

# Kieler Kaffee Klub K<sup>3</sup> Project\*

Witzany, Jan

Luick, Bastian

`first1.last1@xxxxx.com`

`first2.last2@xxxxx.com`

Boottawong, Juti

`first2.last2@xxxxx.com`

2019

July

## **Abstract**

Dies ist eine kurze Zusammenfassung der Inhalte des in deutscher Sprache verfassten Dokuments.

---

\*No procrastination

## Contents

<b>1</b>	<b>System Requirements</b>	<b>4</b>
<b>2</b>	<b>Requirement</b>	<b>4</b>
2.1	User Requirements . . . . .	4
2.1.1	User Types . . . . .	4
2.1.2	User Stories . . . . .	6
<b>3</b>	<b>Mini-Stories</b>	<b>12</b>
<b>4</b>	<b>SiteLang Specification</b>	<b>13</b>
4.1	HERM Story Space . . . . .	18
<b>5</b>	<b>HERM-Schema</b>	<b>20</b>
5.1	HERM-Translation . . . . .	20
5.1.1	Description . . . . .	20
5.1.2	Entity . . . . .	21
5.1.3	Cluster . . . . .	22
5.1.4	Relationship . . . . .	22
5.1.5	Integerity Constraints . . . . .	23
5.1.6	Data Types . . . . .	24
5.2	Constraints Handling . . . . .	27
<b>6</b>	<b>Quantity Analysis</b>	<b>28</b>
<b>7</b>	<b>BPMN of Mini-Stories</b>	<b>29</b>
<b>8</b>	<b>Technology Stack</b>	<b>29</b>
<b>9</b>	<b>Framework</b>	<b>29</b>
<b>10</b>	<b>Obstacles</b>	<b>29</b>
<b>11</b>	<b>Outlook</b>	<b>29</b>

**WORKING TITLE::KAFFEESATT**

Bastian Luick(1018266), Jan Witzany(1011713),  
Juti Boottawong(1030476).

## Scope and Specifications of the Project

To provide miscellaneous information about coffee localities through a web application with students, inbound tourists and coffee fanatics in Kiel.

### VISION

Our Vision is that everyone know where they can find their suitable beverage place.

### MISSION

Provide a sophisticated web application for students, inbound tourists or coffee fanatics to discover a place to relax and enjoy their favorite coffee and supply themselves with coffee making utensil.

### W\*H

#### **Who will be using the system?**

Students, coffee fanatics and inbound tourists that are in Kiel.

#### **When will be the system be used?**

Breaks and Lunches.

#### **Where is the information system used?**

Desktop and Mobile at home, at work, on the go, in the city, near sights.

#### **What is represented in the system?**

Available coffee sorts, price-range, picture gallery, ratings from *Google* etc., direct links to places, misc. information about coffee (fair trade, preparation process, quality criteria, provenance etc..)

#### **How will the system be used?**

Desktop and Mobile via web browser.

#### **Why is the system used?**

To find the place to enjoy coffee or buy coffee accessories.

#### **What is the policy, intention, goal, and aim of the provider?**

To share our love and knowledge of coffee with coffee drinkers, coffee providers and coffee makers in Kiel.

## User & Scenarios Outline

**Students** much free time, high mobility in the area of Kiel (Student Ticket), bicycle routes, price sensitive  
Student thirst for coffee at the university during lectures.

**Inbound Tourist** no knowledge of Kiel, high price tolerance.

Tourist is in the middle of the city during a day trip and want to relax with a coffee.

**Coffee fanatics** want to know everything about the coffee or the coffee supplements high expectation, high demand of information,

At home and want to explore novel coffee localities.

# 1 System Requirements

## Speed

- Navigation < 3 second response time.
- Filtering shops and equipment categories < 5 second response time

## Product Environment

- The client must be connected with the internet during use of application
- Application works only on the following browsers Firefox, Chrome, Edge, Safari.

## Privacy Policy

- Delete permanetly on request user account and his reviews.
- Must allowed cookie to locate the user location.
- Password are encrypted.

## Localizability

- User interface components are in german
- Accept german specific language as input

# 2 Requirement

## 2.1 User Requirements

### 2.1.1 User Types

Possible properties of every user type: vegan, possibility to use own mug, reusable mug, can speak and read German, every user can paid by cash, age 16-35.

**Roles** {Content Provider(Admin, ContentManager), User (All user types)}

### **Admin**

Preferences: List of registered user and List of content

Behavior: Interact through desktop with web application for various task

Constraints:

Demands: Access to all content and user information possibility to delete user and add content manager.

Tasks: {CRUD of all content and user account}

### **Content Manager**

Preferences: Concrete and specific input options.(saved options)

Behavior: Want to upload a bulk of content and previews his inputs. Watch out for changes in the coffee shops.

Constraints: must have preview of create or edit content

Demands: Input pages for various content.

Tasks: {CRUD content}

### **Students**

Preferences: Cheap coffee, place with wlan, near bus station, buy with bitcoin

Behavior: User for orientation mobile devices and get to the locations mainly with bicycle or bus

Constraints: low funds, short on time.

Demands: Student wants to drink coffee and possibly a place to work.

Tasks: {filtering, search, look up, navigation, delete own profile, change own mail, rate shops}

### **Coffee fanatics**

Preferences: High quality coffee, parking lot, wlan, preferable possibility to see coffee making process, have a list of favorites

Behavior: User desktop and mobile devices to find misc. information about coffee shops in Kiel. Is content with paying more than average coffee price for high quality coffee.

Always looking for new shops and coffee beverages.

Constraints: No big companies or franchises.

Demands: Fanatics to experience novel coffee specialities in kiel and buy coffee making utensils.

### Tourists

Preferences: Nearby current location, card payment

Behaviour: Use mobile devices to find coffee shops in Kiel to relax and drink coffee. Is usually near sights.

Constraints: Low mobility, doesn't know localities, short on time, no big companies or franchises.

