University for Applied Sciences

Informatics Department

Applied Informatics

To be defined

Documentation for the Architecture of an Mobile Application for Preventing Food Waste

Bruno Macedo da Silva
676839
inf3645@hs-worms.de

Supervisor	Prof.Dr. Volker Schwarzer
Working Period:	Summer Semester 2022
Due Date:	31.Juni 2022

Inhaltsverzeichnis

1 Introduction and Goals **5**

1.1 Design Purpose 5

1.2 Primary Functionality 5

1.3 Quality Attributes 7

1.4 Constraints 10

2 4+1 Architectural View Model **11**

2.1 Behaviour view 12

2.2 Structural view 13

Literaturverzeichnis **14**

1 Introduction and Goals

According to the Food and Agriculture Organization of the United Nations (FAO) in 2019 931 millions tonnes of food were wasted [FAO, 2013]. This has environmental, but special social consequences. In a world were approximately 9.9% of the [AAH, 2022] population suffers from hunger that waste percentage sounds paradoxal.

According to United Nations (UN) 5% of the globally food loss and waste comes from restaurants [UN, 2022]. The solution for this problem muss be locally applied so its effects can be seen in a global structure. To do so we propose to develop a mobile application that connects restaurants, bakeries and or pastries to clients. The former would offer their remaining products, which are still consumable, prior to the closing time, to a small price and the latter would browser in the app to find which shops are offering products.

1.1 Design Purpose

The main purpose of this architecture is creating exploratory prototype of an App. We aim to test it with potential Stakeholder and regions to analyze the general their acceptance and wishes [Cervantes and Kazman, 2016] and get a fast feedback.

This prototype will also make it feasible to identify unknown needs an wishes of the potential Stakeholder, so we can eventually increase the scope of functionality. Exploring this domain will also provide us with information regarding the behavior of our Stakeholder when it comes to buying and serving food that would be wasted, but is still consumable.

1.2 Primary Functionality

From the following use cases we will be able to define the primary functionality of our application and furthermore identify its main quality attributes

Use Case	Description
UC-1: Register as Client	The = Client register an e-mail address.
UC-2: Login	The Client logins in to the system.
UC-3: Place an order	The Client chooses a Provider.
UC-4: Register payment	The Client register a payment method.
UC-5: Register as Provider	The Provider register their facility and products.
UC-6: Update availability	The Provider upload their availability to provide a product.

Those use cases are also represented in the following use case diagram:

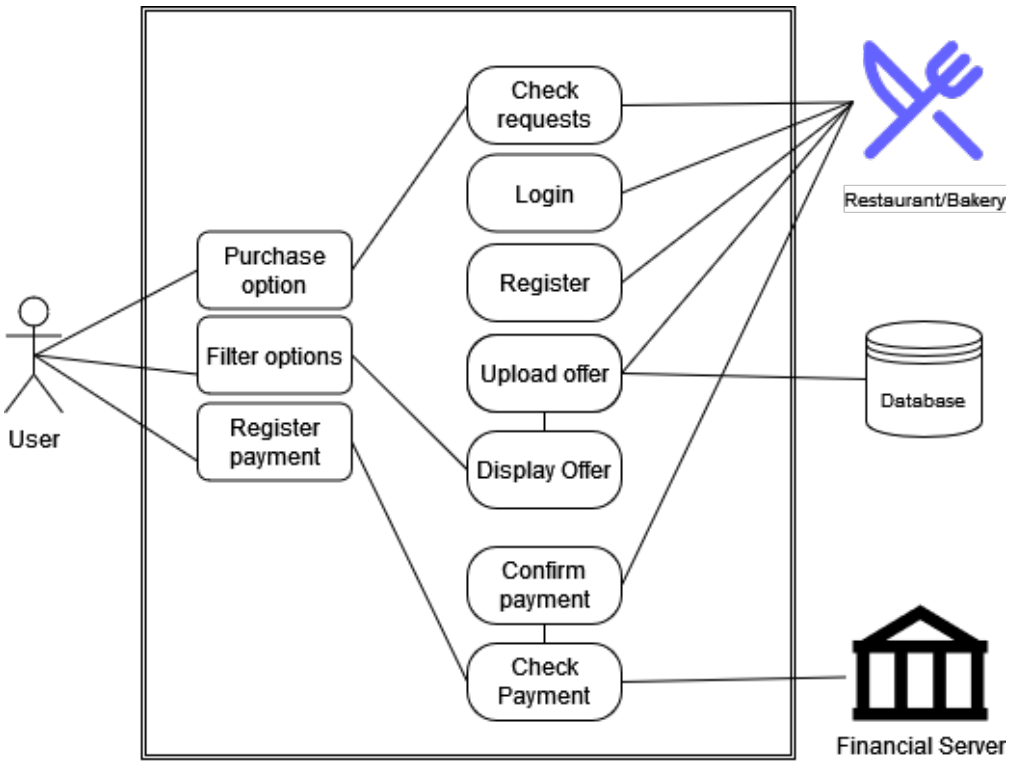

Abbildung 1: Preliminary functions

1.3 Quality Attributes

With the given use cases we will then be able to define the major quality attributes that are involved in the development of this application. We want those qualities to be measurable and testable so we can verify if the system meets the needs our stakeholders [Cervantes and Kazman, 2016].

ID	Quality Attribute	Scenario	Associated Use Case
QA-1	Performance	A Client register their e-mail address and he can immediate browse in the app.	UC-1
QA-2	Performance	A Client opens the app and he can immediate browse in the app.	UC-2
QA-3	Performance	A Client choose a Provider and place his order. After the confirmation of payment, a push-message is displayed in the app confirming the purchase.	UC-3
QA-4	*[to be defined]*	A Client register his credit card or select another payment method and the confirmation as soon as he confirmed with his Provider.	UC-4
QA-5	Usability	A Provider is able to register his company, specify the kind of products he offers and upload a logo or picture of his shop.	UC-05
QA-6	Usability	A Provider is able to update in the app if he is offering for that day any product.	UC-6

The defined quality attributes are represented in the following scenarios:

Performance	
Scenario	Value
Source	Client
Stimulus	wishes to create an account
Artifact	platform
Environment	runtime
Response	immediate access to the app
Response Measure	time between confirmation and access
Source	Client
Stimulus	wants to search fo restaurants or bakeries
Artifact	platform
Environment	peak period, between 6 and 7 pm on Friday
Response	immediate access to the offers
Response Measure	how quick does the Client get an updated regarding availability of products
Source	Client
Stimulus	place an order
Artifact	platform
Environment	peak period, between 6 and 7 pm on Friday
Response	confirmation of the purchase after the payment
Response Measure	time between confirmation of the payment and confirmation of the order

Usability	
Scenario	Value
Source	Provider
Stimulus	wants to offer his remaining products in the app
Artifact	platform
Environment	working time, during afternoon
Response	offer available in the app
Response Measure	How long did the registration and upload process took? Were all necessary information available in the app or did the Provider need to search it outside the app? How long did the registration process took?
Source	Registered Provider
Stimulus	wants wants to make a last minute offer
Artifact	platform
Environment	peak period, between 6 and 7 pm on Friday
Response	immediate availability of the offer in the app
Response Measure	how long did it take for the Provider to upload the offer? Was it easy to input all necessary information like, quantity, location and take-away time? Can he do it without any burden?

1.4 Constraints

In general, we can say that constrains are burdens to the development of the project. They define a set of non-negotiable rules that must be exist [Franzen and Thoms, 2020].

In this project we must distinguish between Technical and Business Constraints. The former describes specific elements of the project, like programming language, released platform (i.e. operational systems) and technical decisions related to the functionalities. The latter deals with management elements[Franzen and Thoms, 2020] such time, budget and team.

