Smart Shopper

Mid-term Report

|  |  |
| --- | --- |
| **Version date:**  **Version number:**  **Document ID:** | **11-09-2023**  **0.1**  **SS-2023-001** |

|  |
| --- |
| **Date : 22/209/2023** |
| **Version : 0.1** |
| **State : Done** |
| **Author : Mustafa Noori** |
| **Total Words :** |
| **Words Intro-Conclusion :** |

**Version history**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author(s)** | **Changes** | **State** |
| 0.1 | 3/1/2022 | Mustafa Noori | Research, Analysis phase, Design phase |  |

**Distribution**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Receivers** |
| 0.1 | 22/09/2023 |  |

# Abstract

This mid-term report for the Smart Shoppe project provides a broad update on the progress of an innovative mobile application designed to simplify the shopping experience for consumers(expatriates) looking for the best prices(cheap) at local supermarkets. This report gives insights into the development made in research, technology stack selection, and application development. It highlights the core functionality of the application, including real-time price comparison, grocery list management, and location-based store information. Challenges encountered and future milestones are also discussed, underlining the commitment to delivering a practical solution that allows users to make informed shopping decisions while improving their grocery expenses.

# Table of Contents

[Abstract iii](#_Toc147232885)

[Table of Contents iv](#_Toc147232886)

[Table of Figures vi](#_Toc147232887)

[Introduction 1](#_Toc147232888)

[1.1 Company: CGI 1](#_Toc147232889)

[1.2 Context 1](#_Toc147232890)

[1.3 Background 1](#_Toc147232891)

[Project Overview 2](#_Toc147232892)

[1.4 Project Objective 2](#_Toc147232893)

[1.5 Scope 3](#_Toc147232894)

[1.6 Out of Scope 3](#_Toc147232895)

[1.7 Limitations: 4](#_Toc147232896)

[1.8 Approach & Phasing 4](#_Toc147232897)

[1.9 Project Team 4](#_Toc147232898)

[1.10 Project Deliverables 5](#_Toc147232899)

[1.11 Project Management 5](#_Toc147232900)

[API’s Research 7](#_Toc147232901)

[3.1 Introduction 7](#_Toc147232902)

[3.2 Literature Review 7](#_Toc147232903)

[1.12 Research Question 7](#_Toc147232904)

[1.13 APIs 8](#_Toc147232905)

[1.14 Methodology 13](#_Toc147232906)

[1.15 API Result 15](#_Toc147232907)

[1.16 Stack 15](#_Toc147232908)

[Analysis 16](#_Toc147232909)

[1.1 User group 16](#_Toc147232910)

[**1.1.1** Persona 16](#_Toc147232911)

[**1.1.2** Background 16](#_Toc147232912)

[**1.1.3** Web usage 16](#_Toc147232913)

[**1.1.4** User Story 17](#_Toc147232914)

[1.2 Functional Requirements 18](#_Toc147232915)

[**1.2.1** Use Case Diagram 18](#_Toc147232916)

[**1.2.2** Scenarios 18](#_Toc147232917)

[1.3 Data Requirements 19](#_Toc147232918)

[**1.3.1** ER (Entity Relation) Data Model 19](#_Toc147232919)

[**1.3.2** Data Dictionary 19](#_Toc147232920)

[**1.3.3** Data Constraints 20](#_Toc147232921)

[Design 21](#_Toc147232922)

[1.4 Functional Design 21](#_Toc147232923)

[**1.4.1** Window Design 21](#_Toc147232924)

[**1.4.2** Navigation Diagram 22](#_Toc147232925)

[1.5 Visual Design 22](#_Toc147232926)

[**1.5.1** Moodboard 23](#_Toc147232927)

[1.6 Technical design 23](#_Toc147232928)

[**1.6.1** Relational Model 24](#_Toc147232929)

[**1.6.2** Database Schema 24](#_Toc147232930)

[Reference 26](#_Toc147232931)

[STATEMENT OF AUTHENTICITY 27](#_Toc147232932)

# Table of Figures

[Figure 2.1: First-Fit packing items 5](#_Toc95304378)

[Figure 2.2: Next-Fit packing items 6](#_Toc95304379)

[Figure 2.3: Best-Fit packing items 6](#_Toc95304380)

[Figure 2.4: First-Fit Decreasing sorting items 7](#_Toc95304381)

[Figure 2.5: Modified-First Fit-Decreasing sorting items 7](#_Toc95304382)

[Figure 2.6: use story GitLab board 8](#_Toc95304383)

[Figure 3.1: Flowchart Beech implementation 11](file:////Users/mustafanoori/Downloads/finalRepver3.docx#_Toc95304384)

[Figure 3.2: Simplified flowchart Beech implementation 12](#_Toc95304385)

[Figure 3.3: Beech algorithm into modules 14](#_Toc95304386)

[Figure 4.1: Layered Architecture 18](#_Toc95304387)

[Figure 4.2: Event-driven architecture 19](#_Toc95304388)

[Figure 4.3: Microservices architecture 21](#_Toc95304389)

[Figure 4.4: Golang stats 22](#_Toc95304390)

[Figure 4.5: C stats 22](#_Toc95304391)

[Figure 4.6: C++ stats 23](#_Toc95304392)

[Figure 4.7: C# stats 23](#_Toc95304393)

[Figure 4.8: Java stats 23](#_Toc95304394)

[Figure 4.9: JavaScript stats 24](#_Toc95304395)

[Figure 4.10: Python stats 24](#_Toc95304396)

[Figure 4.11: Ruby stats 24](#_Toc95304397)

[Figure 4.12: Kotlin stats 25](#_Toc95304398)

[Figure 4.13: TDD test case example 27](#_Toc95304399)

[Figure 4.14: TDD example test implementation 27](#_Toc95304400)

[Figure 4.15: TDD test after implementation example 28](#_Toc95304401)

[Figure 4.16: GitLalgorithm modules 31](#_Toc95304402)

# Introduction

## Company: CGI

CGI is a global IT and business consulting services company founded in 1976. With about 90,000

professionals across 400 locations, they provide comprehensive, scalable consulting services across 21 industries. In CGI Eindhoven, where I'll be working, teams of 200 professionals focus on innovative technology solutions to address practical challenges in various sectors.