Demands: Local cafe shops that are nearby

### 2.1.2 User Stories

Table 1: User Story: User filtering options

User Story ID:	1		
User Story Name:	Search coffee place through filtering		
Created by:	KKK	Date created: June 13, 2019	
Roles	Students Coffee fanatic Tourists		
Description:	The User is on the website and use the presented filtering options to look up shops.		
Preconditions:	1. Know what filtering options mean.		
Postconditions:	Is presented list of shops		
Trigger:	Search button		
Flow:	1. Click on available filtering options 2. filtering results are showed 3. browse through list		

Table 2: User story detail

User Story ID:	2		
User Story Name:	User quick search		
Created by:	KKK	Date created: June 13, 2019	

Table 2 – Continued on next page

Table 2 – *Continued from previous page*

Roles	Student Tourist
Description:	User is on a break and are looking for a nearby coffee shop and use quick search function
Preconditions:	<ol style="list-style-type: none"> <li>1. Is on our landing page</li> <li>2. Click quick search button</li> </ol>
Postconditions:	Get a list of nearby coffee shops
Trigger:	Search button
Normal flow:	<ol style="list-style-type: none"> <li>1. the user clicked on the search button;</li> </ol>

Table 3: User Story Evaluation

User Story ID:	3		
User Story Name:	Evaluate coffee shop		
Created by:	KKK	Date created: June 13, 2019	
Roles	Student Coffee fanatic		
Description:	Evaluate Coffee shops and write a review		
Preconditions:	<ol style="list-style-type: none"> <li>1. The user is logged in.</li> </ol>		
Postconditions:	Can see his evaluation about the shop.		
Trigger:	Star symbol		
Normal flow:	<ol style="list-style-type: none"> <li>1. Click on a specific shop.</li> <li>2. Click on star symbol.</li> </ol>		

Table 4: User Story: Search equipment

User Story ID:	4
----------------	---

Table 4 – *Continued on next page*



Table 4 – *Continued from previous page*

User Story Name:	Search equipment		
Created by:	KKK	Date created: June 13, 2019	
Roles	Coffee fanatic		
Description:	Is on the equipment subpage and select filtering options.		
Preconditions:	1. Is on the equipment subpage		
Postconditions:	Show shops that fits the selected filtering options.		
Trigger:	Filtering options		
Flow:	1. Select filtering options		

Table 5: User story detail

User Story ID:	5		
User Story Name:	Edit review		
Created by:	KKK	Date created: June 13, 2019	
Roles	All		
Description:	User edit reviews		
Preconditions:	1. The user is logged in 2. He has given reviews		
Postconditions:	Review was edited		
Trigger:	Star symbol on the shop page		
Normal flow:	1. User clicked on star symbol.		

Table 6: User Story Manage Content

User Story ID:	6
User Story Name:	Manage Content

Table 6 – *Continued on next page*

Table 6 – *Continued from previous page*

Created by:	KKK	Date created: June 13, 2019	
Roles	Content-Manager Admin		
Description:	The Actor can add, edit or remove content {shop, equipment, informations, events} (do <b>CreateReadUpdateDelete</b> operations on content)		
Preconditions:	<ol style="list-style-type: none"> <li>1. The user is logged in.</li> <li>2. Is on the input page.</li> </ol>		
Postconditions:	Selected CRUD executed on database		
Trigger:	CRUD button		
Flow:	<ol style="list-style-type: none"> <li>1. Fill the input forms.</li> <li>2. Click either on save, delete or publish.</li> </ol>		

Table 7: User story detail

User Story ID:	7		
User Story Name:	Login		
Created by:	KKK	Date created: June 13, 2019	
Roles	Student Coffee fanatic Tourist Content-Manager Admin		
Description:	The user give in the account information and presses the login button.		
Preconditions:	<ol style="list-style-type: none"> <li>1. user is not logged in</li> <li>2. user is registered</li> <li>3. account data is correct</li> </ol>		

Table 7 – *Continued on next page*

Table 7 – *Continued from previous page*

Postconditions:	User is logged in the system and is redirect to account last page. And has access to the corresponding functionality.
Trigger:	Login button
Normal flow:	<ol style="list-style-type: none"> <li>1. Give account data</li> <li>2. click on login</li> </ol>

Table 8: User story detail

User Story ID:	8		
User Story Name:	Registration		
Created by:	KKK	Date created: June 13, 2019	
Roles	Student Tourist Coffee fanatic		
Description:	User register on the website.		
Preconditions:	<ol style="list-style-type: none"> <li>1. Is on the registration page.</li> <li>2. Fill out formula with correct data</li> </ol>		
Postconditions:	Is registered		
Trigger:	Register button		
Flow:	<ol style="list-style-type: none"> <li>1. Fill out input forms</li> <li>2. system checked input</li> <li>3. click on registration button</li> </ol>		

Table 9: User story detail

User Story ID:	9		
User Story Name:	Delete account		
Created by:	KKK	Date created: June 13, 2019	

Table 9 – *Continued on next page*

Table 9 – *Continued from previous page*

Roles	User Admin
Description:	User delete account
Preconditions:	<ol style="list-style-type: none"> <li>1. User has account</li> <li>2. User is logged in</li> </ol>
Postconditions:	Is automatic logged out of application and all reviews of the user are deleted.
Trigger:	Delete button
Flow:	<ol style="list-style-type: none"> <li>1. User clicked on delete button.</li> <li>2. Verify in popup his deletion request.</li> <li>3. Click delete button.</li> </ol>

Table 10: User story detail

User Story ID:	10		
User Story Name:	Manage registered user		
Created by:	KKK	Date created: June 13, 2019	
Roles	Admin		
Description:	Admin create content-manager and delete every other account		
Preconditions:	<ol style="list-style-type: none"> <li>1. User has account</li> <li>2. User is logged in</li> </ol>		
Postconditions:	DELETE operation on database on selected user account and his reviews.		
Trigger:	Button corresponding to the action		

