HAWAI-AST

im Auftrag der Firma  
  
Hochschule für Angewandte Wissenschaften Hamburg (HAW Hamburg)  
Berliner Tor 5  
20099 Hamburg  
Deutschland

Architektur

Fabian Pfaff

Version: 0.1

Status: In Arbeit

Stand: 04.05.2015

**Zusammenfassung**

Dieses Dokument beschreibt die Architektur des <Systemname>.

**Historie**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Status | Datum | Autor(en) | Erläuterung |
| 0.1 | In Arbeit | 04.05.2015 | Fabian Pfaff | Initiale Version für das SEP2 im 4. Semester |
|  |  |  |  |  |
|  |  |  |  |  |

**Inhaltsverzeichnis**

[1 Einleitung 4](#__RefHeading__944_1079646440)

[1.1 Ziele 4](#__RefHeading__946_1079646440)

[1.2 Konventionen 4](#__RefHeading__948_1079646440)

[2 Architektur des Gesamtsystems 5](#__RefHeading__950_1079646440)

[3 Komponenten 8](#__RefHeading__952_1079646440)

[3.1 Komponente CustomerManagement 8](#__RefHeading__954_1079646440)

[3.1.1 Verantwortungen der Komponente 8](#__RefHeading__956_1079646440)

[3.1.2 Außensicht 8](#__RefHeading__958_1079646440)

[3.1.3 Innensicht 9](#__RefHeading__960_1079646440)

[3.1.4 Entwurfsentscheidungen 9](#__RefHeading__962_1079646440)

[3.1.5 Schnittstellen zu Nachbarsystemen 9](#__RefHeading__964_1079646440)

[3.2 Komponente IBikeManagementComponent 10](#__RefHeading__966_1079646440)

[3.2.1 Verantwortungen der Komponente 10](#__RefHeading__968_1079646440)

[3.2.2 Außensicht 10](#__RefHeading__970_1079646440)

[3.2.3 Innensicht 11](#__RefHeading__972_1079646440)

[3.2.4 Entwurfsentscheidungen 11](#__RefHeading__974_1079646440)

[3.2.5 Schnittstellen zu Nachbarsystemen 11](#__RefHeading__976_1079646440)

[3.3 Komponente HTTPRestComponent 12](#__RefHeading__978_1079646440)

[3.3.1 Verantwortungen der Komponente 12](#__RefHeading__980_1079646440)

[4 Infrastruktur 12](#__RefHeading__982_1079646440)

[4.1 Persistenz 12](#__RefHeading__984_1079646440)

[4.2 Logging und Tracing 12](#__RefHeading__986_1079646440)

[4.3 Konfiguration 12](#__RefHeading__988_1079646440)

[5 Offene Punkte 12](#__RefHeading__990_1079646440)

[6 Literatur 12](#__RefHeading__992_1079646440)

# Einleitung

## Ziele

Ziel dieses Dokumentes ist die Beschreibung der Architektur des HAWAI-AST, sowie deren Infrastruktur und grundlegenden Konzepte.

