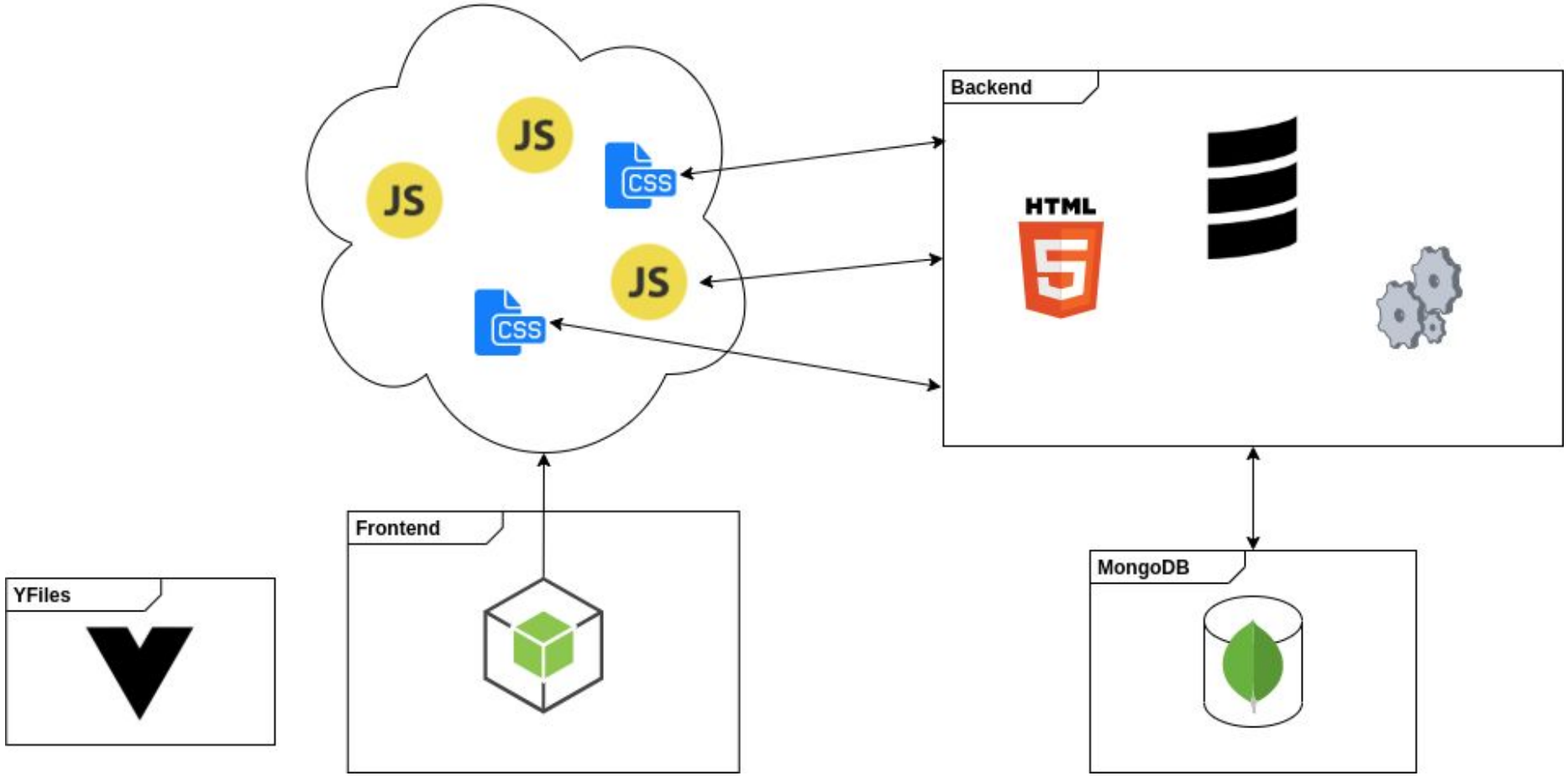
Frontend:

- yFiles direkt im Frontend.
  - Keine legale Verwendung möglich. (Lizenz)
  - Muss von Hand verwaltet werden.
- Falsche Verweise in der Struktur.
  - Verlinkungen laufen ins leere.
  - Verlinkung auf nicht vorhandene Abhängigkeiten.
- Veralteter Code
  - Alte (nicht mehr verwendete) Komponenten noch vorhanden.
  - Unstrukturierte Komponente.
    - Keine klare Domänen-Bezeichnung.
    - Fachliche und Technische Bezeichnung vertauscht.
    - Uneindeutige Verwendung von Komponenten.