Table 10 – *Continued on next page*

Table 10 – *Continued from previous page*

Flow:	<ol style="list-style-type: none"> <li>1. Select user account</li> <li>2. Click delete button</li> <li>3. Popup</li> <li>4. Select yes</li> </ol>
-------	---

Table 11: User Story: Logout

User Story ID:	11		
User Story Name:	Logout		
Created by:	KKK	Date created: June 13, 2019	
Roles	All		
Description:	The User is on the website and use the logout button		
Preconditions:	<ol style="list-style-type: none"> <li>1. User is logged in</li> </ol>		
Postconditions:	Is logout		
Trigger:	Logout button		
Flow:	<ol style="list-style-type: none"> <li>1. user clicked on the logout button.</li> </ol>		

### 3 Mini-Stories

#### **Search coffee shop through filtering(all)**

*Landing page, search page*

Preconditions: free access

Actions: select preferences to filter the shops

PostCond: shop list is updated corresponding to the selected options

#### **Evaluate coffee shop first time(registered user**

*Every page*

Preconditions: Free access

Actions: Skip to login, log as corresponding role

Postconditions: Is logged as user, stayed on shop site

*Content/shop/*

Preconditions: (Logged in)  
Action: (Evaluate shop)  
Postconditions: (Evaluation of user saved to user account and recalculate average rating)

#### **Manage Content (admin, content manager)**

*Every page*

PreCondition: Free access  
Actions: Skip to login, log as corresponding role  
PostCond: Is logged in, is on account management

#### *Account Management*

Preconditions: (logged in)  
Actions: (Skip to Content Management)  
Postconditions: (Is on content management)

#### *Content Management*

Preconditions: (Logged in)  
Actions: (CRUD action on content)  
Postcondition: (Corresponding crud action on content)

## **4 SiteLang Specification**

The following figures show various and distinct flow, structure and behaviour of the web information system from KAFFEESATT web application. Specifications: On every page there is the navigation bar. Furthermore it is possible to login or logout on every page as well. If user is not log in and want to use a log in feature he will be directed to the login input form.

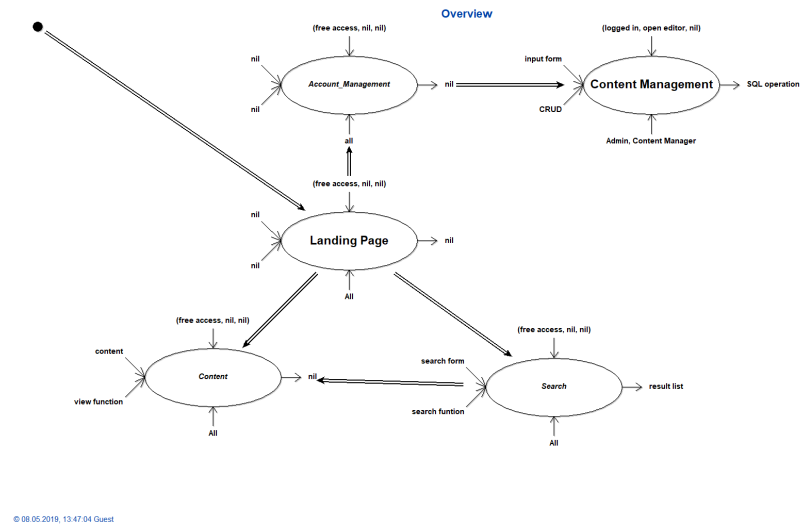


Figure 1: Overview of KAFFEESATT

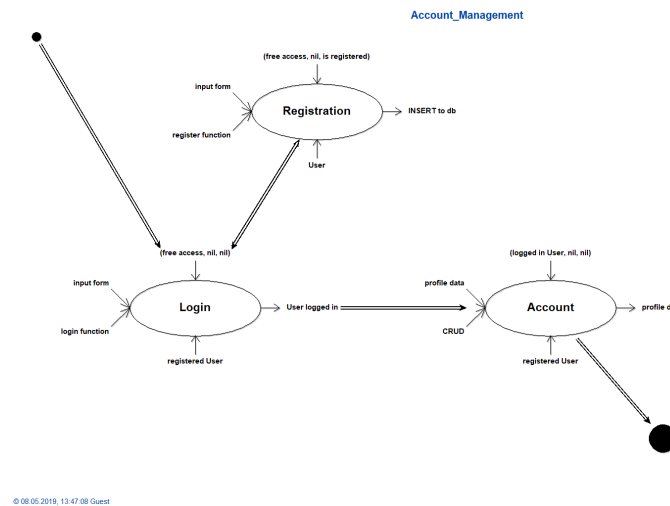
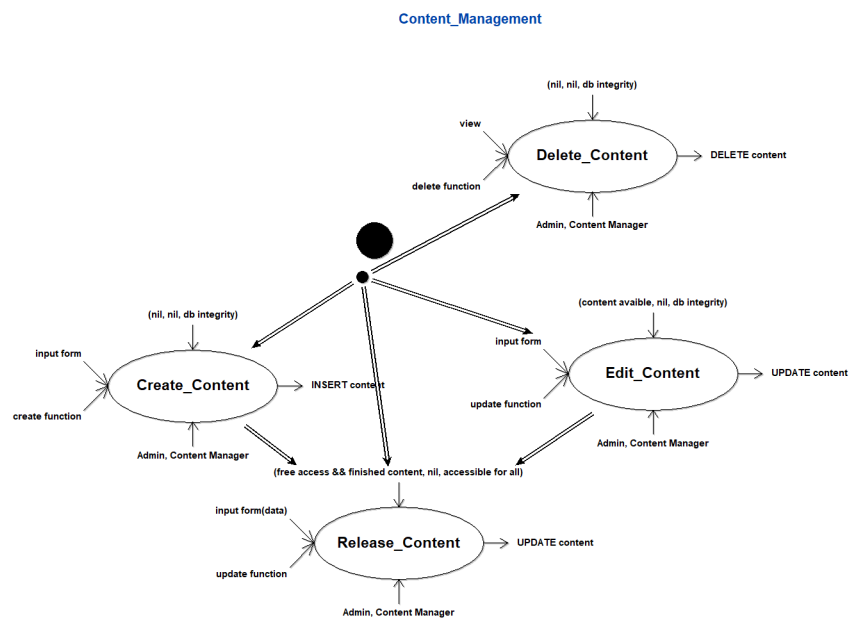