CGI's expertise spans finance, government, healthcare, energy, and telecommunications. They're known for cutting-edge solutions that drive digital transformation and leverage emerging technologies to meet evolving industry needs. CGI boasts a rich history of providing cutting-edge technology solutions to clients across diverse industries, solidifying its position as a trusted partner in the world of IT and consultancy.

## Context

In recent years, the grocery shopping landscape has undergone significant transformations. The rise of online shopping has brought convenience to consumers' doorsteps. However, amidst this convenience, several challenges have emerged. Shoppers are now seeking more efficient and cost-effective ways to complete their grocery purchases. This change in consumer behavior has created a pressing need for innovative solutions.

## Background

The Smart Shopper project has been conceived to address the multifaceted challenges of online grocery shopping. Today's consumers face several issues while shopping for groceries online, such as unpredictable price variations, limited product information, complex decision-making, time-consuming processes, and missed opportunities for savings.

Online shoppers are often left to manually compare prices across multiple supermarkets, leading to inefficiencies and sometimes even dissatisfaction. Recognizing these challenges, CGI has undertaken the Smart Shopper project with the aim of simplifying the online grocery shopping experience. The project's primary goal is to empower consumers with a user-friendly mobile application that enables them to create grocery lists, receive real-time price information, discover current offers, and optimize their shopping across various supermarkets.

This project aligns with CGI's commitment to providing innovative solutions that enhance consumers' lives, save them valuable time and money, and streamline the online grocery shopping process.

# Project Overview

The Smart Shopper assignment aims to transform the online grocery shopping experience by developing a mobile application. This app will allow expatriates to create grocery lists, receive real-time information on the most affordable prices, discover current offers, and closeest supermarkets, and identify opportunities for bulk purchases across various supermarkets. By simplifying the online grocery shopping process and providing cost-effective decision-making tools, this project seeks to save users both time and money. With a focus on enhancing the efficiency and ease of grocery shopping, the Smart Shopper app aligns with the evolving needs of modern consumers.

## Project Objective

The main objective of the Smart Shopper project is to design and develop an intuitive mobile app that will empower users(expats) to create, manage, and optimize their grocery lists efficiently. The application aims to provide real-time price comparison functionality for grocery items across multiple supermarkets. Users should be able to input their shopping lists, and the app will analyze and display the most cost-effective options. To enhance the shopping experience, the application will include a store locator feature. Users will be able to find nearby supermarkets, view their details, and access additional information like store hours and contact information. To cater to a diverse user base, the project will incorporate multilingual support, allowing users to interact with the application in their preferred language. Implement image recognition technology to enable users to add items to their grocery lists by taking pictures. This feature enhances the user interface and simplifies list creation.

The application will facilitate sharing and collaboration among users. Friends or family members can share their grocery lists, making it easier to coordinate shopping. Ultimately, the project aims to streamline the grocery shopping process for users, saving them time and money. By providing accurate pricing information, store details, and collaboration features, the application empowers users to make informed choices and optimize their shopping trips.

To define the SMART objectives of the Smart Shopper project.

* Specific: Develop a mobile application that allows users to create and compare grocery lists across various supermarkets.
* Measurable: Achieve a 20% reduction in users' monthly grocery expenses.
* Achievable: Complete the development and launch of the mobile app within a 12-month timeline.
* Relevant: Provide a user-friendly interface with multi-language support for international users.
* Time-bound: Complete the project within the given timeframe.

## Scope

To explain the project's scope, including its various components and limitations. The scope of the Smart Shopper project covers the following areas:

|  |  |
| --- | --- |
| Scope areas | Explain scope |
| Business process / portfolio | This includes improving consumers' grocery shopping experience and managing different data collection of products from supermarkets. |
| Geographical | Availability for expatriates in the Netherlands |
| Organizational | Collaboration with supermarkets and gathering real-time pricing data. |
| Application | It outlines the features of the Smart Shopper mobile application functionality, including its user interface where expatriates will have the ability to create a shopping list, look at products, and compare prices. |
| Information | It relates to the data processed by the application, including product pricing, discounts, promotions, user-generated grocery lists, and user profiles. |
| Infrastructure | This involves applying CGI's computer and network infrastructure and offers scalability, security, and technical support to ensure the efficient development and deployment of the mobile app. |

## Out of Scope

To give a clear image of the conclusion and prevent unrealistic expectations, it is crucial to specify what is outside of scope. The following will not be the project's responsibility:

* Local Stores: Repositories for information that aren't available to the public.
* Store Operations: The project is not in charge of overseeing the actual layout or running of the store.
* Payment: Although the app offers price information, it does not permit in-app purchases.
* Delivery Services: No service options are offered by the app.
* Expansion: The app's geographic reach is restricted to a certain area, and it does not function abroad.
* Distribution: The app will not be made available through open channels like app stores.
* A multilingual app with many languages: (n > 2).