Hochschule Konstanz
Fakultät Informatik

Systemarchitektur

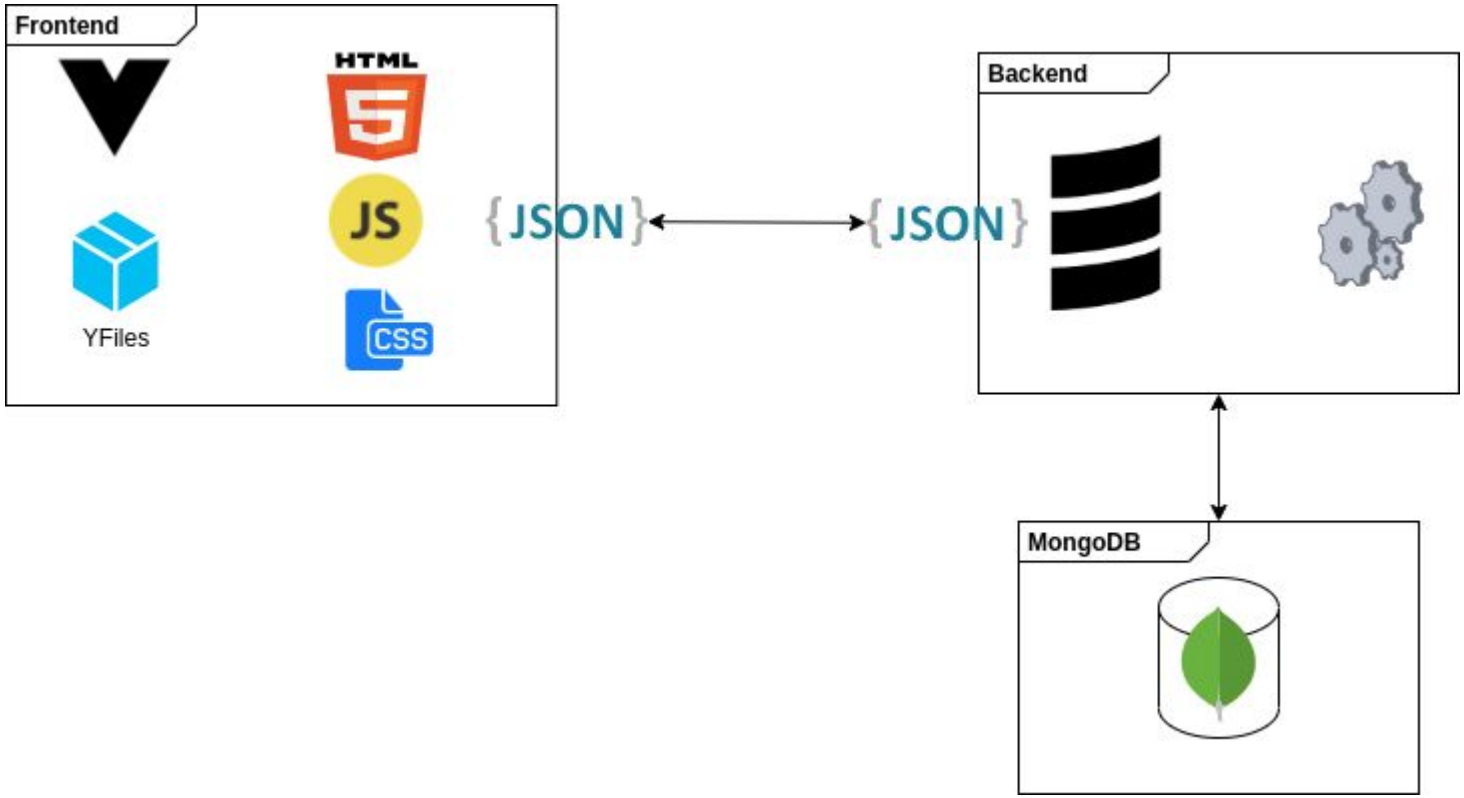
Alte Systemarchitektur



Nachteile der alten Architektur

- JavaScript und CSS Bundles sind sehr schwer zu debuggen und entwickeln
- Entwicklung der UI ist nur möglich wenn Backend und Frontend laufen
- Sehr lange Bauzeit des Frontends da alle JavaScript und CSS Dateien in Bundles gepackt werden müssen
- Sehr lange Ladezeit, da für jede Navigation JavaScript und CSS bundles angefragt werden müssen
- Sehr fragil, da ein Fehler in einer JavaScript Datei, das komplette Bundle und damit das Projekt zerstört
- YFiles ist ein eigenständiges Vue-Projekt und kann daher nur mit sehr großem Aufwand (Verwendung von Node-Vue-Loadern) im Frontend verwendet werden
- Backend ist nicht lauffähig
- Frontend nur nach Fixes lauffähig
- Backend routet alle Anfragen durch
- Hohe Komplexität bei der Fehlerfindung
    - Backend könnte noch altes Bundle im Cache haben

Neue Systemarchitektur



Vorteile der neuen Architektur

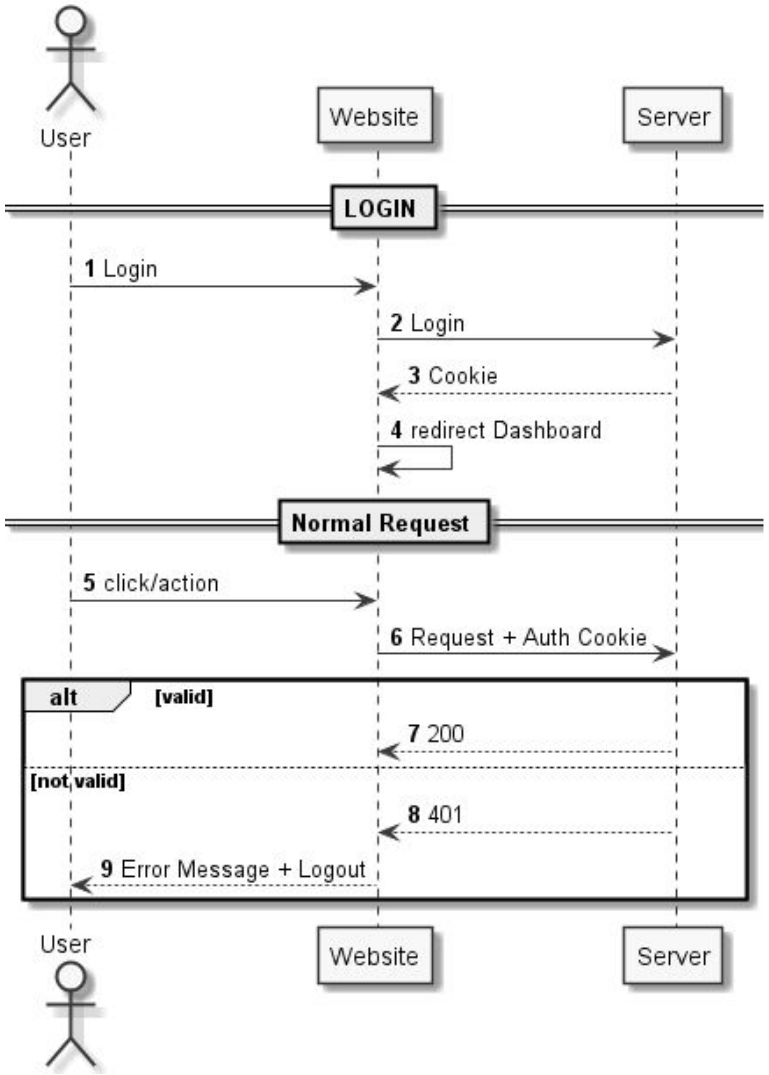
- Das Frontend wird nun auch als eigenständiges Frontend verwendet (und nicht als Artefakt-Server wie zuvor)
- Das Backend liefert nun JSON-Objekte
- HTML, JavaScript und CSS sind nun alle in ein eigenständiges Frontend Projekt verlagert worden und wurden komplett in Vue.js umgeschrieben
- Model und View sind nun strikt voneinander getrennt und können so besser entwickelt und debugged werden
- Durch das Migrieren in Vue.js konnte das bestehende YFiles Projekt leichter integriert werden
- Das Projekt ist beim Starten der App, sowie beim Laden der Webseite und beim Navigieren wesentlich schneller
- Das Frontend unterteilt sich in Komponenten, die unabhängig von anderen Komponenten arbeiten und leicht ausgetauscht, oder umgeschrieben werden können
- Skalierung des Frontend möglich.
- Unterschiedliche Frontends (z.B. Mobile-Version) können entwickelt werden, mit dem selben Backend.

