CookWise

System Requirements Document

Group 03

PA2591 HT25 lp2
Requirements Engineering and Product Management

VERSION: 3
REVISION DATE: 10/01/2026

Project Supervisor: Bhuwan Paudel

Team Members:

Name	Role
Md Asif Iqbal Ahmed	Project manager
Letian Zhou	Customer coordinator
Runlin Long	Project member
Oghenero Jeremi Adjaino	Project member

Contents

1 Introduction

1.1 Purpose and scope

This document specifies the requirements for CookWise, a meal planning and grocery shopping optimization tool designed for the Swedish market. CookWise helps users plan their meals and optimize their grocery shopping by suggesting recipes based on current sales and promotions at major Swedish grocery stores such as ICA, Willys, and Coop.

What the System Will Do
CookWise will provide the following core capabilities:

- Suggest recipes based on items that are currently on sale at local grocery stores (ICA, Willys, Coop)
- Allow users to search for recipes by specific ingredients
- Allow users to filter recipes by cuisine type, dietary restrictions, budget, and cooking time
- Display detailed recipe information including ingredients, quantities, cooking instructions, and nutritional data
- Enable users to rate and provide feedback on recipes
- Generate shopping lists automatically from selected recipes
- Show estimated cost savings by comparing regular prices versus sale prices across different stores
- Provide a map view showing store locations, price comparisons, and navigation routes for the shopping list
- Enable users to create accounts and save favorite recipes for future reference

Benefits of the Product
CookWise delivers several key benefits to users:

- **Cost Savings:** Users can reduce grocery expenses by taking advantage of current sales and promotions when planning meals
- **Meal Variety:** Users discover new recipes and expand their cooking repertoire while staying within budget through intelligent sale based recipe recommendations
- **Informed Store Selection:** The system provides transparent price comparisons and store recommendations through a single platform, eliminating the need to manually check multiple store websites while enabling data driven shopping choices
- **Time Efficiency:** The system eliminates the need to manually browse multiple store flyers and match sales to recipes, significantly reducing weekly planning time
- **Reduced Food Waste:** By providing exact ingredient quantities and utilizing items on sale, the system helps minimize food waste at the household level

System Goals
The primary goals of CookWise are detailed in Section 1.4 as formal goal level requirements, covering cost effectiveness, meal variety through sale based recipe discovery, informed store selection, time efficiency, and waste reduction.

What the System Will NOT Do
To clarify the boundaries and scope of this project, CookWise will explicitly NOT include:

- Processing of actual purchases or payment transactions

- Real time inventory tracking or stock availability at physical stores
- Home delivery, delivery scheduling, meal kit assembly, or courier services
- Creation of fully custom or AI generated personalized recipes without human review
- Nutritional consultation or medical dietary advice

1.2 Definitions, acronyms, and abbreviations

This section provides definitions of all terms, acronyms, abbreviations, and domain specific terminology used throughout this document.

Requirement Identifiers
Requirements in this document are identified using the following prefixes:

- **DL**: Domain Level Requirement (e.g., DL1, DL2, DL3, DL4, DL5, DL6)
- **PR**: Product Requirement (e.g., PR1, PR2, PR3, ..., PR14)
- **DR**: Data Requirement (e.g., DR1, DR2, DR3, DR4)
- **QR**: Quality Requirement (e.g., QR1, QR2, QR3, QR4, QR5)

Terms and Definitions

Term	Definition
ACID Properties	A set of properties that guarantee reliable database transactions: Atomicity (transactions complete fully or not at all), Consistency (database remains in a valid state), Isolation (concurrent transactions don't interfere), and Durability (committed transactions persist permanently).
API	Application Programming Interface. A set of rules and protocols that allow different software applications to communicate with each other.
Consolidated Shopping List	A shopping list that combines ingredients from multiple selected recipes, aggregating quantities for items that appear in more than one recipe.
CookWise	The name of the meal planning and grocery shopping optimization system described in this document. The product is designed for the Swedish market.
GDPR	General Data Protection Regulation. A comprehensive data privacy and security law in the European Union (EU 2016/679) that mandates how the system must handle and protect personal data of users.
Recipe	A structured set of instructions for preparing a dish, including a list of ingredients with quantities, step by step cooking instructions, preparation time, and nutritional information.
Sale Item	A product that is currently offered at a discounted price or promotional price by a grocery retailer. Sale items are the basis for recipe suggestions in CookWise.
Shopping List	An automatically generated list of ingredients required for selected recipes, organized by product category and store, with quantities and estimated prices.
Stakeholder	Any individual, group, or organization that has an interest in or is affected by the CookWise system. Stakeholders include end users, grocery retailers, development team members, and regulatory bodies.