## Limitations:

The project does not involve physical retail operations but focuses on the digital platform. Language accessibility for international users may be limited initially.

## Approach & Phasing

Objective: To detail the approach and phases planned for the project.

#### Approach

The project will investigate the availability of supermarket APIs and select the technology stack. Define user requirements, acceptance criteria, and deliverables. Create the application's architecture, user interface, and database structure. Develop the mobile app, implement testing procedures, and refine user experience. Lastly, documentation, conducting user training and preparing for the app's launch.

#### Phasing

The project phases are setting up project infrastructure and team formation. Investigate supermarket APIs and select the technology stack. Define user requirements and project scope. Create application architecture and user interface. Develop the mobile app and perform rigorous testing. Finalize documentation and prepare for launch.

## Project Team

|  |  |
| --- | --- |
| Teams | |
| **Name(s)** | **Role** |
| Johan Peters | Project Supervisore |
| Jelle Baede | CGI Coach |
| Denis Gorianin | Stakeholder |
| Mustafa Noori | Developer |

## Project Deliverables

The project has been divided into the following milestones which represent the important aspects of this project.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Phase | Milestones | Planned date | Major Deliverables | Comments |
| Project Preparation | M1: Project Initiation & Team Formation | 01-09-23 | Project Charter, Team Formation | Initiate the project and establish the team. |
| Design | M2: Requirements Analysis & System Design | 01-10-23 | Requirements Documentation, System Design | Define project scope, design the system. |
| Realization | M3: Application Development & Testing | 15-10-23 | Developed Application, Testing Reports | Build and thoroughly test the application. |
| Final Preparation | M4: Infrastructure Setup & Data Integration | 01-11-23 | Infrastructure Ready, Data Integrated | Prepare technical environment and data. |
| Go Live and Support | M5: Application Deployment & User Support | 01-12-23 | Application Deployed, User Support in Place | Launch the application and support users. |
| End of Project | M6: Project Evaluation & Closure | 15-01-24 | Project Evaluation Report, Assessment, Closure Activities | Assess project outcomes, close the project. |

## Project Management

Project management is essential to ensure the successful planning, execution, and completion of all project tasks and objectives. In the developing process of the Smart Shopper application, an agile approach will be used by conducting a daily stand-up and a two-week sprint. This will allow to break down the project into smaller chunks, which will be more manageable and easier to focus tasks which are called use cases.

Figure 1-Personal Trello board

At the start of a sprint tasks will be prioritized using "planning poker," a technique that involved assigning numerical values to tasks based on complexity. This helped to focus on tasks following the MoSCoW principle which stands for must have, should have, could have, and would like to have.

At the end of each sprint, a sprint retrospective will be held to review progress and identify areas for improvement. This includes reflecting on what had gone well during the sprint, as well as identifying any challenges or issues that had arisen and developing strategies to address them.

Regular progress reports will also be provided to the project steering committee (PSC) at the end of each sprint. These reports included an overview of completed tasks during the sprint, any challenges or issues that had risen and how they were addressed, and an update on overall progress, including key milestones and upcoming challenges or deadlines.

Here are the key aspects of project management in this initiative:

* A detailed project plan is developed, outlining all project phases, activities, timelines, and milestones. This plan serves as a roadmap for the project team, helping stay organized and focused on achieving project objectives. Clear and well-defined project scope are crucial. Scope outlines the features and functionalities of the Smart Shopper application, ensuring that the project stays within its intended boundaries.
* Identifying and mitigating risks is a continuous process throughout the project's lifecycle. The project team will assess potential risks and develop strategies to minimize their impact on project outcomes. Quality assurance processes will be integrated into each phase of the project. This includes testing, quality control, and validation of deliverables to ensure that the final product meets user expectations.
* Effective communication is vital for project success. Within the project clear communication channels within the team and with stakeholders are established to ensure everyone is informed about project progress and any changes.
* A change management process will be in place to handle any changes to project scope, objectives, or requirements. Changes will be evaluated for their impact on the project, and decisions will be made accordingly. Thorough documentation will be maintained throughout the project, including project plans, technical documentation, and user manuals, to support project transparency and future maintenance.
* At the end of the project, a closure phase will ensure that all project objectives have been met, and deliverables are successfully handed over to the relevant parties. A final report will summarize the project's outcomes and lessons learned.

# API’s Research

## Introduction

This chapter investigates several API choices and other alternative methods while assessing their applicability to the project. Based on the research findings, recommendations on the best method(s) to utilize are made after discussing the benefits and drawbacks of each strategy.

## Literature Review

It is crucial to conduct a comprehensive examination of the APIs and web scraping techniques that are now available and that can supply information on the products at various Dutch grocery stores before beginning to construct the project. APIs provide a more efficient and reliable technique for getting data than web scraping, which also allows for more customization and flexibility.

Finding APIs and web scraping techniques that can provide details on the products, deals, and prices at big Dutch supermarkets is the main objective of this study. This data will be enhanced to the Smart Shopper application's database to give expatriates up-to-date, precise information regarding products and prices.

In the initial phase of the Smart Shopper project, extensive research into supermarket APIs was conducted to assess their availability, usability, and relevance to the project's objectives. For Smart Shopper implementation it's important to research the available APIs and other alternative techniques for web scraping of different supermarket data in the Netherlands.

This chapter investigates several API choices and web-scraping methods while assessing their applicability to the project. Based on the research findings, recommendations on the best method(s) to utilize are made after discussing the benefits and drawbacks of each strategy.