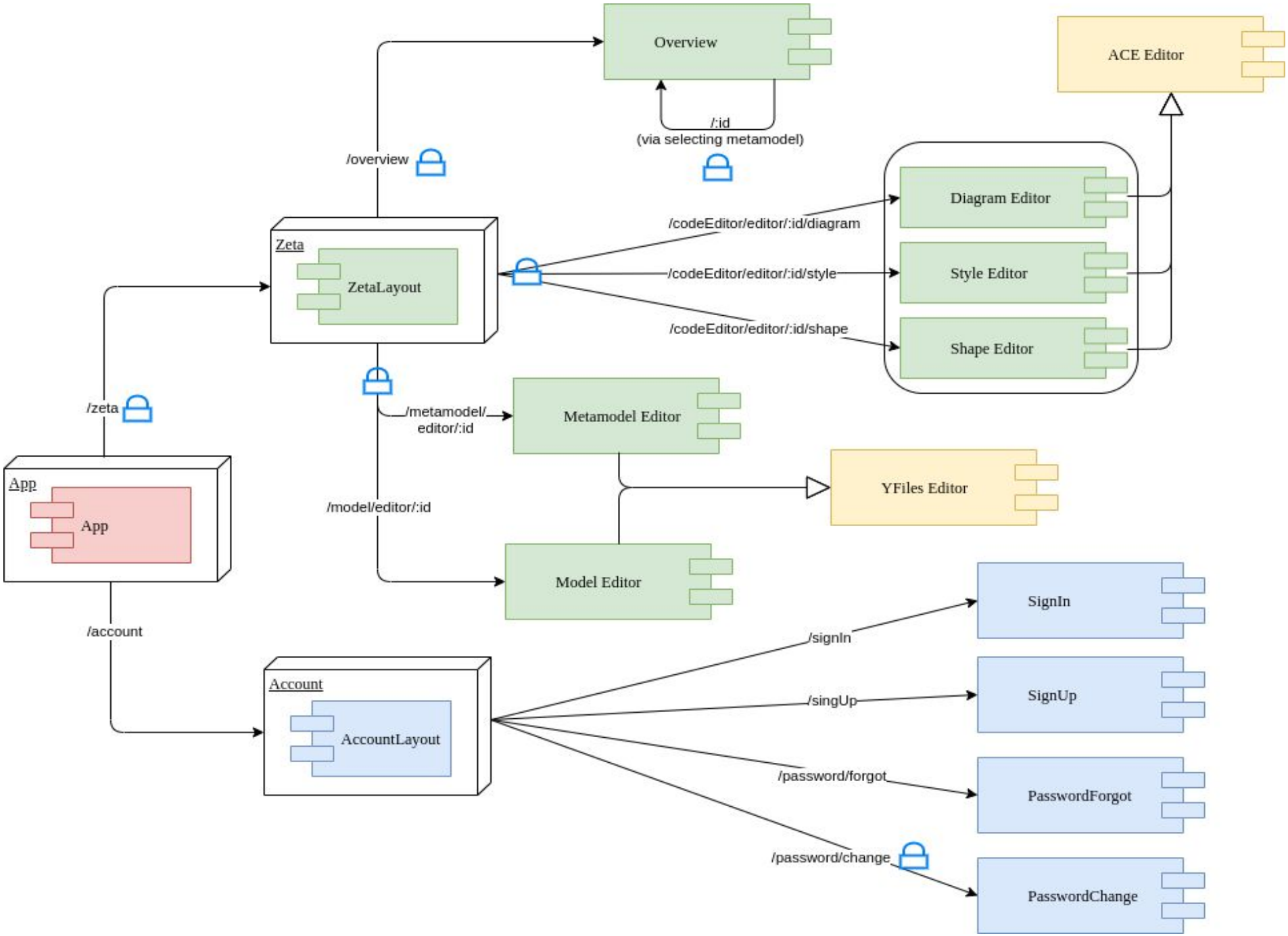
Hochschule Konstanz
Fakultät Informatik

Frontend

Login

- Anbindung Webseite an Silhouette
- Setzen von Cookie
- Handling in der Applikation




Overview
/:id
(via selecting metamodel)
ACE Editor
/overview
Diagram Editor
/codeEditor/editor/:id/diagram
Zeta
ZetaLayout
/codeEditor/editor/:id/style
Style Editor
/codeEditor/editor/:id/shape
Shape Editor
/zeta
/metamodel/
editor/:id
Metamodel Editor
App
App
YFiles Editor
/model/editor/:id
Model Editor
/account
SignIn
/signIn
Account
AccountLayout
/singUp
SignUp
/password/forgot
PasswordForgot
/password/change
PasswordChange

Vuetify - Framework

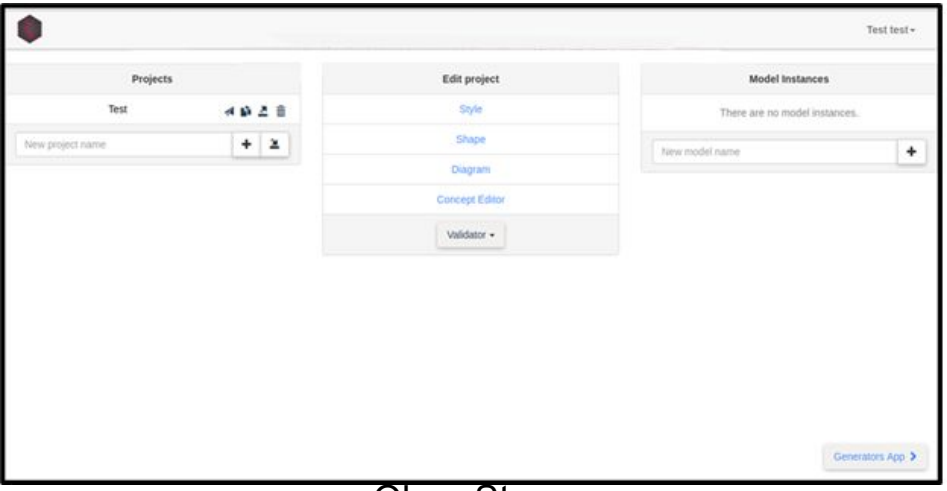
- UI-Framework mit bereits fertigen UI-Komponenten
- Geeignet für Web- und mobile Anwendungen
- Erfordert keine Design Kenntnisse
- Einfach und schnell zu verwenden

Vuetify
Material Component Framework

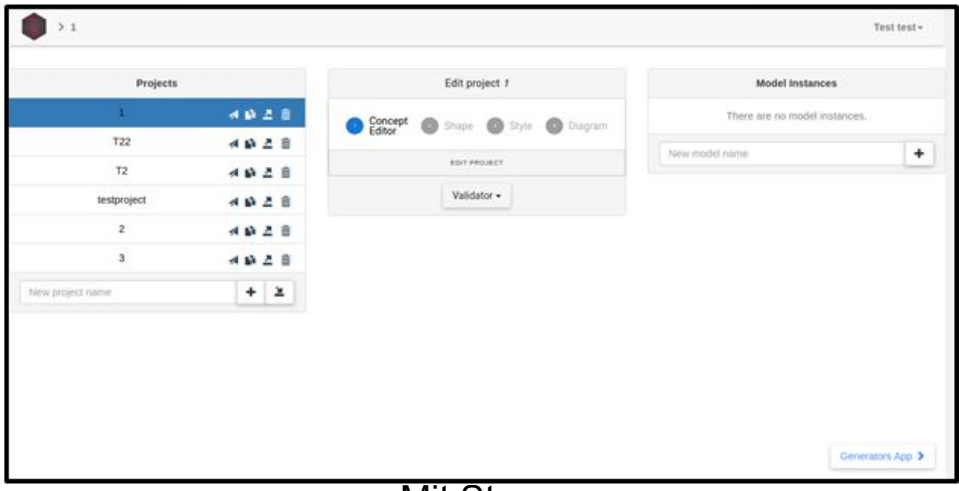
- Einpflegen der Abhängigkeiten in Zeta
- Einführung des neuen Teams in die Welt des Vuetifys
- Erweiterung der “Zeta-Technologies” im GitHub - Wiki

Zeta-Overview um Stepper erweitert

- Alte Overview: Unübersichtlich und durcheinander
- Neue Overview: Intuitiven Workflow bei der Erstellung eines neuen Projekts