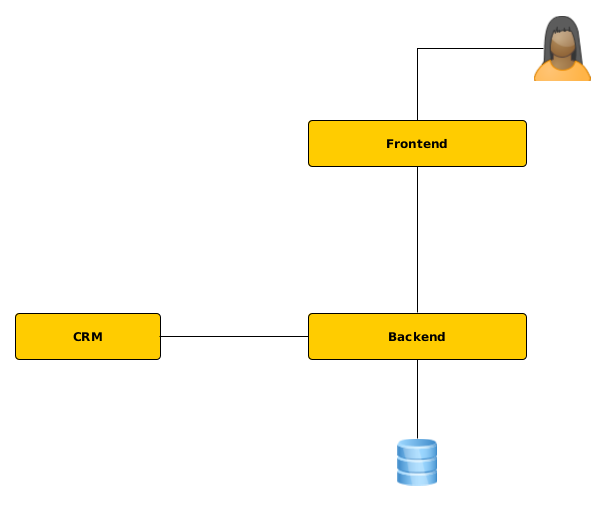
## Konventionen

Eigennamen werden in Anführungszeichen geschrieben:  
„Klassenname“

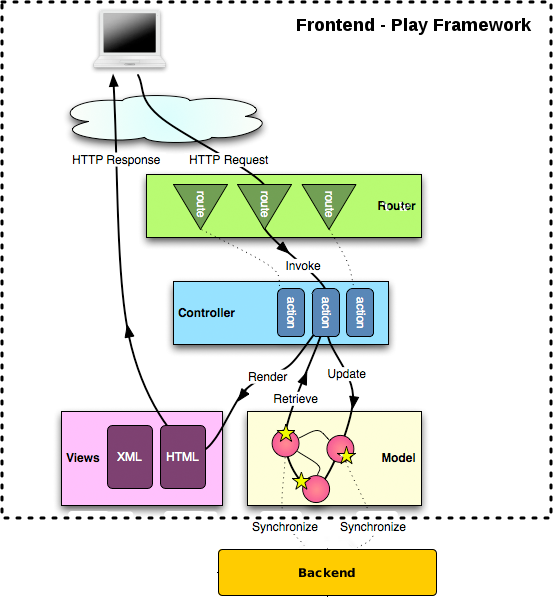
Sourcecode wird folgendermaßen dargestellt, Schlüsselwörter und Typen sind dabei farbig markiert: **int** createKunde(String name);

# Architektur des Gesamtsystems

In diesem Kapitel wird die Architektur des HAWAI-AST beschrieben.



**Frontend**

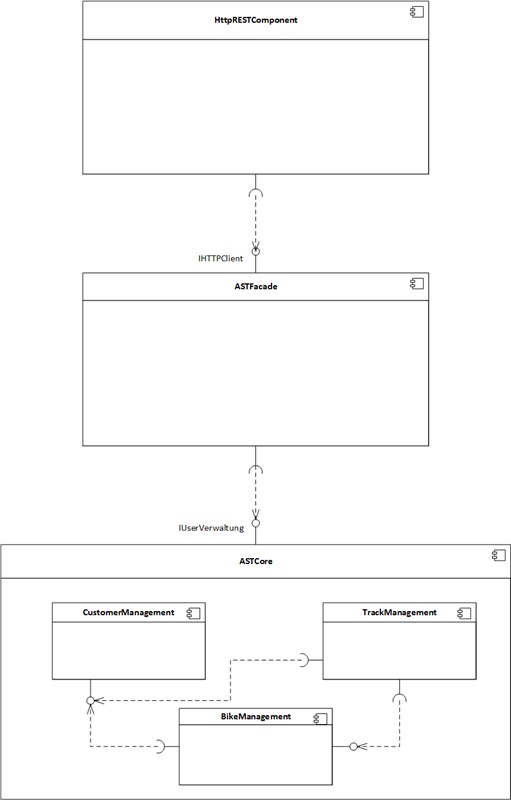


Der Endnutzer kann nur mit diesem Teil der Software kommunizieren.  
Das Frontend wird mir Hilfe des Play-Frameworks erstellt und dient dem Bereitstellen einer Weboberfläche als graphische Benutzerschnittstelle.  
Das Frontend kommuniziert über eine REST-Schnittstelle mit dem Backend.