## Research Question

The primary goal of the API research was to determine whether major supermarkets provide APIs or data feeds that can be integrated into the Smart Shopper mobile application. These APIs would serve as a crucial source of real-time pricing data and product information.

How can the functionality of the Smart Shopper mobile application be improved through the incorporation of outside grocery store data APIs to provide real-time pricing and store information, ultimately optimizing users' grocery shopping experiences?

This research question examines how integrating external APIs, a crucial project component, could improve the application's functionality and user experience in line with project objectives. It also suggests that the study would investigate how this relationship improves consumers' grocery shopping efficiency, which is one of the key benefits of the Smart Shopper program.

## APIs

These findings are instrumental in shaping the project's direction and determining the technical architecture for retrieving real-time supermarket data, a critical component of the "Smart Shopper" application. there are many other APIs and data sources available here are a few to consider for the Netherlands markets:

* Checkjebon API
* ShoppingScraper API
* Superscanner API
* Websrapping.Amsterdam API

|  |  |
| --- | --- |
| **Criteria** | **Detail** |
| **API** | Checkjebon API |
| **Description** | Checkjebon API is designed for price comparison and product information retrieval. It allows developers to access and use data related to various products from different sources. Its an open source project the data can be used freely. |
| **Link** | [Checkjebon API Documentation](https://www.checkjebon.com/api) |
| **Type of Data Format** | The API provides data in JSON format, which is widely supported and easy to work with. |
| **NL Grocery Stores List** | The API covers a range of products, including grocery items of various supermarkets and some of them are as follow:   * Albert Heijn * Aldi * Coop * DekaMarkt * Dirk * Hoogvliet * Jan Linders * Jumbo * Plus * SPAR * Vomar |
| **Price** | It is free to use and other pricing details for API usage are available on the website. It offers different subscription tiers. |
| **Data Accuracy** | Data accuracy can vary based on the accuracy of data from source websites. It's advisable to verify data for critical applications. |
| **Data Update** | Checkjebon API offers regular updates based on changes in source websites. |
| **Usage Limits** | The API imposes rate limits, and the number of requests allowed depends on the subscription tier. |
| **Security** | Access to the API is secured using an API key for authentication and access control. |
| **Developer Support** | The API provides developer documentation for integration. |
| **User Review** | Limited user reviews are available, suggesting a smaller user community. |
| **Community and Documentation** | While documentation is available, it may not be as comprehensive as some other APIs. |
| **Licensing and Term of Use** | The website specifies terms of use for API access. |
| **Advantages** | * Free to use. * Designed for price comparison and product information retrieval. * Provides data in JSON format. * Regular updates based on changes in source websites. * Offers different subscription tiers. * Secured access using an API key. * Covers a range of products, including grocery items. |
| **Disadvantages** | * Data accuracy can vary based on the accuracy of data from source websites. * Documentation may not be as comprehensive as some other APIs. * Limited user reviews and a smaller user community. * Pricing may not be as competitive compared to some alternatives. |

|  |  |
| --- | --- |
| **Criteria** | **Detail** |
| **API** | Shopping Scraper API |
| **Description** | Shopping Scraper API is a web scraping service designed for retrieving product info, product images, and price information from various online retailers. It allows developers to access and integrate scraped data into their applications. |
| **Link** | [Shopping Scraper API Documentation] (https://www.shoppingscraperapi.com/) |
| **Type of Data Format** | The API provides data in JSON format, making it easy to work with and integrate into applications. |
| **NL Grocery Stores List** | The API does not specifically mention a list of NL grocery stores. It appears to focus on online retailers in general. But it contains data for below NL supermarkets:   * Alber Heijn * Jumbo |
| **Price** | Pricing details are available on the website, including different subscription plans as below. It offers a free tier with limited usage.   * Essential: €129/mo * Growth: €199/mo * Advanced: €399/mo * Enterprise: €749+/mo |
| **Data Accuracy** | The accuracy of data scraped by the API may vary depending on the source websites. Users should verify data for critical applications. |
| **Data Update** | The API regularly updates data based on changes in the source websites. |
| **Usage Limits** | Usage limits vary depending on the subscription plan selected, with higher tiers offering more requests. |
| **Security** | API access is secured using an API key for authentication and access control. |
| **Developer Support** | The API provides developer documentation and support via email. |
| **User Review** | Limited user reviews are available, indicating relatively fewer users compared to some other APIs. |
| **Community and Documentation** | While documentation is available, it appears to be relatively basic. The API seems to have a smaller user community. |
| **Licensing and Term of Use** | The website specifies the terms of use and licensing for API access. |
| **Advantages** | * Provides access to scraped product and price data from various online retailers. * Offers data in JSON format for easy integration into applications. * Regularly updates data based on changes in source websites. * Offers a free tier with limited usage, making it cost-effective for small projects. * Secured access using an API key. * Developer support via email. |
| **Disadvantages** | * Accuracy of scraped data may vary depending on source websites. * Basic documentation may require additional effort for integration. * Smaller user community and limited user reviews. * Limited coverage and focus on specific types of products. |