Ohne Stepper

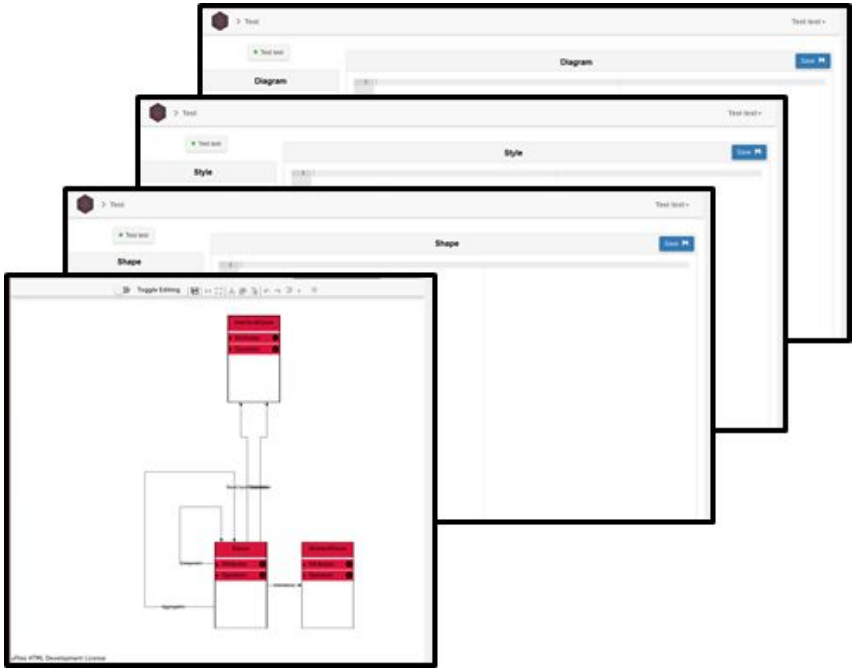


Mit Stepper

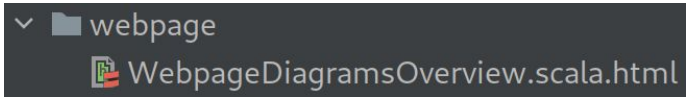
Hochschule Konstanz
Fakultät Informatik

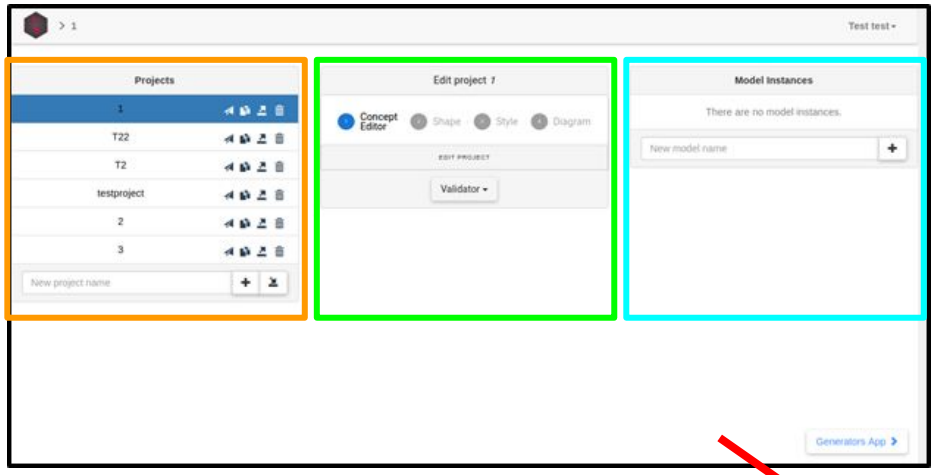
Stepper - Step by Step

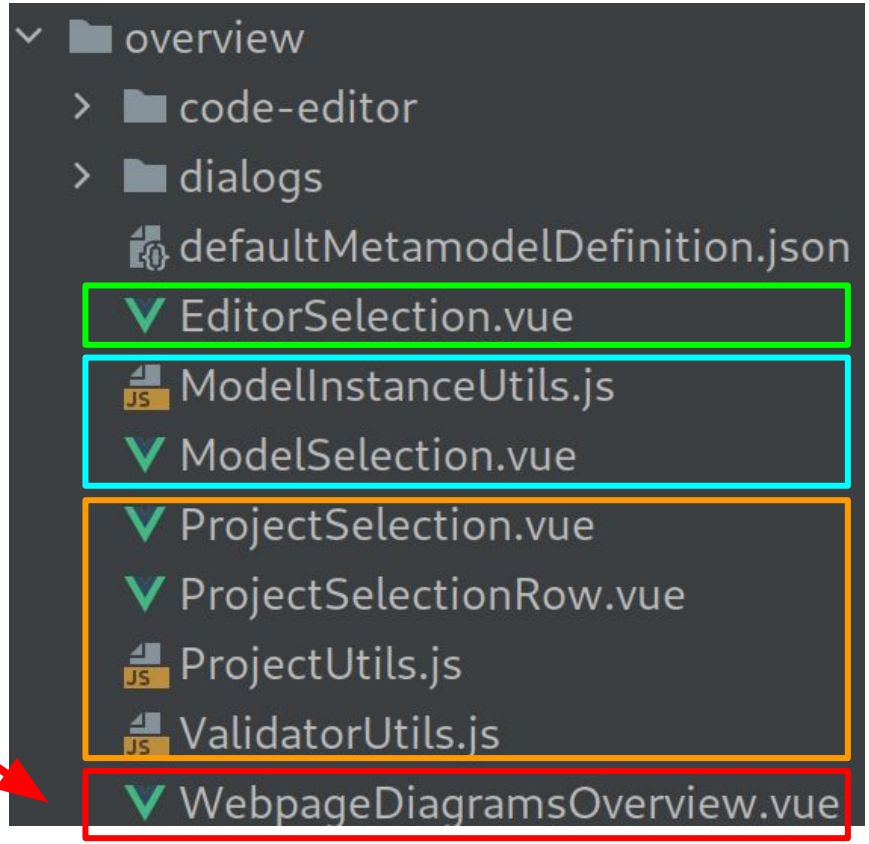
- Keine Ladezeiten mehr für das Öffnen der einzelnen Editoren in einer eigenen Seite
- Bessere Übersicht über die einzelnen Komponenten
- Gleichzeitiges Bearbeiten der einzelnen Komponenten innerhalb einer Seite