Term	Definition
Total Basket Price	The sum of all ingredient prices for a complete shopping list at a specific store, including all applicable promotions and discounts. This value is used to compare the cost-effectiveness of shopping at different stores for the same set of ingredients or equivalent items.
Web Scraping	An automated technique for extracting data from websites by parsing HTML content. CookWise uses web scraping to collect product prices, sale information, and store locations from retailer websites on a daily basis.

1.3 Overview

This document is structured to systematically present the requirements for the CookWise system, from initial stakeholder analysis through detailed specifications to a phased release plan. The organization of subsequent sections is as follows:

Section 2: Stakeholder Identification and Analysis identifies all individuals, groups, and organizations with an interest in the CookWise system. It analyzes their roles, influence levels, expectations, and potential impact on the project. This section establishes who the system must satisfy and whose needs must be balanced during development.

Section 3: Requirements Elicitation Techniques describes the specific methods used to gather requirements from stakeholders identified in Section 2. Five techniques are employed: brainstorming sessions, semi structured interviews, requirements workshops, questionnaires, and prototyping. Each technique's purpose, execution process, and key findings are documented.

Section 4: System Requirements forms the core of this document, detailing all requirements for CookWise. This section is subdivided into four parts:

- **Section 4.1: Domain Requirements** describes high level business rules and constraints inherent to the grocery shopping and meal planning domain that the system must respect.
- **Section 4.2: Product Requirements** specifies the concrete functional behaviors and capabilities the system must provide to users.
- **Section 4.3: Data Requirements** defines the data entities (such as users, recipes, products, stores), their attributes, and the relationships between them that the system must manage. An Entity Relationship Diagram (ERD) is included.
- **Section 4.4: Quality Requirements** elaborates on non functional requirements using the QUPER model. Five quality aspects are specified in detail: accuracy, reliability, performance, usability, and scalability.

Section 5: Requirements Prioritization presents the prioritization of all requirements from Section 4 using three complementary techniques: the 100 Dollar Test (ratio scale prioritization), ranking (ordinal scale prioritization), and Numerical Assignment (ordinal scale prioritization). The results guide development sequencing and resource allocation.

Section 6: Release Planning proposes a phased delivery strategy based on the prioritized requirements. Requirements are grouped into two major releases, each consisting of three two week sprints. Dependencies between requirements and risk factors are considered in the release plan.

Section 7: Policy and Regulation Requirements lists all legal, regulatory, and compliance requirements that CookWise must adhere to. This includes GDPR compliance, consumer protection laws, food information regulations, accessibility standards, and Swedish national legislation.

Section 8: References provides a comprehensive list of all cited documents, academic sources, standards, legal texts, and technical documentation referenced throughout this Software Requirements Document.

Section 9: Document Revision History tracks all versions of this document, including dates, version numbers, and descriptions of changes made in each revision.

Section 10: Appendices contains supplementary materials including the questionnaire, the PlantUML source code for diagrams, the AI-generated prototype prompt, and other supporting documentation.

This structure ensures a logical progression from understanding stakeholders and gathering their needs, to formally specifying requirements, prioritizing them, and planning their implementation in releases.

1.4 Goals of the Product (Goal Level Requirements)

This section describes the high level business goals and strategic objectives that CookWise aims to achieve.

Goal_1: Increase Grocery Cost Effectiveness
Users shall be able to reduce their grocery shopping expenses by leveraging sale items and promotions when planning meals.

Goal_2: Expand Meal Variety and Recipe Discovery
Users shall discover new recipes and expand their cooking repertoire while staying within their budget constraints through intelligent recipe recommendations based on current sales and promotions.

Goal_3: Optimize Store Selection
Users shall be able to identify the most cost effective grocery store for their shopping needs based on total basket price and proximity to their location.

Goal_4: Reduce Shopping Planning Time
Users shall spend significantly less time on meal planning and shopping list preparation compared to manual browsing of multiple store flyers and recipe websites.