|  |  |
| --- | --- |
| **Criteria** | **Detail** |
| **API** | Super scanner API |
| **Description** | Superscanner API is a price comparison service that allows users to compare prices of various products across different online stores. It provides developers with access to its data. |
| **Link** | [Superscanner API Documentation](https://superscanner.io/api-docs) |
| **Type of Data Format** | The API offers data in JSON format, which is commonly used and easy to work with. |
| **NL Grocery Stores List** | While the API focuses on various products, it does include a category for grocery products of the below supermarkets.   * Albert Heijn * Jumbo * Makro * Dirk * Nettorama * PicNic * Plus * Spar * Hoogvliet |
| **Price** | The API offers a free tier with rate limits. Pricing details for higher tiers are available on the website. |
| **Data Accuracy** | Data accuracy may vary based on the availability and accuracy of data from online stores. It's advisable to verify data for critical applications. |
| **Data Update** | Superscanner API provides regular updates based on the data from online stores. |
| **Usage Limits** | The API has rate limiting, but higher tiers offer more requests allowed. |
| **Security** | Access to the API requires an API key for authentication and security. |
| **Developer Support** | While documentation is available with some examples, it may be considered relatively basic. |
| **User Review** | Limited user reviews are available, suggesting a smaller user base. |
| **Community and Documentation** | Documentation is available on the website but may not be as extensive as some other APIs. |
| **Licensing and Term of Use** | The website specifies the terms of use for API access. |
| **Advantages** | * Allows users to compare prices of various products across different online stores. * Data provided in JSON format for easy integration. * Regular updates based on data from online stores. * Offers a free tier with rate limits. * Access is secured using an API key. * Covers a range of product categories. |
| **Disadvantages** | * Data accuracy may vary based on the availability and accuracy of data from online stores. * Basic documentation. * Limited user reviews and a relatively smaller user base. * Documentation may not be as comprehensive as some other APIs. |

|  |  |
| --- | --- |
| **Criteria** | **Detail** |
| **API** | Websrapping.Amsterdam API |
| **Description** | Websrapping.Amsterdam offers web scraping services for various types of data, including product and price information. It allows developers to access and retrieve data from websites. |
| **Link** | [Websrapping.Amsterdam API Documentation](https://webscraping.amsterdam/api) |
| **Type of Data Format** | The API provides data in JSON format, a widely used and supported format. |
| **NL Grocery Stores List** | The API's focus is not specifically mentioned, but it appears to cover a broader range of websites rather than being limited to grocery stores and they are as follow:   * Albert Heijn * Jumbo * Coop * Lidl * Plus * Ekoplaza * Dirk |
| **Price** | Pricing details for API usage are available on the website, including different pricing tiers as below: |
| **Data Accuracy** | Data accuracy may vary based on the source websites from which data is scraped. |
| **Data Update** | Updates depend on changes in the source websites. |
| **Usage Limits** | The API specifies rate limits, and the number of requests allowed depends on the pricing tier. |
| **Security** | API access is secured using an API key for authentication. |
| **Developer Support** | Developer documentation is provided, along with contact information for support. |
| **User Review** | Limited user reviews are available, indicating a relatively smaller user base. |
| **Community and Documentation** | While documentation is available, the user community may be smaller compared to some other APIs. |
| **Licensing and Term of Use** | The website specifies terms of use for API access. |
| **Advantages** | * Offers web scraping services for various types of data. * Provides data in JSON format. * Regular updates based on changes in source websites. * Offers different pricing tiers. * Secured access using an API key. * Developer documentation provided. |
| **Disadvantages** | * Focus may not be specific to grocery stores; covers a broader range of websites. * Data accuracy may vary based on the source websites. * Limited user reviews and a smaller user community. * Documentation may require additional effort for integration. |

## Methodology

The API research yielded the following key findings:

* Availability: Several major supermarket chains, including [List Supermarket Chains], offer APIs or data feeds for accessing pricing and product information.
* Documentation: API documentation for [List Supermarket Chains] is available, providing details on endpoints, data formats, and usage guidelines.
* Technical Feasibility: Initial assessments indicate that integrating these APIs into the "Smart Shopper" mobile application is technically feasible.

There were several steps in the method used to carry out the API research. The research involved a systematic approach:

* Online Research: Extensive online searches and documentation reviews were conducted to identify potential APIs or data sources provided by major supermarkets. First, the data needs for the Smart Shopper application were determined, particularly the requirement for current data on prices and promotions at Dutch supermarkets. A thorough research for existing APIs that offer the required data was done. This required Googling specific search terms on search engines and social media networks, as well as browsing online API directories like ProgrammableWeb and RapidAPI.
* API Documentation Review: If APIs were found, their documentation was reviewed to assess their compatibility and suitability for the project's objectives. Finally, a proper looked into the potential of creating an own data through web scraping in the event that the accessible APIs did not meet the necessary criteria. this API research's technique was meticulous and rigorous, guaranteeing that the Smart Shopper application will have access to the most trustworthy and pertinent data from the chosen API.
* Technical Feasibility: An assessment of the technical feasibility of integrating identified APIs into the mobile application was performed. The collected APIs were assessed according to a few factors, including the data's accessibility and accuracy, simplicity of integration, price, and support. To identify which API was most suited for the Smart Shopper application, a comparative analysis of those that fit the selection criteria was done.

## API Result

The Checkjebon API has been found to be the most appropriate for the project based on the evaluation. It offers information about the goods, costs, and special offers at the largest supermarkets in the Netherlands, and it covers a wide range of establishments and goods. The API is also rather simple to use and works nicely with the application. Utilizing the API comes at a fair price that is within the project's budget (Free).

The constraints of the other APIs considered made them less suitable for the purpose. For instance, both the Webscraping and the ShoppingScraper API only provided a very limited amount of coverage for Dutch grocery stores. Amsterdam API, which was also the most expensive API, offered data in a format that was not acceptable for the application.

## Stack

Mobile App Development Framework:

* React Native: If you prefer a cross-platform solution with a single codebase, React Native is a popular choice. It allows you to build native-like mobile apps using JavaScript and React.

Backend Development:

* Node.js: If you need to build a backend server or API for your application, Node.js is a versatile and popular choice for server-side development.