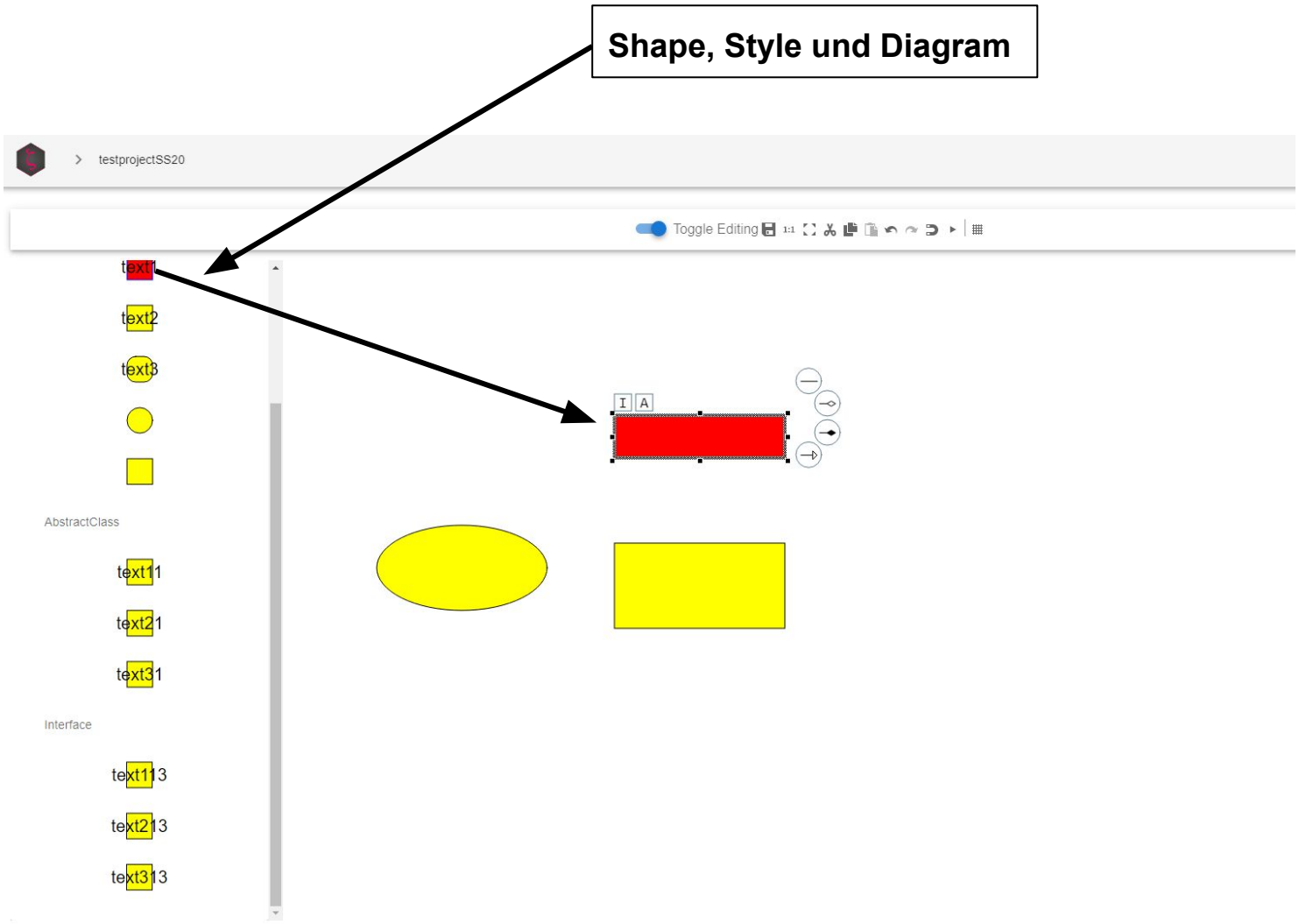
Modulare Overview





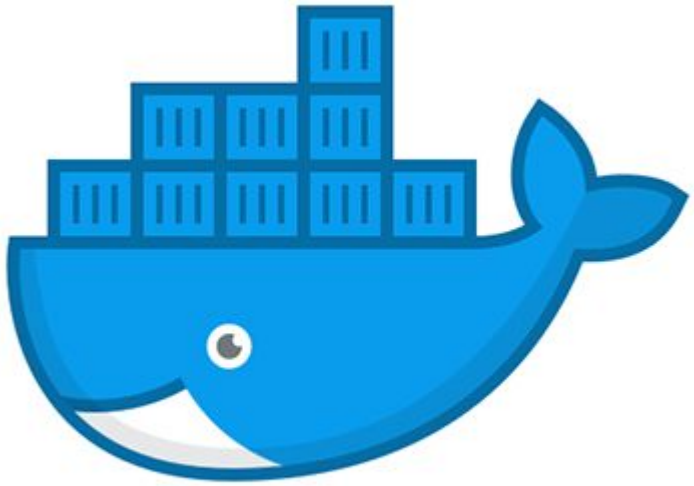


Model Editor



Frontend-Docker

- Neueste Node Version
- Y-Files Abhängigkeiten
- Liefert Webseite aus
- Wird in docker-compose referenziert



Hochschule Konstanz
Fakultät Informatik

Backend

Hochschule Konstanz
Fakultät Informatik

Backend Code-Coverage

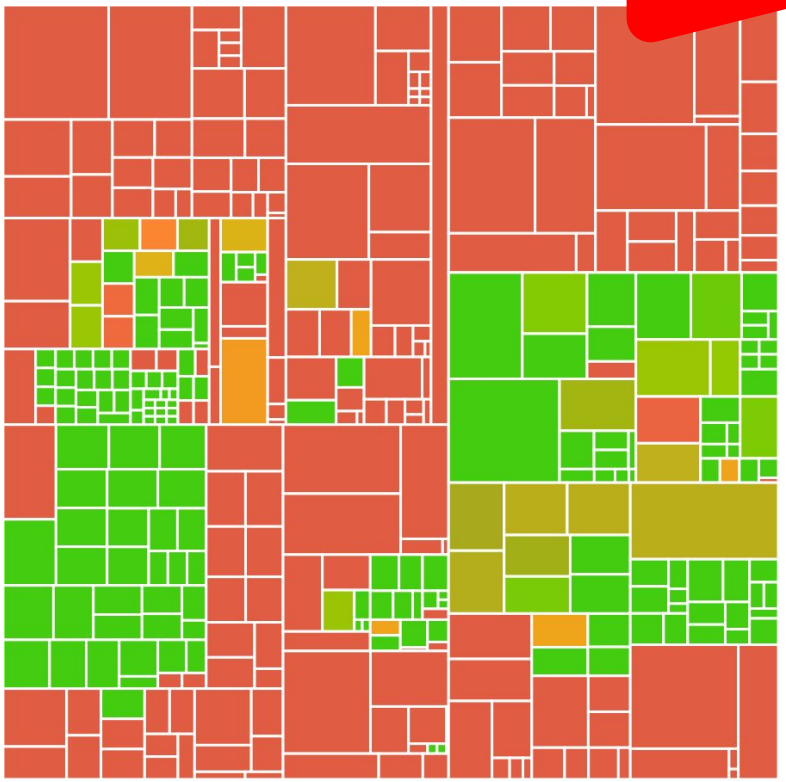
Master (Mit Dep Fix)