ID	Constraint	Category	Description
CT-1	Programming Language	Technical	Java, Kotlin, iOS, Swift
CT-2	Platform	Technical	Android, IoS
CT-3	Payment	Technical	Creating own framework or integrating with existing one (Google Pay, Apple Pay, PayPall)
CT-4	Login	Technical	Using or not federation or creating own login system
CT-5	Time to first prototype release	Business	How long until a first prototype that can be tested wir real users
CT-6	Testing time	Business	Time window to test general acceptance
CT-7	Budget	Business	To maintain a team during the testing phase
CT-8	Team	Business	To analyze the main usage of the app for further development

2 4+1 Architectural View Model

In this section we will describe the App using the 4+1 Architectural View Model. With this model we will represent the App using five different views, which should focus on specific elements of the project. Each view provide a different purpose [Kruchten, 1995]. For this project we will provide the 3 following views of the 4+1 Architectural View Model:

- **Scenario view**: simple description for the end user
- **Behaviour view**: description of the existing processes
- **Structural view**: object-oriented decomposition

The scenario view was presented in the section 1.2 of this project.

2.1 Behaviour view

The following Activity Diagram depicts the register and login procedure within the app.

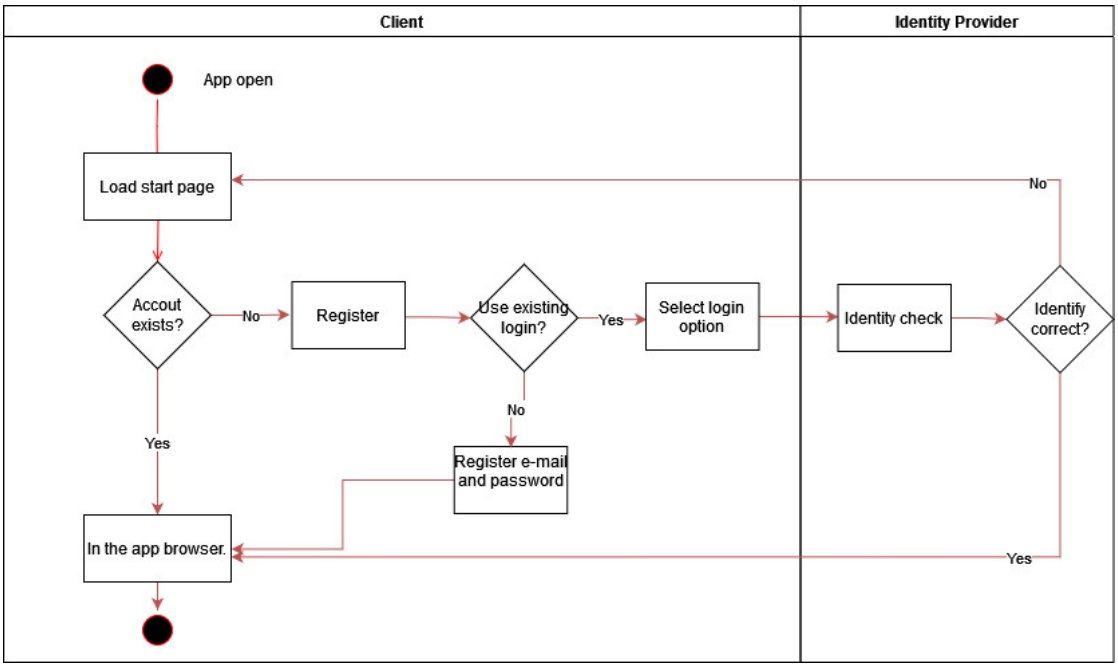

Abbildung 2: Login procedures

2.2 Structural view

To describe this view we choose a Class Diagram. With it we may provide a static description of elements within the structure of our system. They can also be used during the programming process to display what is needed to be done.