Database:

* Non-functional
* SQLite: A lightweight, embedded database that's suitable for mobile app development. Both Android and iOS support SQLite.

Version Control:

* Git: Use Git for version control to track changes in your codebase and collaborate with the development team effectively.
* Jira(Project Board)

# Analysis

During the analysis phase of the Smart Shopper project, a comprehensive approach was taken to meticulously gather and document both the functional and non-functional requirements of the application. This critical phase involved a series of systematic steps, including extensive user interviews, market research, and competitive analysis, to ensure a thorough understanding of user needs and industry trends. Use cases were meticulously identified and documented, highlighting the various scenarios in which the application would be utilized. Additionally, detailed diagrams and tables were crafted to provide a visual representation of the project's scope and requirements. This rigorous analysis phase was essential in shaping the foundation of the Smart Shopper application, aligning it closely with user expectations and market demands.

## User group

A user group is type of people mainly focus toward technology (software and computer), but not always computer associated.

### Persona

Persona is the characteristic of a fictious person such as name, age, marital status, address, education, hobbies, pets and use on daily basis computer, mobile smart phones and internet, who are "Budget-Conscious Shopper," "Busy Parent," and "Tech-Savvy Millennial." For example, the "Budget-Conscious Shopper" persona could be named Sarah, a 32-year-old office worker. She's budget-conscious, prefers online shopping, and often buys groceries for her family.

### Background

Sarah, the Budget-Conscious Shopper, typically check groceries prices online due to her Budget-Conscious and saving money. She has a moderate-sized grocery list, often includes fresh produce, and looks for deals to save money.

### Web usage

Sarah uses her smartphone and laptop daily. She's comfortable with mobile apps for shopping and frequently uses websites to compare prices.

### User Story

Sarah's user story might be: "As a Budget-Conscious Shopper, I want to compare prices across different supermarkets to ensure I'm getting the best deals on my grocery list."

## Functional Requirements

Functional requirements are declaration of a structure and its mechanisms based on the requirements that a designer determines for a software or device functional behavior as expected for a wanted input.

### Use Case Diagram

The use case diagram is the dynamic assortment of all steps and approach of all user group in a one diagram. It prohibits sets of actions, services and function of a system by using actor and use cases. Figure 1: Use Case Diagram depicts the use case diagram that shows use cases that are related to the Smart Shopper.

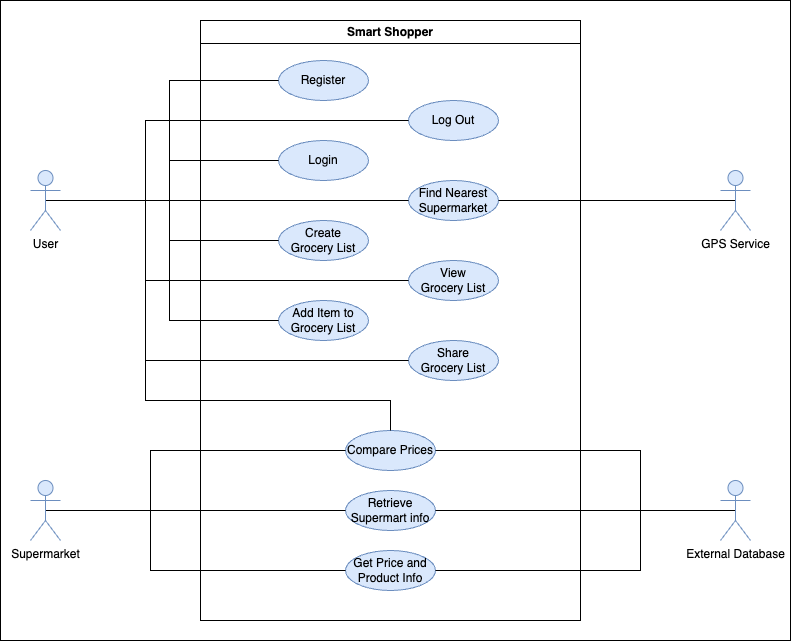


Figure 1: Use Case Diagram

### Scenarios

A scenario written outline of a stags of collaboration between an actor (a member of a user group) and the system.

#### Example of UC11 – **Search for Products**

|  |  |
| --- | --- |
| **ID** | 11 |
| **Name** | **Search for Products** |
| **Priority** | High |
| **Description** | This use case allows users to search for specific grocery products within the application. |
| **Actor** | Shopper |
| **Precondition** | User is logged in. |
| **Scenario** | 1. User opens the Smart Shopper app. 2. User enters a product name or keyword in the search bar. 3. The application displays a list of products matching the search query. 4. User selects a product from the list to view more details. |
| **Exceptions** | If no results match the search query, the application displays a message indicating that no products were found. |
| **Extensions** | None |
| **Postconditions** | User can view product details or proceed to add the product to their shopping list. |

#### UC12 –User Registration

|  |  |
| --- | --- |
| **ID** | 3 |
| **Name** | User Registration |
| **Priority** | High |
| **Description** | This use case describes the process by which a new user registers for an account in the Smart Shopper application to access its features. |
| **Actor** | * **Primary Actor**: New User * **Secondary Actors**: None |
| **Precondition** | * The user has downloaded and installed the Smart Shopper application. * The user has not yet registered for an account. |
| **Scenario** | 1. The user opens the Smart Shopper application. 2. The application presents the user with a registration screen. 3. The user enters their personal information, including their name, email address, and password. 4. The user confirms their password to ensure accuracy. 5. The user submits the registration form. 6. The application validates the user's inputs. 7. If the inputs are valid, the application creates a new user account. 8. The application logs the user into their new account. 9. The user is redirected to the application's main screen. |
| **Exceptions** | **User Already Registered**:   * If the user's email address is already registered in the system, the application displays an error message, indicating that the email is already in use.   **Invalid Inputs**:   * If the user provides invalid or incomplete information, the application displays an error message and prompts the user to correct the information. |
| **Extensions** | None |
| **Postconditions** | * The user's registration information is saved in the system. * The user is logged into their new account. |

A diagram of a software system

Description automatically generated

## Data Requirements

Data requirements is about the data used and produced throughout the procedure of achieving a goal by users.