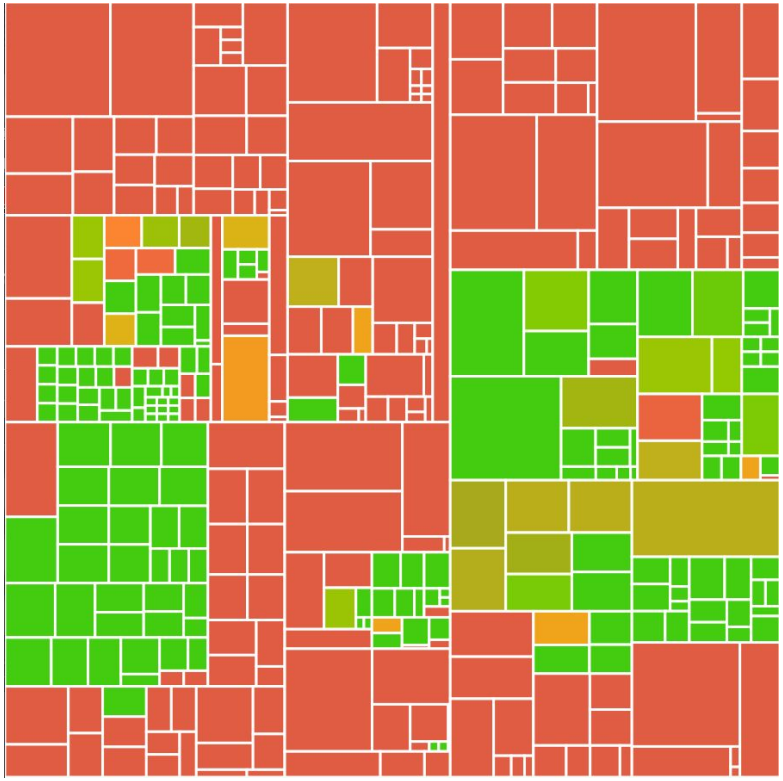
37.88%

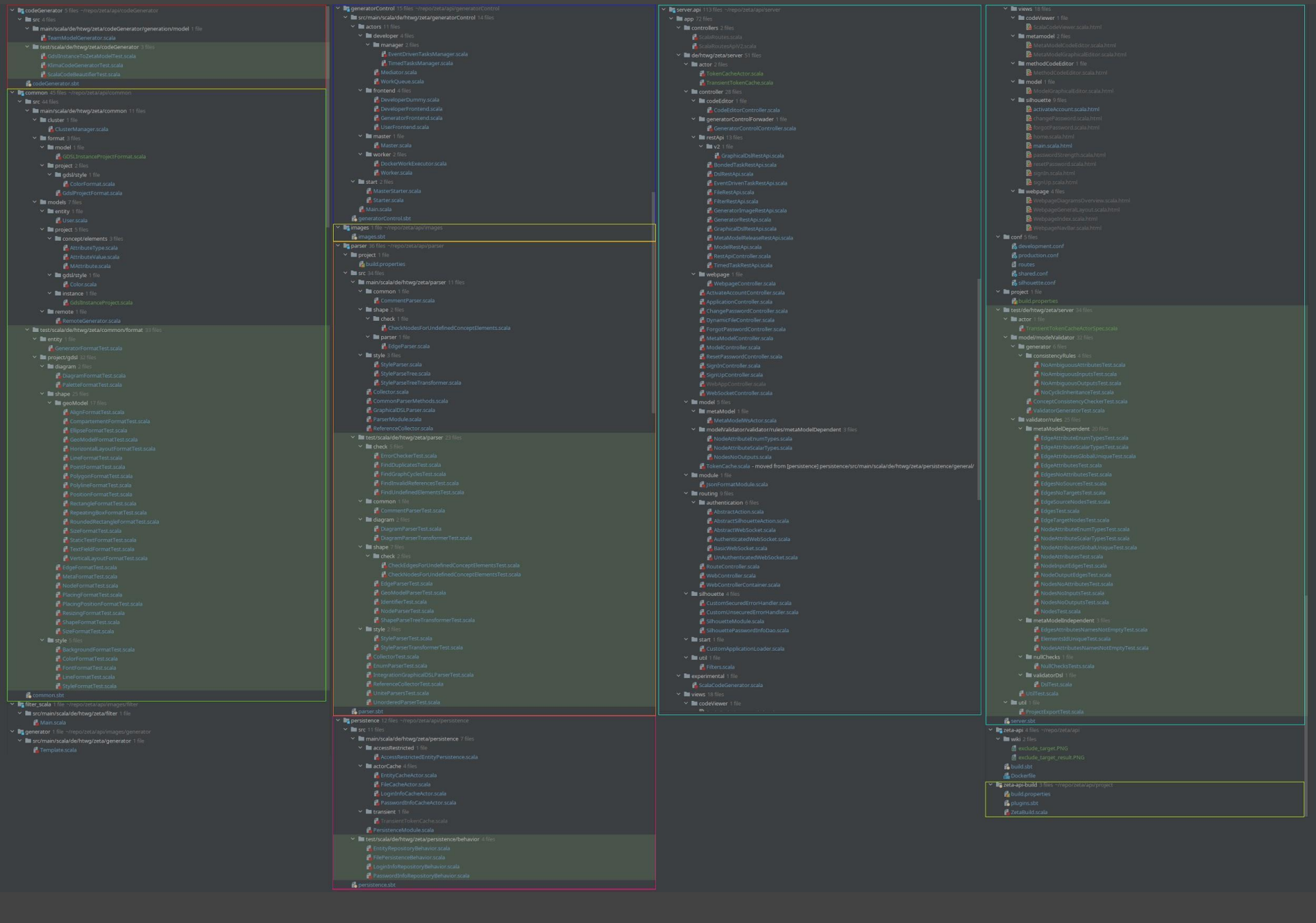
+0,32%

Dev (13.01.2021)