**Backend**

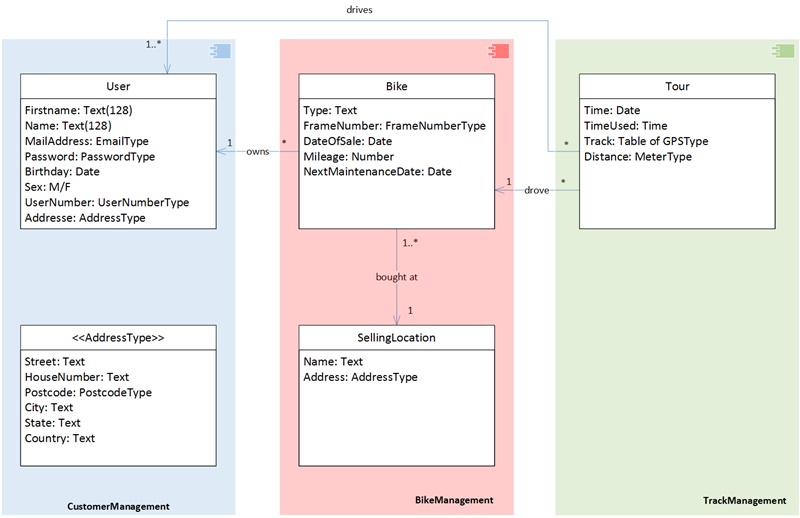
Das Backend dient als Zwischenschicht zur Persistenz, bestehend aus einer SQL-Datenbank und dem CRM „SuiteCRM“. Das Backend wird mit Hilfe des Spring-Frameworks umgesetzt.

Komponentenschnitt Backend:



Nur die „HTTPRestComponent“ kommuniziert mit der Außenwelt.  
Die Facade ist eine Zwischenschicht zwischen rein internen Komponenten („ASTCore“) und den Komponenten, die Schnittstellen nach außen anbieten.

**Datenmodel**

Im Datenmodel ist die Zugehörigkeit der einzelnen Datentypen zu ihrer Komponente im Backend durch farbliche Hinterlegung gekennzeichnet.

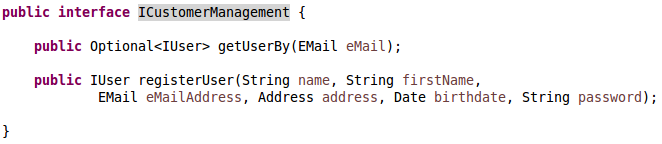
# Komponenten

## Komponente CustomerManagement

### Verantwortungen der Komponente

Die CustomerManegementComponent verwaltet den Zugriff auf die Benutzerkonten und deren Persistenz.

### Außensicht



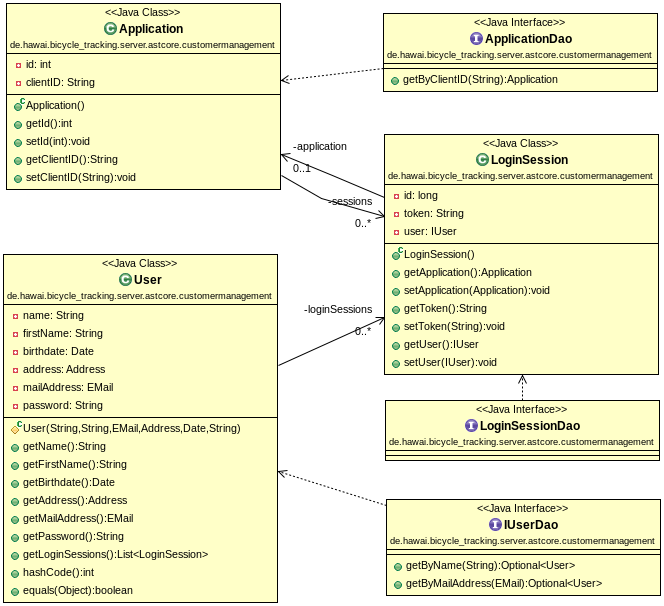
**public** Optional<IUser> getUserBy(EMail eMail):

Gibt den Benutzer mit der gegebenen Email zurück.

**public** IUser registerUser(String name, String firstName, EMail eMailAddress, Address address, Date birthdate, String password):

Erstellt einen neuen Benutzer mit den gegebenen Attributen und persistiert diesen.

### Innensicht



Die Entität „User“, die einen Benutzer der Software repräsentiert bekommt bei einem Login eine „LoginSession“ zugeordnet.

Eine „LoginSession“ wird einer „Application“ zugeordnet, um Sessions, die von verschiedenen Clients gestartet wurden zu unterscheiden.