Goal_5: Reduce Food Waste and Environmental Impact
Users shall reduce food waste at the household level through accurate ingredient quantities and strategic meal planning, contributing to environmental sustainability.

1.5 Context diagram for the system

The context diagram in Figure 1 illustrates the CookWise system and its interactions with external entities. The CookWise Main System (shown in blue) is the central component that processes user inputs and provides recipe recommendations, shopping lists, and store price comparisons.
External entities include:

- **Users**: Primary actors who provide preferences and location, receiving recipe suggestions and shopping lists.
- **Supermarket Websites**: Data sources (ICA, Willys, Coop) providing product prices, sale data, and store locations via web scraping.
- **Maps and Geolocation Service**: External service (Google Maps) providing store coordinates and navigation.
- **AI Recipe Generation Service**: External AI service (OpenAI/Claude) generating recipe suggestions based on sale items and user preferences.
- **External Authentication Service**: Third party authentication providers (Google and Apple) handling user login and account creation.

The diagram shows data flows between CookWise and external entities, with arrows indicating direction of information exchange.

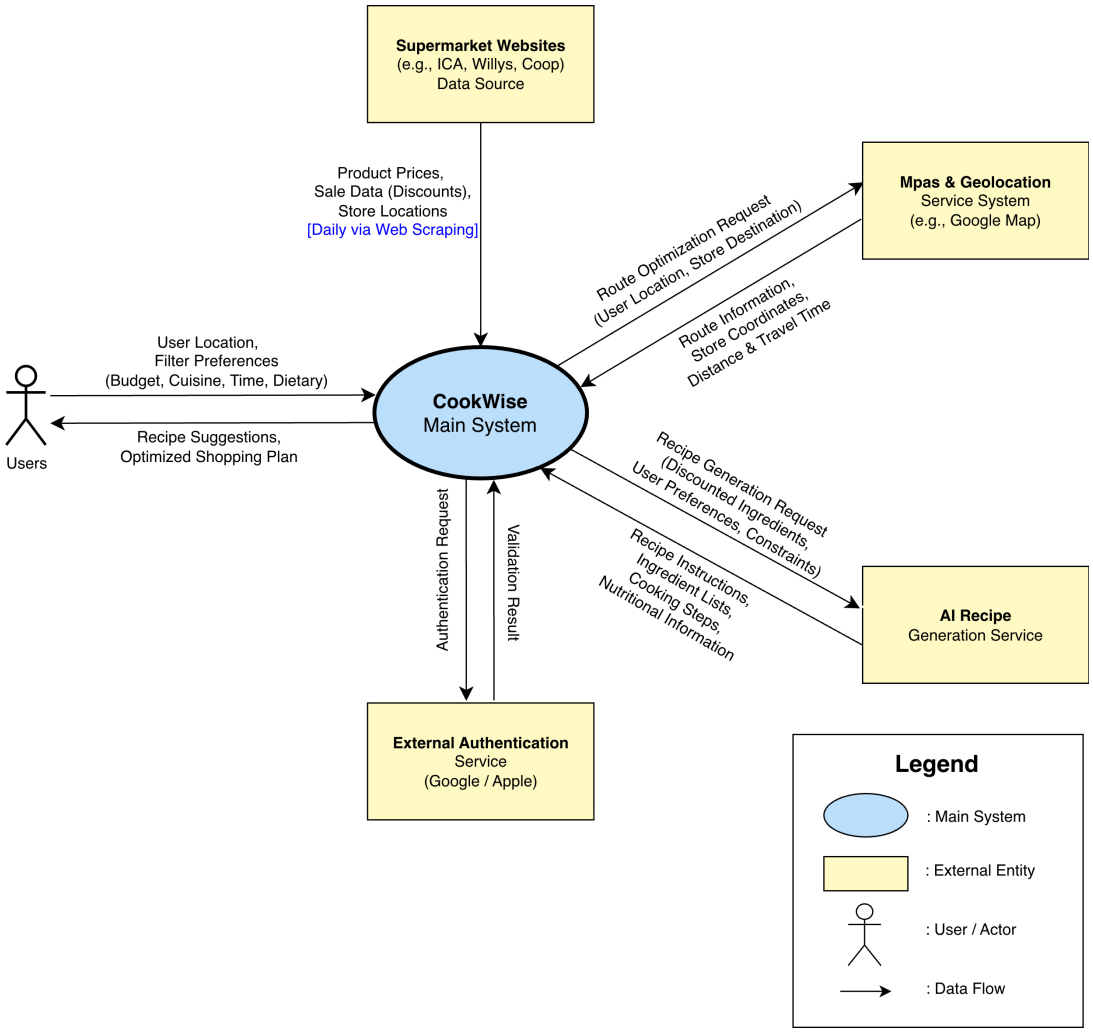

Figure 1: Context Diagram for CookWise System

2 Stakeholder Identification and Analysis

The following stakeholders have been identified for the system. For each stakeholder, we describe who they are, what goals they have for the system, why they would use or support it, and what concerns or risks they see. They are organized into four categories based on their relationship to the product.

2.1 Daily Users

These are the end users who will interact with the application regularly.

Stakeholder	Description	Priority
Budget Conscious Home Cooks	Families and individuals who do household grocery shopping and are highly price sensitive. They want the system to help them discover recipes based on current store sales, calculate their total savings, and reduce food waste by providing exact ingredient amounts. However, they worry that the system might recommend cheap items that do not match their preferences. They are concerned about the time needed to learn the system.	Critical
Time Constrained Users	Busy professionals, young workers, and people who recently moved to Sweden. These users have limited time for meal planning and grocery shopping. They need quick recipe ideas, efficient shopping routes, and help finding local stores. Their main worry is whether the app will be easy to use or add complexity to their schedules.	Critical
Culinary Enthusiasts	People who love cooking and enjoy trying new recipes and cuisines. They want the system to suggest creative recipes using seasonal ingredients or discounted specialty items. This helps them explore new dishes while keeping costs reasonable. They are concerned that the system might not provide enough variety beyond basic recipes.	High

2.2 Internal Team (Development and Management)

These are team members within our organization who are responsible for building and managing the product.

Stakeholder	Description	Priority