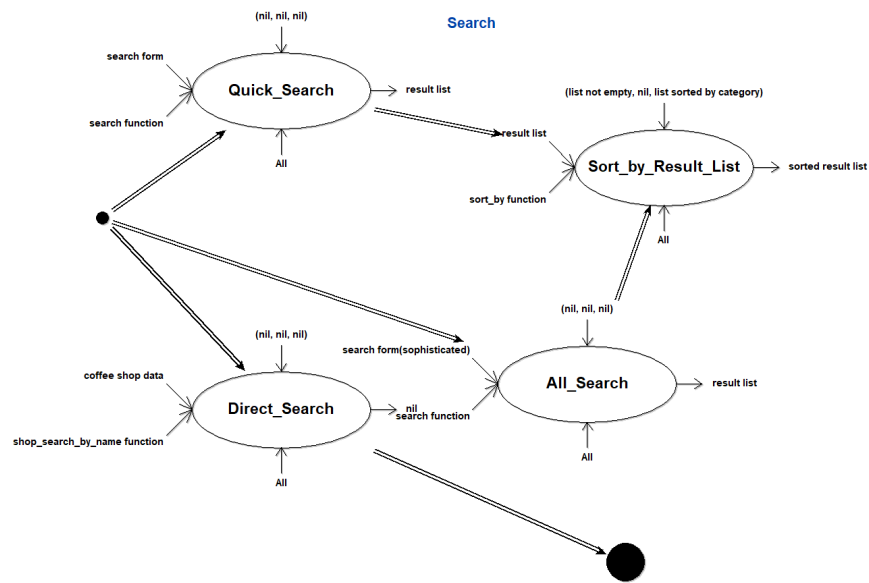
Figure 2: Account Management of KAFFEESATT



© 07.05.2019, 16:40:07 Guest

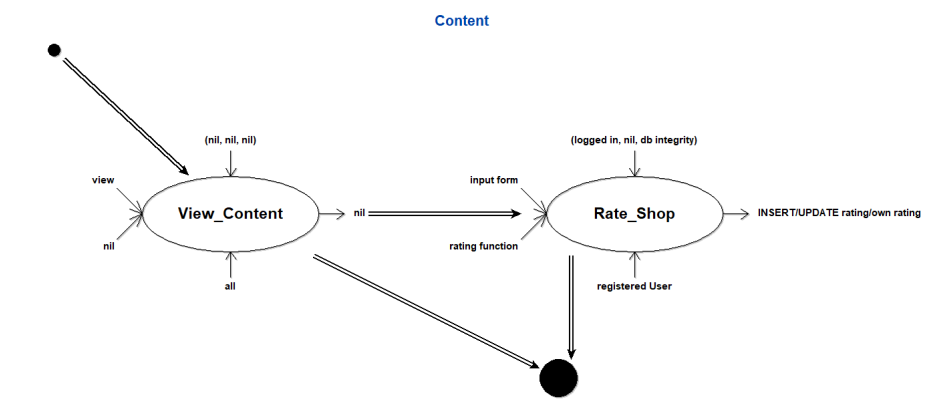
Figure 3: Content Management of KAFFEESATT





© 07.05.2019, 16.42.14 Guest

Figure 4: Search of KAFFEESATT



© 07.05.2019, 16:35:00 Guest

Figure 5: Content KAFFEESATT

## 4.1 HERM Story Space

### Defintions

#### SETs

- Content are the following entities  $C := \{\text{Shop}\}$  with their following attributes.
- Article are the following entities  $A := \{\text{Blend, Beans, Coffee\_Drink, Equipment}\}$ .
- User are the following entity  $U := \{\text{User and their specialization}\}$ .

#### FUNCTIONs

- $filter :: (C \times filterContent) \rightarrow Boolean : x \mapsto$  if Content satisfied filter flags: return true; else false;
- $filter :: (A \times filterArticle) \rightarrow Boolean : x \mapsto$  if Article satisfied filter flags: return true; else false;
- $filterContent :: C \rightarrow Value : \{C.Attributes\} = \{\text{poi, workstation, equipment, wlan, outdoor, fair\_trade, child\_friendly, disabled\_friendly, latte\_art, pet\_friendly, food, franchise, price\_class}\}$
- $filterArticle :: A \rightarrow Value : \{A.Attributes\} = \{\text{category, sub\_category}\}$
- $reduced(filterContent) :: \{quickserch(X) | X \in C.Attributes\} = \{\text{POI, Workstation, Rösterei}\}$
- $id : (C \cup A) \rightarrow id : x \rightarrow$  give the primary key of x
- $Result - List(X) : \text{List of members of Set X}$
- $Result(X) : \text{specific member of Set X}$

### Functionality by Scence

#### Overview

Scene (Content-Management)

View (in) Input-Form(C || A)

View (out) Execute corresponding SQL command

Scene (Search)

View (in) Input-Form(C)

View (out) Result-List(C)

Scene (Content-Management)

View (in) Input-Form(C)

View (out) INSERT/READ/UPDATE/DELETE(C)

Scene (Content)  
View (in) Content

### **ContentManagment**

Scene (Create\_Content)  
View (in) Input-Form(C || A)  
View (out) INSERT(C || A)

Scene (Release\_Content)  
View (in) Input-Form(C || A)  
View (out) UPDATE(C || A)

Scene (Edit\_Content)  
View (in) Input-Form(C || A)  
View (out) UPDATE(C || A)