### Entwurfsentscheidungen

### Schnittstellen zu Nachbarsystemen

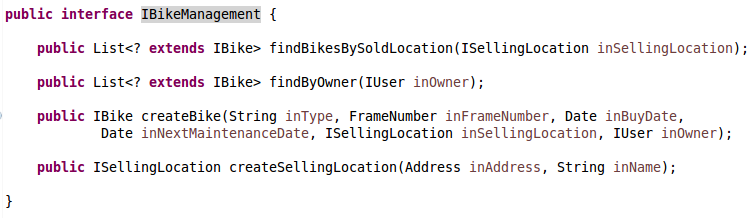
Keine.

## Komponente IBikeManagementComponent

### Verantwortungen der Komponente

Die „BikeManagement“-Komponente verwaltet die von Nutzern erstellten Fahrräder.

### Außensicht

**public** List<? **extends** IBike> findBikesBySoldLocation(ISellingLocation inSellingLocation):

Gibt alle an einem Verkaufsstandpunkt verkauften Fahrrad zurück.

**public** List<? **extends** IBike> findByOwner(IUser inOwner):

Gibt alle Fahrräder eines Nutzers zurück.

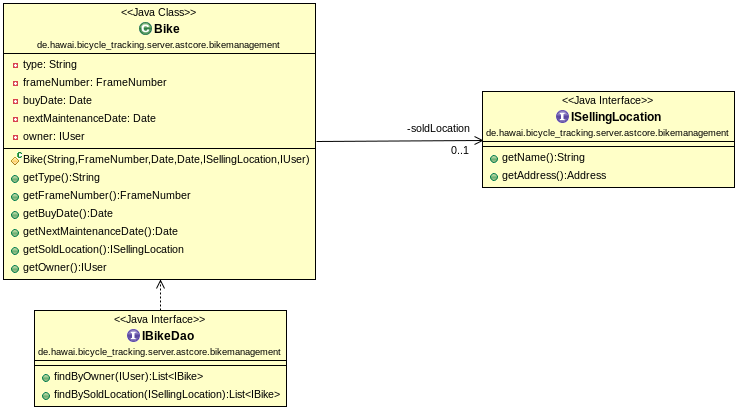
**public** IBike createBike(String inType, FrameNumber inFrameNumber, Date inBuyDate, Date inNextMaintenanceDate, ISellingLocation inSellingLocation, IUser inOwner):

Erstellt ein Fahrrad mit den gegebenen Attributen und persistiert dieses.

**public** ISellingLocation createSellingLocation(Address inAddress, String inName):

Erstellt einen Verkaufsstandort mit den gegebenen Attributen und persistiert diesen.

### Innensicht



Die Entität „Bike“ repräsentiert ein Fahrrad, das ein Kunde über die Weboberfläche angelegt hat.

Einem „Bike“ kann eine „SellingLocation“ (Verkaufsort) zugeordnet werden.

### Entwurfsentscheidungen

### Schnittstellen zu Nachbarsystemen

Keine.

## Komponente HTTPRestComponent

### Verantwortungen der Komponente

Die HTTPRest-Komponente stellt die Schnittstellen zu Nachbarsystemen bereit.  
Die Spezifikation der Schnittstelle findet sich im Anhang.

# Infrastruktur

Dieses Kapitel beschreibt die Infrastrukturkomponenten und -vorgaben des Projekts.

## Persistenz

Die Daten werden in einem externen CRM-System persistiert.  
Zur schnelleren Bearbeitung von Anfragen soll das Backend jedoch eine lokale Kopie auf einer Relationalen Datenbank vorhalten.

## Logging und Tracing

## Konfiguration

Das Frontend kommuniziert per JSON-Paketen mit dem Backend.  
Dabei identifiziert es sich ihm gegenüber durch den Headereintrag („Client-ID“; „DEV-101“).

# Offene Punkte

* Festlegung auf eine spezifische Datenbank
* Spezifikation der Schnittstelle zum CRM-System
* Caching-Konzept für Daten aus dem CRM im Backend

# Literatur