38.20%






codeGenerator 5 files ~/repo/zeta/api/codeGenerator
src 4 files
main/scala/de/htwg/zeta/codeGenerator/generation/model 1 file
TeamModelGenerator.scala
test/scala/de/htwg/zeta/codeGenerator 3 files
GdslInstanceToZetaModelTest.scala
KlimaCodeGeneratorTest.scala
ScalaCodeBeautifierTest.scala
codeGenerator.sbt
common 45 files ~/repo/zeta/api/common
src 44 files
main/scala/de/htwg/zeta/common 11 files
cluster 1 file
ClusterManager.scala
format 3 files
model 1 file
GDSLInstanceProjectFormat.scala
project 2 files
gdsl/style 1 file
ColorFormat.scala
GdslProjectFormat.scala
models 7 files
entity 1 file
User.scala
project 5 files
concept/elements 3 files
AttributeType.scala
AttributeValue.scala
MAttribute.scala
gdsl/style 1 file
Color.scala
instance 1 file
GdslInstanceProject.scala
remote 1 file
RemoteGenerator.scala
test/scala/de/htwg/zeta/common/format 33 files
entity 1 file
GeneratorFormatTest.scala
project/gdsl 32 files
diagram 2 files
DiagramFormatTest.scala
PaletteFormatTest.scala
shape 25 files
geoModel 17 files
AlignFormatTest.scala
CompartementFormatTest.scala
EllipseFormatTest.scala
GeoModelFormatTest.scala
HorizontalLayoutFormatTest.scala
LineFormatTest.scala
PointFormatTest.scala
PolygonFormatTest.scala
PolylineFormatTest.scala
PositionFormatTest.scala
RectangleFormatTest.scala
RepeatingBoxFormatTest.scala
RoundedRectangleFormatTest.scala
SizeFormatTest.scala
StaticTextFormatTest.scala
TextFieldFormatTest.scala
VerticalLayoutFormatTest.scala
EdgeFormatTest.scala
MetaFormatTest.scala
NodeFormatTest.scala
PlacingFormatTest.scala
PlacingPositionFormatTest.scala
ResizingFormatTest.scala
ShapeFormatTest.scala
SizeFormatTest.scala
style 5 files
BackgroundFormatTest.scala
ColorFormatTest.scala
FontFormatTest.scala
LineFormatTest.scala
StyleFormatTest.scala
common.sbt
filter_scala 1 file ~/repo/zeta/api/images/filter
src/main/scala/de/htwg/zeta/filter 1 file
Main.scala
generator 1 file ~/repo/zeta/api/images/generator
src/main/scala/de/htwg/zeta/generator 1 file
Template.scala
generatorControl 15 files ~/repo/zeta/api/generatorControl
src/main/scala/de/htwg/zeta/generatorControl 14 files
actors 11 files
developer 4 files
manager 2 files
EventDrivenTasksManager.scala
TimedTasksManager.scala
Mediator.scala
WorkQueue.scala
frontend 4 files
DeveloperDummy.scala
DeveloperFrontend.scala
GeneratorFrontend.scala
UserFrontend.scala
master 1 file
Master.scala
worker 2 files
DockerWorkExecutor.scala
Worker.scala
start 2 files
MasterStarter.scala
Starter.scala
Main.scala
generatorControl.sbt
images 1 file ~/repo/zeta/api/images
images.sbt
parser 32 files ~/repo/zeta/api/parser
project 1 file
build.properties
src 34 files
main/scala/de/htwg/zeta/parser 11 files
common 1 file
CommentParser.scala
shape 2 files
check 1 file
CheckNodesForUndefinedConceptElements.scala
parser 1 file
EdgeParser.scala
style 3 files
StyleParser.scala
StyleParseTree.scala
StyleParseTreeTransformer.scala
Collector.scala
CommonParserMethods.scala
GraphicalDSLParser.scala
ParserModule.scala
ReferenceCollector.scala
test/scala/de/htwg/zeta/parser 23 files
check 8 files
ErrorCheckerTest.scala
FindDuplicatesTest.scala
FindGraphCyclesTest.scala
FindInvalidReferencesTest.scala
FindUndefinedElementsTest.scala
common 1 file
CommentParserTest.scala
diagram 2 files
DiagramParserTest.scala
DiagramParserTransformerTest.scala
shape 7 files
check 2 files
CheckEdgesForUndefinedConceptElementsTest.scala
CheckNodesForUndefinedConceptElementsTest.scala
EdgeParserTest.scala
GeoModelParserTest.scala
IdentifierTest.scala
NodeParserTest.scala
ShapeParseTreeTransformerTest.scala
style 2 files
StyleParserTest.scala
StyleParserTransformerTest.scala
CollectorTest.scala
EnumParserTest.scala
IntegrationGraphicalDSLParserTest.scala
ReferenceCollectorTest.scala
UniteParsersTest.scala
UnorderedParserTest.scala
parser.sbt
persistence 12 files ~/repo/zeta/api/persistence
src 11 files
main/scala/de/htwg/zeta/persistence 7 files
accessRestricted 1 file
AccessRestrictedEntityPersistence.scala
actorCache 4 files
EntityCacheActor.scala
FileCacheActor.scala
LoginInfoCacheActor.scala
PasswordInfoCacheActor.scala
transient 1 file
TransientTokenCache.scala
PersistenceModule.scala
test/scala/de/htwg/zeta/persistence/behavior 4 files
EntityRepositoryBehavior.scala
FilePersistenceBehavior.scala
LoginInfoRepositoryBehavior.scala
PasswordInfoRepositoryBehavior.scala
persistence.sbt
server.api 113 files ~/repo/zeta/api/server
app 72 files
controllers 2 files
ScalaRoutes.scala
ScalaRoutesApiV2.scala
de/htwg/zeta/server 51 files
actor 2 files
TokenCacheActor.scala
TransientTokenCache.scala
controller 28 files
codeEditor 1 file
CodeEditorController.scala
generatorControlForwader 1 file
GeneratorControlController.scala
restApi 13 files
v2 1 file
GraphicalDslRestApi.scala
BondedTaskRestApi.scala
DslRestApi.scala
EventDrivenTaskRestApi.scala
FileRestApi.scala
FilterRestApi.scala
GeneratorImageRestApi.scala
GeneratorRestApi.scala
GraphicalDslRestApi.scala
MetaModelReleaseRestApi.scala
ModelRestApi.scala
RestApiController.scala
TimedTaskRestApi.scala
webpage 1 file
WebpageController.scala
ActivateAccountController.scala
ApplicationController.scala
ChangePasswordController.scala
DynamicFileController.scala
ForgotPasswordController.scala
MetaModelController.scala
ModelController.scala
ResetPasswordController.scala
SignInController.scala
SignUpController.scala
WebAppController.scala
WebSocketController.scala
model 5 files
metaModel 1 file
MetaModelWsActor.scala
modelValidator/validator/rules/metaModelDependent 3 files
NodeAttributeEnumTypes.scala
NodeAttributeScalarTypes.scala
NodesNoOutputs.scala
TokenCache.scala - moved from [persistence] persistence/src/main/scala/de/htwg/zeta/persistence/general/
module 1 file
JsonFormatModule.scala
routing 9 files
authentication 6 files
AbstractAction.scala
AbstractSilhouetteAction.scala
AbstractWebSocket.scala
AuthenticatedWebSocket.scala
BasicWebSocket.scala
UnAuthenticatedWebSocket.scala
RouteController.scala
WebController.scala
WebControllerContainer.scala
silhouette 4 files
CustomSecuredErrorHandler.scala
CustomUnsecuredErrorHandler.scala
SilhouetteModule.scala
SilhouettePasswordInfoDao.scala
start 1 file
CustomApplicationLoader.scala
util 1 file
Filters.scala
experimental 1 file
ScalaCodeGenerator.scala
views 18 files
codeViewer 1 file
views 18 files
codeViewer 1 file
ScalaCodeViewer.scala.html
metamodel 2 files
MetaModelCodeEditor.scala.html
MetaModelGraphicalEditor.scala.html
methodCodeEditor 1 file
MethodCodeEditor.scala.html
model 1 file
ModelGraphicalEditor.scala.html
silhouette 9 files
activateAccount.scala.html
changePassword.scala.html
forgotPassword.scala.html
home.scala.html
main.scala.html
passwordStrength.scala.html
resetPassword.scala.html
signIn.scala.html
signUp.scala.html
webpage 4 files
WebpageDiagramsOverview.scala.html
WebpageGeneralLayout.scala.html
WebpageIndex.scala.html
WebpageNavBar.scala.html
conf 5 files
development.conf
production.conf
routes
shared.conf
silhouette.conf
project 1 file
build.properties
test/de/htwg/zeta/server 34 files
actor 1 file
TransientTokenCacheActorSpec.scala
model/modelValidator 32 files
generator 6 files
consistencyRules 4 files
NoAmbiguousAttributesTest.scala
NoAmbiguousInputsTest.scala
NoAmbiguousOutputsTest.scala
NoCyclicInheritanceTest.scala
ConceptConsistencyCheckerTest.scala
ValidatorGeneratorTest.scala
validator/rules 25 files
metaModelDependent 20 files
EdgeAttributeEnumTypesTest.scala
EdgeAttributeScalarTypesTest.scala
EdgeAttributesGlobalUniqueTest.scala
EdgeAttributesTest.scala
EdgesNoAttributesTest.scala
EdgesNoSourcesTest.scala
EdgesNoTargetsTest.scala
EdgeSourceNodesTest.scala
EdgesTest.scala
EdgeTargetNodesTest.scala
NodeAttributeEnumTypesTest.scala
NodeAttributeScalarTypesTest.scala
NodeAttributesGlobalUniqueTest.scala
NodeAttributesTest.scala
NodeInputEdgesTest.scala
NodeOutputEdgesTest.scala
NodesNoAttributesTest.scala
NodesNoInputsTest.scala
NodesNoOutputsTest.scala
NodesTest.scala
metaModelIndependent 3 files
EdgesAttributesNamesNotEmptyTest.scala
ElementsIdUniqueTest.scala
NodesAttributesNamesNotEmptyTest.scala
nullChecks 3 file
NullChecksTests.scala
validatorDsl 1 file
DslTest.scala
UtilTest.scala
util 1 file
ProjectExportTest.scala
server.sbt
zeta-api 4 files ~/repo/zeta/api
wiki 2 files
exclude_targer.PNG
exclude_targer_result.PNG
build.sbt
Dockerfile
zeta-api-build 3 files ~/repo/zeta/api/project
build.properties
plugins.sbt
ZetaBuild.scala