Scene (Delete\_Content)  
View (in) View(C || A)  
View (out) Delete(C || A)

### **Content**

Scene (View\_Content)  
View (in) View(C || A)

Scene (Rate\_Shop)  
View (in) Input-Form(C.Rating)  
View (out) INSERT /UPDATE(C.Rating)

### **Search**

Page(LandingPage)  
Scene (Quick\_Search)  
View (in) Input-Form(reduced (filterContent))  
View (out) Result-List( $x|x \in C, filter(x) = true$ )

Page(Wiki, Coffee\_Shop)  
Scene (Direct\_Search)  
View (in) Input-Form(C.Name++C.Address || A.Name)  
View (out) Result(C) || Result(A)

Page(Coffee\_Shop,Wiki)  
Scene (Elaborate\_Search)  
View (in) Input-Form(filter)

View (out) Result-List( $x|x \in C||A, filter(x) = true$ )

Page(Coffee\_Shop,Wiki)

Scene (Sort\_by\_Result)

View (in) Result-List( $C || A$ )

View (out) Result-List(sort\_by( $C || A$ ))

## Account\_Management

Scene (Login)

View (in) Input-Form(U)

Scene (Account)

View (in) Input-Form(U)

View (out) READ/UPDATE(U)

Scene (Registration)

View (in) Input-Form(U)

View (out) INSERT(U)

## 5 HERM-Schema

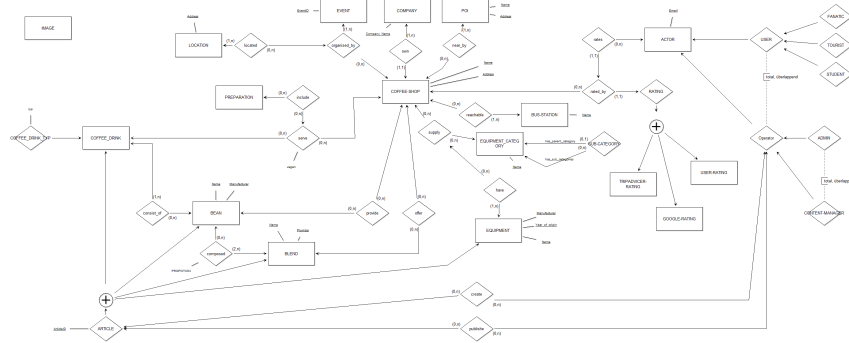


Figure 6: Simplify domain model

### 5.1 HERM-Translation

#### 5.1.1 Description

Higher-Order

Located was translate by taken the primary key of LOCATION aswell as the

primary keys from the relationship of organised\_by.

Rated\_By was translated by

Includes was translate by taken the primary key of PREPARATION aswell as the primary keys from the relationship of serves.

Sells was translated by taken the primary key of EQUIPMENT aswell as the primary keys from the relationship supplies.

Cluster

The ARTICLE cluster with the connection to the following entites: EQUIPMENT, COFFEE\_DRINK, BEAN and BLEND was transform with full participation key approach.

The RATING cluster with the connection to the following entities: GOOGLE-RATING, USER-RATING, TRIPADVISER-RATING was transform with the surrogate full participation key approach.

Complex data type

Address is a combination of the following attributes: StreetNr, StreetName, PostCode and Place.

### 5.1.2 Entity

(EQUIPMENT(Manufacturer, Year\_of\_origin, Name)  
(Manufacturer, Year\_of\_origin, Name))  
(EVENT(EventID, Time, Name, Access\_fee, Description)(EventID))  
(COFFEE-SHOP(Name, Address, Outdoor, Fair\_trade, Disabled\_friendly, Description, Wlan, Child\_friendly, Website, Fouding\_year, Pets\_friendly, Latte\_art, Seats, Workstation, Food, Price\_class, Franchise)(Name, Address))  
(BUS-STATION(Name, Line)(Name, Line))  
(COMPANY(Name)(Name))  
(BEAN(Name, Manufacturer, Provenance, Fair\_trade, Type)  
(Name, Manufacturer))  
(POI(Name, Address, Description)(Name, Address))  
(GOOGLE-RATING())()  
(USER-RATING())()  
(TRIPADVISER-RATING())()  
(BLEND(Name, Manufacturer, Provenance, Price\_range)(Name, Manufacturer))  
(LOCATION(Address, Description)(Address))  
(EQUIPMENT\_CATEGORY(Name)(Name))  
(ACTOR(Email, Actor\_Name, Password)(Email))  
(PREPARATION(Name, Description, Type)(Name))  
(COFFEE\_DRINK(Name, Description)(Name))  
(OPENING-TIME(Close, Open, Weekday)(Close, Open, Weekday))  
(USER(Email)(Email))  
(STUDENT(Email)(Email))  
(TOURIST(Email)(Email))

(FANATIC(Email)(Email))  
 (ADMIN(Email)(Email))  
 (CONTENT-MANAGER(Email)(Email))

### 5.1.3 Cluster

(RATING(RatingID,RATINGId)(RatingID, RATINGId))  
 (GOOGLE-RATING(RatingID, RATINGId)(RatingID, RATINGId))  
 (USER-RATING(RatingID, RATINGId)(RatingID, RATINGId))  
 (TRIPADVICER-RATING(RatingID, RATINGId)(RatingID, RATINGId))  
  
 (ARTICLE(ArticleID)(ArticleID)) (ARTICLEEQUIPMENT(ArticleID, Manufacturer, Year\_of\_origin, Name, Exposition)(ArticleID))  
 (ARTICLEBLEND(ArticleID, Name, Manufacturer, Exposition)(ArticleID))  
 (ARTICLEBEAN(ArticleID, Name, Manufacturer, Exposition)(ArticleID))  
 (ARTICLECOFFEE\_DRINK(ArticleID, Name, Exposition)(ArticleID))