### Data Dictionary

A unified source of metadata or all term glossary for an application domain during the analysis phase. The main purpose of creating it is to understand the terms of analyzing it before starting to design data modeling. The follow Table 1: Data DictionaryTable 1: Data Dictionary has been used as its example.

|  |  |  |
| --- | --- | --- |
| **User Data** | | |
| **Data:** | **Type:** | **Description:** |
| User ID | Numeric | A unique identifier for each registered user. |
| Username | Alphanumeric | The chosen username of the user. |
| Email | String (Email Format) | The email address associated with the user's account. |
| Password | String (Hashed) | The user's password for account authentication. |

Table 1: Data Dictionary

|  |  |  |
| --- | --- | --- |
| **Grocery Data** | | |
| **Data:** | **Type:** | **Description:** |
| Grocery List ID | Numeric | A unique identifier for each user's grocery list. |
| Product Name | Alphanumeric | The name of a grocery product. |
| Product ID | Numeric | A unique identifier for each grocery product. |
| Product Price | Decimal | The price of a grocery product in euros. |

|  |  |  |
| --- | --- | --- |
| **Supermarket Data** | | |
| **Data:** | **Type:** | **Description:** |
| Supermarket Name | Alphanumeric | The name of a supermarket or store. |
| Supermarket location | Alphanumeric | The physical location (address) of a supermarket. |

|  |  |  |
| --- | --- | --- |
| **Location Data** | | |
| **Data:** | **Type:** | **Description:** |
| Location Data | Geographic Coordinates (Latitude, Longitude) | The user's current geographical location. |

|  |  |  |
| --- | --- | --- |
| **Product Image Data** | | |
| **Data:** | **Type:** | **Description:** |
| Product Image | File (Image) | An image representing the grocery product for a user-friendly UI. |

### Data Constraints

Data constraint explains DO and DON’T in the value assortment of attributes and entities while analyzing, constraints are verbally explained. The following Table 2: Data Constraints has been demonstrated as its example.

|  |  |  |
| --- | --- | --- |
| **Data:** | **Description:** | **CONSTRAINS TYPE** |
| User ID | A unique identifier for each registered user. | Numeric, auto-increment, unique. |
| Username | The chosen username of the user. | Alphanumeric, 4-20 characters, unique. |
| Email | The email address associated with the user's account. | Valid email format, unique. |
| Password | The user's password for account authentication. | At least 8 characters, a mix of upper and lower case letters, numbers, and special characters. |
| Grocery List ID | A unique identifier for each user's grocery list. | Numeric, auto-increment, unique. |
| Product Name | The name of a grocery product. | Alphanumeric, 2-50 characters. |
| Product ID | A unique identifier for each grocery product. | Numeric, auto-increment, unique. |
| Product Price | The price of a grocery product in euros. | Numeric, two decimal places. |
| Supermarket Name | The name of a supermarket or store. | Alphanumeric, 2-100 characters. |
| Supermarket Location | The physical location (address) of a supermarket. | Alphanumeric, up to 255 characters. |
| User Location | The user's current geographical location. | Latitude and longitude coordinates. |
| Product Image | An image representing the grocery product for a user-friendly UI. | Image file format (e.g., JPG, PNG). |

Functional Requirements:

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Title** | **Priority** | **Description** |
| FR-1 | User Registration | High | * Users can create accounts with their personal information. * Registration requires a valid email address and password. * User profiles store preferences and shopping history. |
| FR-2 | Product Search | High | * Users can search for grocery products by name or category. * The system provides real-time search suggestions. * Results include product names, prices, and store information. |
| FR-3 | Price Comparison | High | * Users can compare prices of the same product across different supermarkets. * Price comparisons are displayed clearly for each product. * The system identifies the cheapest option and highlights it. |
| FR-4 | Creating and Managing Lists | Medium | * Users can create, edit, and delete shopping lists. * Lists can be customized with product quantities. * Users can set budget limits for their lists. |
| FR-5 | Store Locator | Medium | * Users can find nearby supermarkets on a map. * The system provides directions to selected stores. * Store details include address, hours, and contact information. |
| FR-6 | Sharing Shopping Lists | Medium | * Users can share their created shopping lists with others. * Shared lists can be accessed and edited collaboratively by recipients. |

Non-Functional Requirements:

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Title** | **Priority** | **Description** |
| NFR-1 | Performance | High | * The application must provide real-time price comparisons, ensuring fast response times. * It should handle concurrent user requests efficiently, even during peak hours. |
| NFR-2 | Usability | High | * The user interface must be intuitive and user-friendly. * Navigation should be straightforward, even for first-time users. * Search suggestions and filters should enhance the user experience. |
| NFR-3 | Data Accuracy | High | * Price and product information must be accurate and up to date. * Store details and locations should be precise |
| NFR-4 | Security | Medium | * User data, including personal information and lists, must be securely stored. * Secure login and data encryption are necessary. |
| NFR-5 | Scalability | Medium | * The system should be scalable to accommodate future growth in users and data. * It should support additional supermarkets and products. |
| NFR-6 | Compatibility | Medium | * The mobile app should be compatible with a range of Android and iOS devices. * Web compatibility should cover major browsers. |
| NFR-7 | Availability | Low | * The application should be available 24/7, with minimal downtime for maintenance. |
| NFR-8 | Accessibility | Low | * The app should comply with accessibility standards, ensuring usability for users with disabilities. |
| NFR-9 | User Engagement | Low | * Features like user reviews and ratings aim to enhance user engagement and trust. |

# Design

Design is a plan or an overview of an object for its construction or implementing a process or an activity for all its stages. There are important steps to achieve a goal for better a design and constraints, may consider visual, practical, socio-political or financial considerations, and might intermingle certain setting. Main example can be designs include patterns, diagrams, blueprints and drawing process etc.

## Functional Design

Functional design is used as a model to simplify hardware and software devices designs like 3D modeling and software models. Each modular also assure the functionality assigned to the part to decrease the side effect of its responsibility to the minimum level as it can.

### Window Design

Window design is a process of the appearance of a web page and to select right color scheme, proper layout with appropriate font with correct text size. Every window has its own font, color, layout and design, but in general all pages should have similar graphic design. The following Figure 3: Window Design demonstrate as example.

A screenshot of a checkout page

Description automatically generated

Figure 3: Window Design

### Navigation Diagram

Navigation diagram main purpose is to demonstrate the navigation on a home page to the other pages as shown in the Figure 4: Navigation Diagram, whereas pages are shown as individual window or page, with the page name above. Arrows have been used to point out the navigation from one page to another and shows navigation between the all pages.

A screenshot of a computer program

Description automatically generated

Figure 4: Navigation Diagram

## Visual Design

Visual design is the using of imager font, color, layout, design, typography and shapes forms to increase and improve usability of experiences. Field like UI design and graphic design have grown out instantly.

### Moodboard

A Moodboard is an arrangement of images or creative composition of picture, material pieces of text elements intended concept to present an atmosphere. A Moodboard is also used as to clarify equivocal, which are hard to explain ideas or generate ideas for designer and the collation customer. As an example, below Figure 5: Moodboard represent the combination of images, color and font used together for the windows design.

A collage of images of people in different poses

Description automatically generated

Figure 5: Moodboard

## Technical design

Technical design is a solution to the activities and its problem while developed designed has been completed. A designer co-ordinates technical design before even actual design begin.

### Relational Model

The Relational Model is a way of handling the data and it structure consistency, where all data is characterized as group into relation in terms of tuples. In the below example Figure 6: Relational Model has been illustrated with its primary keys and foreign key. A screenshot of a computer program

Description automatically generated

Figure 6: Relational Model

### Database Schema

A database schema is a blueprint of a structure and design and its constraint and relation. The very core stage of creating any software or webpage. Below Figure 7: Data Schema depict how it were generated.

A screenshot of a computer program

Description automatically generated

Figure 7: Data Schema

# Reference

A. (2021, April 1). *Perl, Python, Ruby, PHP, C, C++, Lua, tcl, javascript and Java benchmark/comparison.* Blog[spot. http://onlyjob.blogspot.com/2011/03/perl5-python-](http://onlyjob.blogspot.com/2011/03/perl5-python-) ruby-php-c-c-lua-tcl.html

Aken, M. V. (2019). *An efficient bin-packing algorithm for packing groceries in a fulfillment center.* Delft: Delft University of Technology.

Akin, M. (2021). *Two dimensional bin packing.* Opgehaald van researchgate: https:/[/www.researchgate.net/publication/343880146\_Bin\_Packing\_Problem\_Indust](http://www.researchgate.net/publication/343880146_Bin_Packing_Problem_Indust) rial\_Engineering\_Operational\_Research

Albers, S., & Mitzenmacher, M. (2000). Anverage-Case Analyses of First Fit and Random Fit Bin Packing. *Random Structures 16.*, 240-259.

Amazon Web Services, Inc. Retrieved April 9, 2021, from https://aws.amazon.com/amazon-mq/?amazon-mq.sort- by=item.additionalFields.postDateTime&amazon-mq.sort-order=desc

Analytics India Magazine. https://analyticsindiamag.com/9-best-kubernetes- alternatives-for-devops-engineers/

# STATEMENT OF AUTHENTICITY

Issued by the FHTenL Examination Board, September 2017

I, the undersigned, hereby certify that I have compiled and written this document and the underlying work / pieces of work without assistance from anyone except the specifically assigned academic supervisor. This work is solely my own, and I am solely responsible for the content, organization, and making of this document and the underlying work / pieces of work.

I hereby acknowledge that I have read the instructions for preparation and submission of documents/pieces of work provided by my course / my academic institution, and I understand that this document and the underlying pieces of work will not be accepted for evaluation or for the award of academic credits if it is determined that they have not been prepared in compliance with those instructions and this statement of authenticity.

I further certify that I did not commit plagiarism, did neither take over nor paraphrase (digital or printed, translated or original) material (e.g., ideas, data, pieces of text, figures, diagrams, tables, recordings, videos, code, ...) produced by others without correct and complete citation and correct and complete reference of the source(s). I understand that this document and the underlying work / pieces of work will not be accepted for evaluation or for the award of academic credits if it is determined that they embody plagiarism.

Name (in capital letters): MSUTAFA NOORI

Student number: 3014630

Place / Date: Venlo, 10-10-2023

Signature: MN