Backend Dependencies

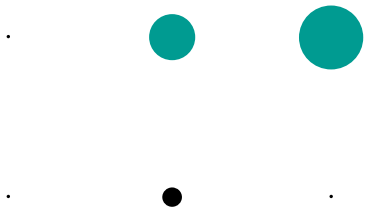
Imports

Bezeichnung	Master	Dev
scalaVersion	2.12.4	2.12.12
silhouetteVersion	5.0.3	6.1.1
playVersion	2.6.10	2.7.5
akkaVersion	2.5.8	2.6.10
javaFX/ScalaFX	8	11
scala-guice	4.1.1	4.2.11
scalatest	3.0.4	3.2.2
webjars-play	2.6.3	2.7.3
com.iheart.ficus	1.4.7	1.5.0
play-json	2.6.10	2.7.4
Mock-Test	org.scalamock	mockito-scala

Backend Dependencies

Imports

Bezeichnung	Master	Dev
reactivemongo	0.12.7	0.13.0
reactivemongo-play-json	0.12.7	0.13.0
scala-parser-combinators	1.0.6	1.1.2
org.rogach.scallop	3.1.1	3.5.1
scalariform	0.2.6	0.2.10



Backend Dependencies

Plugins

Bezeichnung	Master	Dev
sbt-plugin	2.6.10	2.7.5
sbt-wartremover	2.2.1	2.4.10
sbt-native-packager	1.3.2	1.7.4
sbt-updates	0.3.3	0.5.1
sbt-twirl	1.3.16	1.4.2
ch.epfl.scala.sbt-scalafix	-	0.9.21

CL-Tool

Bezeichnung	Master	Dev
sbt	1.2.8	1.3.13

Backend Dependencies

Ergebnis

Bezeichnung	Release	Dev
Scala	18.08.2017	13.07.2020
SBT	27.08.2017	27.07.2020
Playframework	21.12.2017	20.05.2020
Akka	07.12.2017	09.08.2020
Silhouette	17.12.2017	09.09.2019

Verbesserungen

- Docker-Compose angepasst
  - Backend Runtime läuft stabiler.
- GitHub Actions anstatt Travis CI
  - Travis Build Time $\leqq$ 610s.$_{maxT}$ (510s.$_{avg}$)
  - GitHub Actions $\leqq$ 441s.$_{maxT}$ (370s.$_{avg}$)
- Container Build (parallel)
  - Backend (sbt Image)
  - Frontend (NodeJs Image)
- Test (parallel)
  - Backend (ScalaTest)
  - Frontend (Jest)

Hochschule Konstanz
Fakultät Informatik

Granulare Beschreibung der Ergebnisse

Granulare Beschreibung der Ergebnisse

- Backend update von Java 8 auf Java 11
- Docker Setup für Backend
- Neues Docker Setup für das alte Frontend
- Fehlgeschlagene Backend Tests
- Fehler in Javascript/CSS bundling führte zu fehlerhaften SignIn, SignUp, Home und main Komponenten
- Alte Jest Frontend Unit tests sind nicht durchgelaufen
- Migration von YFiles auf eigenes Repo
- Eingeschränkte Freigabe des YFiles Repos um private Daten wie Lizenzen und Source Code zu verbergen
- Update des alten Stand der Github Wiki Seite, die fehlerhaften Code beinhaltete
- Veraltete Frontend Dependencies wurden auf einen neueren Stand migriert
- Webpack wurde von version 3 auf Webpack 5 migriert
- Sicherheitslücken durch alte Pakete wurden behoben

Granulare Beschreibung der Ergebnisse

- Auslagerung von statischer Umgebung in globale Environments, z.b. .env file
- Übergabe des Github Tokens an Docker um zur Laufzeit YFiles und YFiles Lizenzen aus privatem Repo zu klonen und zu integrieren
- YFiles deps wurden im frontend auf lokale Entwicklung umgestellt (nur Klonen des Repos ist notwendig und alles funktioniert)
- Docker build Abbruch bei Fehlen des User Tokens (mit Anleitung zur Integration des Github Tokens)
- SignUp löste ein Fehler im SMTP Protokoll aus und verhinderte das Anlegen eines Accounts
- Aktoren im Backend waren fehlerhaft und resultierten in einem Absturz des Backends
- Bundling von Bootstrap komponenten resultierte in Fehlern von Dropdown menüs und fehlenden Styles
- “Illegal Reflective Access Operations” durch veraltete Scala libs und Guice
- Migration auf neues Typesafe repo, da dies in der Zwischenzeit gemoved wurde

Granulare Beschreibung der Ergebnisse

- Backend Docker Images stürzten aufgrund von begrenztem Speicher von JVM und Docker regelmäßig ab -> auf 4gb erhöht
- Integration von dem Vue Projekt des alten Teams in das alte bestehende Projekt
- Bundling des Vue Projektes mittels Vue Loadern und Webpack (ursprünglich war der YFiles Editor ein separates und eigenständiges Vue Projekt)
- Toolbar des Yfiles Editors war initial offen anstatt geschlossen
- Implementierung von REST Schnittstellen für den neuen YFiles editor
- Fixen das nicht funktionsfähigen master standes auf einen lauffähigen state
- SignIn/SignUp Responses wurden durch JSON ersetzt
- Fehler durch CORS Config in der API
- Dockerisierung des neuen Frontends und Integration in ein docker-compose, zusammen mit mongodb und dem backend
- **Komplette Migration aller UI Komponenten in ein Vue-Frontend** (Dies beinhaltet Autorisierung, Routing, Umschreibung der Twirl konzepte in Vue template Direktiven, ersetzen von JQuery durch Data bindings und Aufteilung in unabhängige Komponenten
- Performance Steigerung durch eine Single Page Application (es werden nicht mehr für jede Funktionalität einzelne HTML Seiten ausgeliefert)

Granulare Beschreibung der Ergebnisse

- Umstellung des Login Response auf entsprechende 401 und 403 Status Codes, statt 303 + komplette html response der Login Seite
- Code Editoren sind nun wieder lauffähig und an die JSON Response vom Backend angepasst
- Einbettung von Yfiles in die Concept komponente
- Neuimplementierung der Import/Export Funktionalität
- Error / Info / Success Dialoge zur verbesserten User Experience
- Funktionsfähige Validierung und Validierungsgenerator Erzeugung im Frontend
- Aufteilung des Dashboard in mehrere Komponenten
- Implementierung eines Steppers zur verbesserten Übersicht und UX
- Verwendung von Github Actions für Builds
- YFiles Bug resultierte darin, dass Kantennamen nicht gesetzt werden konnten
- Anbindung des Concept Editors an das Backend
- YFiles wurde mit default JSON und nicht mit dem aktuellen Stand aus dem Backend initialisiert
- Das Erstellen und Löschen von Modell Instanzen funktioniert nicht
- Neue Frontend Jest Tests
- Neue Backend Unit Tests für spezifische Module
- Aufräumen von nicht verwendeten REST APIs und Komponente im Frontend, sowie branches, issues und twirl code der ins Frontend migriert wurde

Granulare Beschreibung der Ergebnisse

- Aktueller Standpunkt:
  - Implementierung der Yfiles Komponente als Model-Editor
- Was noch fehlt:
  - Fertigstellung des Model-Editors und Generierung des Codes

Hochschule Konstanz
Fakultät Informatik

Live Show

Habt ihr noch Fragen?