### 5.1.4 Relationship

(consists\_of(Name, Manufacturer, Name)(Name, Manufacturer, Name))  
 (serves(Name, Address, Name, vegan)(Name, Address, Name))  
 (near\_by(Name, Address, Name, Address)(Name, Address, Name, Address))  
 (reachable(Name, Name, Address)(Name, Name, Address))  
 (owns(Name, Address, Name)(Name, Address))  
 (supplies(Name, Name, Address)(Name, Name, Address))  
 (provides(Name, Address, Name, Manufacturer)(Name, Address, Name, Manufacturer))  
 (composed(Name, Manufacturer, Name, Manufacturer, Propotion)(Name, Manufacturer, Name, Manufacturer))  
 (offers(Name, Manufacturer, Name, Address)(Name, Manufacturer, Name, Address))  
 (organised\_by(Name, Address, EventID)(Name, Address, EventID))  
 (OPERATOR(Email)(Email))  
 (SUB-CATEGORY(Name)(Name))  
 (COFFEE\_DRINK\_TYP(Name,Typ)(Name))  
 (belongs\_to(Manufacturer, Year\_of\_origin , Name ,Name)(Manufacturer, Year\_of\_origin, Name))  
 (Opens(Name, Address, Close, Open, Weekday)(Name, Address, Close, Open, Weekday))  
 (includes(Name, Address, Name, Name)(Name, Address, Name, Name))  
 (rated\_by(RatingID, RATINGId, Name, Address)(RatingID ,RATINGId))  
 (located(Address, Name, Address , EventID)(Address, Name, Address, EventID))  
 (sells(Manufacturer, Year\_of\_origin, Name, Name, Name, Address)(Manufacturer,

Year\_of\_origin, Name , Name , Name , Address))  
 (creates(Email , ArticleID)(Email , ArticleID))  
 (publishes(Email , ArticleID)(Email , ArticleID))  
 (rates(RatingID , RATINGId , Email)(RatingID , RATINGId))

### 5.1.5 Integrity Constraints

EVENT[EventID]  $\subseteq$  organised\_by[EventID]  
 BUS-STATION[Name]  $\subseteq$  reachable[Name]  
 COMPANY[Name]  $\subseteq$  owns[Name]  
 POI[Name, Address]  $\subseteq$  near\_by[Name, Address]  
 LOCATION[Address]  $\subseteq$  located[Address]  
 COFFEE\_DRINK[Name]  $\subseteq$  consists\_of[Name]  
 USER[Email]  $\subseteq$  ACTOR[Email]  
 consists\_of[Name, Manufacturer]  $\subseteq$  BEAN[Name, Manufacturer]  
 consists\_of[Name]  $\subseteq$  COFFEE\_DRINK[Name]  
 serves[Name, Address]  $\subseteq$  COFFEE-SHOP[Name, Address]  
 serves[Name]  $\subseteq$  COFFEE\_DRINK[Name]  
 near\_by[Name, Address]  $\subseteq$  COFFEE-SHOP[Name, Address]  
 near\_by[Name, Address]  $\subseteq$  POI[Name, Address]  
 reachable[Name, Line]  $\subseteq$  BUS-STATION[Name, Line]  
 reachable[Name, Address]  $\subseteq$  COFFEE-SHOP[Name, Address]  
 owns[Name, Address]  $\subseteq$  COFFEE-SHOP[Name, Address]  
 owns[Name]  $\subseteq$  COMPANY[Name]  
 supplies[Name]  $\subseteq$  EQUIPMENT\_CATEGORY[Name]  
 supplies[Name, Address]  $\subseteq$  COFFEE-SHOP[Name, Address]  
 provides[Name, Address]  $\subseteq$  COFFEE-SHOP[Name, Address]  
 provides[Name, Manufacturer]  $\subseteq$  BEAN[Name, Manufacturer]  
 composed[Name, Manufacturer]  $\subseteq$  BEAN[Name, Manufacturer]  
 composed[Name, Manufacturer]  $\subseteq$  BLEND[Name, Manufacturer]  
 offers[Name, Manufacturer]  $\subseteq$  BLEND[Name, Manufacturer]  
 offers[Name, Address]  $\subseteq$  COFFEE-SHOP[Name, Address]  
 organised\_by[Name, Address]  $\subseteq$  COFFEE-SHOP[Name, Address]  
 organised\_by[EventID]  $\subseteq$  EVENT[EventID]  
 OPERATOR[Email]  $\subseteq$  ACTOR[Email]  
 SUB-CATEGORY[Name]  $\subseteq$  EQUIPMENT\_CATEGORY[Name]  
 SUB-CATEGORY[Name]  $\subseteq$  EQUIPMENT\_CATEGORY[Name]  
 COFFEE\_DRINK\_TYP[Name]  $\subseteq$  COFFEE\_DRINK[Name]  
 belongs\_to[Name]  $\subseteq$  EQUIPMENT\_CATEGORY[Name]  
 belongs\_to[Manufacturer, Year\_of\_origin, Name]  $\subseteq$  EQUIPMENT[Manufacturer, Year\_of\_origin, Name]  
 Opens[Name, Address]  $\subseteq$  COFFEE-SHOP[Name, Address]  
 Opens[Close, Open, Weekday]  $\subseteq$  Opening-Time[Close, Open, Weekday]  
 includes[Name, Address, Name]  $\subseteq$  serves[Name, Address, Name]  
 includes[Name]  $\subseteq$  PREPARATION[Name]  
 rated\_by[Name, Address]  $\subseteq$  COFFEE-SHOP[Name, Address]