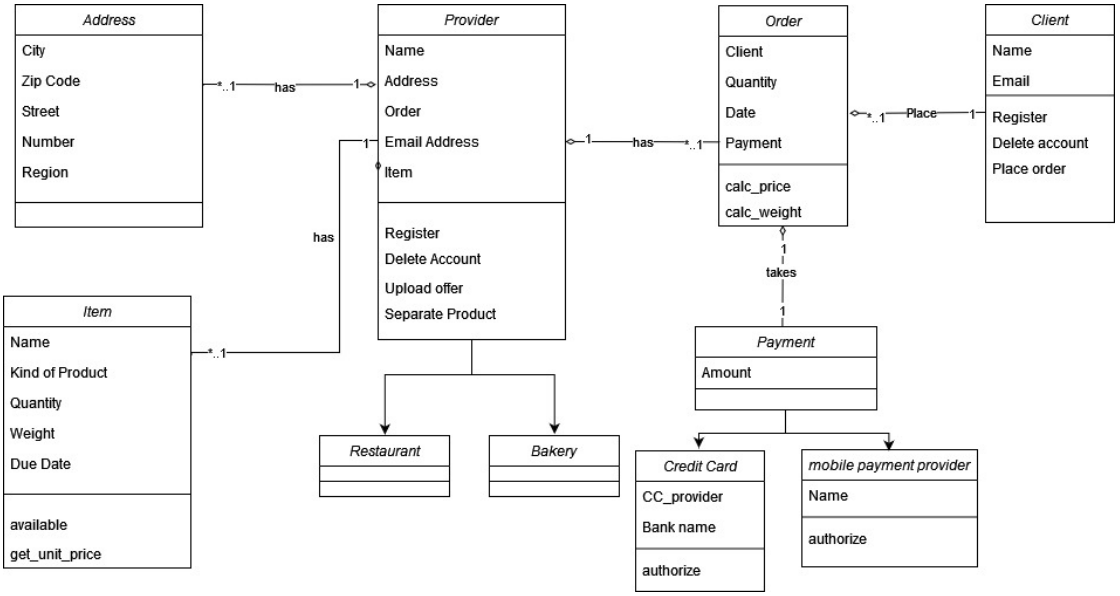

Abbildung 3: Classes of the project

Literaturverzeichnis

[AAH, 2022] AAH (2022). World hunger: Key facts and statistics 2022. *actionagainsthunger.org*. https://www.actionagainsthunger.org/world-hunger-facts-statistics, Zugriff: 18.05.2022.

[Baresi, 2009] Baresi, L. (2009). *Activity Diagrams*, pages 41–45. Springer US, Boston, MA. https://doi.org/10.1007/978-0-387-39940-9_9, Zugriff: 26.05.2022.

[Cervantes and Kazman, 2016] Cervantes, H. and Kazman, R. (2016). *Designing Software Architectures: A Practical Approach*. Pearson Education, Boston.

[FAO, 2013] FAO (2013). Food wastage: Key facts and figures. *fao.org*. https://www.fao.org/news/story/en/item/196402/icode/, Zugriff: 18.05.2022.

[FAO, 2022] FAO (2022). 17 *fao.org*. https://www.fao.org/food-loss-reduction/news/detail/en/c/1378973/, Zugriff: 18.05.2022.

[Franzen and Thoms, 2020] Franzen, E. and Thoms, M. S. (2020). Architectural drivers in modern software architecture. https://medium.com/@janerikfra/architectural-drivers-in-modern-software-architecture-cb7a42527bf2, Zugriff: 18.05.2022.

[IBM, 2004] IBM (2004). What is Class Diagram? https://developer.ibm.com/articles/the-class-diagram/, Zugriff: 18.05.2022.

[Kruchten, 1995] Kruchten, P. (1995). The 4+1 View Model of architecture. *IEEE Software*, 12(6):42–50. https://doi.org/10.1109/52.469759, Zugriff: 18.05.2022.

[UN, 2022] UN (2022). Stop food loss and waste, for the people, for the panet. *un.org*. https://www.un.org/en/observances/end-food-waste-day, Zugriff: 18.05.2022.