rated\_by[RatingID, RATINGId]⊆RATING[RatingID, RATINGId]  
 located[Address]⊆LOCATION[Address]  
 located[Name, Address, EventID]⊆organised\_by[Name, Address, EventID]  
 sells[Manufacturer, Year\_of\_origin, Name]⊆belongs\_to[Manufacturer, Year\_of\_origin, Name]  
 sells[Name, Name, Address]⊆supplies[Name, Name, Address]  
 STUDENT[Email]⊆USER[Email]  
 TOURIST[Email]⊆USER[Email]  
 FANATIC[Email]⊆USER[Email]  
 ADMIN[Email]⊆OPERATOR[Email]  
 CONTENT-MANAGER[Email]⊆OPERATOR[Email]  
 creates[Email]⊆OPERATOR[Email]  
 creates[ArticleID]⊆ARTICLEEQUIPMENT[ArticleID]  
 creates[ArticleID]⊆ARTICLEBLEND[ArticleID]  
 creates[ArticleID]⊆ARTICLEBEAN[ArticleID]  
 creates[ArticleID]⊆ARTICLECOFFEE\_DRINK[ArticleID]  
 publishes[Email]⊆OPERATOR[Email]  
 publishes[ArticleID]⊆ARTICLEEQUIPMENT[ArticleID]  
 publishes[ArticleID]⊆ARTICLEBLEND[ArticleID]  
 publishes[ArticleID]⊆ARTICLEBEAN[ArticleID]  
 publishes[ArticleID]⊆ARTICLECOFFEE\_DRINK[ArticleID]  
 rates[RatingID, RATINGId]⊆rated\_by[RatingID, RATINGId]  
 rates[Email]⊆ACTOR[Email]  
 ARTICLEEQUIPMENT[ArticleID] || ARTICLEBLEND[ArticleID] || ARTICLEBEAN[ArticleID] ||  
 ARTICLECOFFEE\_DRINK[ArticleID]

### 5.1.6 Data Types

EQUIPMENT.Manufacturer::VARCHAR(n)  
 EQUIPMENT.Year\_of\_origin::VARCHAR(n)  
 EQUIPMENT.Name::VARCHAR(n)  
 EVENT.EventID::INTEGER EVENT.Time::INTEGER EVENT.Name::VARCHAR(n)  
 EVENT.Access\_fee::INTEGER EVENT.Description::VARCHAR(n)  
 COFFEE-SHOP.Name::VARCHAR(n)  
 COFFEE-SHOP.Address::VARCHAR(n)  
 COFFEE-SHOP.Outdoor::BOOLEAN COFFEE-SHOP.Fair\_trade::BOOLEAN  
 COFFEE-SHOP.Disabled\_friendly::BOOLEAN COFFEE-SHOP.Description::VARCHAR(n)  
 COFFEE-SHOP.Wlan::BOOLEAN COFFEE-SHOP.Child\_friendly::BOOLEAN  
 COFFEE-SHOP.Website::VARCHAR(n)  
 COFFEE-SHOP.Fouding\_year::INTEGER COFFEE-SHOP.Pets\_friendly::BOOLEAN  
 COFFEE-SHOP.Latte\_art::VARCHAR(n)  
 COFFEE-SHOP.Seats::VARCHAR(n)  
 COFFEE-SHOP.Workstation::BOOLEAN COFFEE-SHOP.Food::VARCHAR(n)  
 COFFEE-SHOP.Price\_class::VARCHAR(n)

COFFEE-SHOP.Franchise::BOOLEAN BUS-STATION.Name::VARCHAR(n)  
 BUS-STATION.Line::VARCHAR(n)  
 COMPANY.Name::VARCHAR(n)  
 BEAN.Name::VARCHAR(n)  
 BEAN.Manufacturer::VARCHAR(n)  
 BEAN.Provenance::VARCHAR(n)  
 BEAN.Fair\_trade::BOOLEAN BEAN.Type::VARCHAR(n)  
 POI.Name::VARCHAR(n)  
 POI.Address::VARCHAR(n)  
 POI.Description::CHARACTER(n)  
 BLEND.Name::VARCHAR(n)  
 BLEND.Manufacturer::VARCHAR(n)  
 BLEND.Provenance::VARCHAR(n)  
 BLEND.Price\_range::INTEGER LOCATION.Address::VARCHAR(n)  
 LOCATION.Description::VARCHAR(n)  
 EQUIPMENT\_CATEGORY.Name::VARCHAR(n)  
 ACTOR.Email::VARCHAR(n)  
 ACTOR.Actor\_Name::VARCHAR(n)  
 ACTOR.Password::VARCHAR(n)  
 PREPARATION.Description::VARCHAR(n)  
 PREPARATION.Type::VARCHAR(n)  
 PREPARATION.Name::VARCHAR(n)  
 COFFEE\_DRINK.Name::VARCHAR(n)  
 COFFEE\_DRINK.Description::VARCHAR(n)  
 OPENING-TIME.Close::INTEGER OPENING-TIME.Open::INTEGER OPENING-TIME.Weekday::VARCHAR(n)  
 USER.Email::VARCHAR(n)  
 RATING.RatingID::INTEGER RATING.RATINGId::INTEGER consists\_of.Name::VARCHAR(n)  
 consists\_of.Manufacturer::VARCHAR(n)  
 consists\_of.Name::VARCHAR(n)  
 serves.vegan::BOOLEAN serves.Name::VARCHAR(n)  
 serves.Address::VARCHAR(n)  
 serves.Name::VARCHAR(n)  
 near\_by.Name::VARCHAR(n)  
 near\_by.Address::VARCHAR(n)  
 near\_by.Name::VARCHAR(n)  
 near\_by.Address::VARCHAR(n)  
 reachable.Name::VARCHAR(n)  
 reachable.Name::VARCHAR(n)  
 reachable.Address::VARCHAR(n)  
 owns.Name::VARCHAR(n)  
 owns.Address::VARCHAR(n)  
 owns.Name::VARCHAR(n)  
 supplies.Name::VARCHAR(n)  
 supplies.Name::VARCHAR(n)  
 supplies.Address::VARCHAR(n)

provides.Name::VARCHAR(n)  
 provides.Address::VARCHAR(n)  
 provides.Name::VARCHAR(n)  
 provides.Manufacturer::VARCHAR(n)  
 composed.Propotion::VARCHAR(n)  
 composed.Name::VARCHAR(n)  
 composed.Manufacturer::VARCHAR(n)  
 composed.Name::VARCHAR(n)  
 composed.Manufacturer::VARCHAR(n)  
 offers.Name::VARCHAR(n)  
 offers.Manufacturer::VARCHAR(n)  
 offers.Name::VARCHAR(n)  
 offers.Address::VARCHAR(n)  
 organised\_by.Name::VARCHAR(n)  
 organised\_by.Address::VARCHAR(n)  
 organised\_by.EventID::INTEGER OPERATOR.Email::VARCHAR(n)  
 SUB-CATEGORY.Name::CHAR COFFEE\_DRINK\_TYP.Type::VARCHAR(n)  
 COFFEE\_DRINK\_TYP.Name::VARCHAR(n)  
 belongs\_to.Manufacturer::VARCHAR(n)  
 belongs\_to.Year\_of\_origin::VARCHAR(n)  
 belongs\_to.Name::VARCHAR(n)  
 belongs\_to.Name::VARCHAR(n)  
 Opens.Name::VARCHAR(n)  
 Opens.Address::VARCHAR(n)  
 Opens.Close::INTEGER Opens.Open::INTEGER Opens.Weekday::VARCHAR(n)  
 RATINGGOOGLE-RATING.RatingID::INTEGER RATINGGOOGLE-RATING.RATINGId::INTEGER  
 RATINGUSER-RATING.RatingID::INTEGER RATINGUSER-RATING.RATINGId::INTEGER  
 RATINGTRIPADVICER-RATING.RatingID::INTEGER RATINGTRIPADVICER-  
 RATING.RATINGId::INTEGER ARTICLEEQUIPMENT.ArticleID::INTEGER  
 ARTICLEEQUIPMENT.Manufacturer::VARCHAR(n)  
 ARTICLEEQUIPMENT.Year\_of\_origin::VARCHAR(n)  
 ARTICLEEQUIPMENT.Name::VARCHAR(n)  
 ARTICLEEQUIPMENT.Exposition::CHARACTER(n)  
 ARTICLEBLEND.ArticleID::INTEGER ARTICLEBLEND.Name::VARCHAR(n)  
 ARTICLEBLEND.Manufacturer::VARCHAR(n)  
 ARTICLEBLEND.Exposition::CHARACTER(n)  
 ARTICLEBEAN.ArticleID::INTEGER ARTICLEBEAN.Name::VARCHAR(n)  
 ARTICLEBEAN.Manufacturer::VARCHAR(n)  
 ARTICLEBEAN.Exposition::CHARACTER(n)  
 ARTICLECOFFEE\_DRINK.ArticleID::INTEGER ARTICLECOFFEE\_DRINK.Name::VARCHAR(n)  
 ARTICLECOFFEE\_DRINK.Exposition::CHARACTER(n)  
 includes.Name::VARCHAR(n)  
 includes.Address::VARCHAR(n)  
 includes.Name::VARCHAR(n)  
 includes.Name::VARCHAR(n)  
 rated\_by.RatingID::INTEGER rated\_by.RATINGId::INTEGER rated\_by.Name::VARCHAR(n)

rated\_by.Address::VARCHAR(n)  
 located.Address::VARCHAR(n)  
 located.Name::VARCHAR(n)  
 located.Address::VARCHAR(n)  
 located.EventID::INTEGER sells.Manufacturer::VARCHAR(n)  
 sells.Year\_of\_origin::VARCHAR(n)  
 sells.Name::VARCHAR(n)  
 sells.Name::VARCHAR(n)  
 sells.Name::VARCHAR(n)  
 sells.Address::VARCHAR(n)  
 STUDENT.Email::VARCHAR(n)  
 TOURIST.Email::VARCHAR(n)  
 FANATIC.Email::VARCHAR(n)  
 ADMIN.Email::VARCHAR(n)  
 CONTENT-MANAGER.Email::VARCHAR(n)  
 creates.Email::VARCHAR(n)  
 creates.ArticleID::INTEGER publishes.Email::VARCHAR(n)  
 publishes.ArticleID::INTEGER rates.RatingID::INTEGER rates.RATINGId::INTEGER  
 rates.Email::VARCHAR(n)

## 5.2 Constraints Handling

Referential constraints are enforced through the database management system by adding constraint to the tables which have the corresponding references. The majority of the referential constraints are foreign keys.

Exclude constraints are enforced by using triggers to ensure that insert are only possible if there is no overlap when inserting the data.

## 6 Quantity Analysis

### ENTITY

- SHOP: 50
- USER: 300
- COMPANY: 30
- COFFEE\_DRINK: 100
- SHOP: 50
- BLEND: 100
- BEANS: 50
- EVENT: 300
- BUSSTATION:60
- POI:15

**RELATIONSHIP** A CoffeeShop belongs to exactly **1** Company. Also it has **7** OpeningTimes for each day of the week. We assume an average amount of **3** BusStations and **2** POIs for each CoffeeShop. It serves about **30** beverages and offers **10** Beans, **20** Blends and **4** EquipmentCategories. For future proof we provide a sells Table where Shops can offer specific Equipment models, but we do not consider it yet.

A CoffeeShop can organize several Events, we calculate with an average about **5** Events per year. We do not delete expired Events yet, this might be implemented in later work.

### CLUSTER

- ARTICLE: 100
- ARTICLE\_BEAN: 100
- ARTICLE\_DRINK: 100
- ARTICLE\_BLEND: 100
- GOOGLE-RATING:50
- TRIPADVISOR-RATING:50
- USER-RATING:400

**Function Calls** Frequency of usage of functionality differentiated between content-manager and regular user:

**Content-Manger**

- Create: 300 per year
- Update: 1000 per year
- Delete: 50 per year

**User**

- Search: 400 per day
- Filter: 300 per day

## **7 BPMN of Mini-Stories**

## **8 Technology Stack**

## **9 Framework**

## **10 Obstacles**

## **11 Outlook**