Product Manager	The person who owns the product vision and decides which features are most important. They gather user feedback, study the market, and ensure the product achieves business success. They must balance what different stakeholders want and make smart decisions about what to build and when. Their main concern is ensuring the product provides real value to users while remaining technically and financially feasible.	Critical
Development Team	The technical staff who design, build, test, and maintain the system. They need clear requirements that explain exactly what to build. They must ensure the system works reliably. Their concerns include handling technical challenges, connecting multiple external services such as AI for recipes and maps for locations, and ensuring the system performs well when many users access it simultaneously.	Critical
Marketing & Sales Team	The team who promotes the application, acquires new users, and builds partnerships with grocery stores. They need strong messages that clearly explain why users should choose this app over others. Their success depends on securing data partnerships with stores and achieving user adoption targets. They worry about market competition and the difficulty of convincing stores to collaborate.	High

2.3 Business Partners and Resource Providers

External organizations provide essential data, services, or resources for the system.

Stakeholder	Description	Priority
Grocery Store Chains (ICA, Willys, Coop)	Swedish supermarket chains that provide pricing and product data through web scraping. In the future, there will be official partnerships where stores directly share their data. They benefit from increased customer traffic and faster turnover of expiring or seasonal products. Their concerns include data sharing agreements, competitive implications, and alignment with their own digital strategies.	Critical
AI Service Provider	An external company that provides artificial intelligence services through an API to generate and recommend recipes. The quality, speed, and cost of their service directly affects how well the recipe feature works. Their concerns include API usage costs at scale and maintaining reliable service availability.	High
Map & Geolocation Service Provider	An external service such as Google Maps that provides location data, calculates distances between places, and suggests routes. This service is essential for helping users identify the closest and most convenient stores. Their concerns include usage costs, API rate limits, and compliance with their terms of service.	High

2.4 Regulatory Compliance

Note on Regulatory Authorities: Data protection authorities (such as GDPR enforcement agencies in the European Union) and other regulatory bodies are not explicitly listed as stakeholders in this document. Instead, their influence is captured through regulatory and compliance requirements embedded in Section 4 (Domain Level Requirements) and Section 7 (Policy and Regulation Requirements). Specifically, DL2 addresses GDPR compliance for user and partner data, ensuring the system meets all data protection obligations without treating regulatory bodies as direct stakeholders.

3 Requirements Elicitation Techniques

This section describes the elicitation techniques used to gather requirements for the CookWise system. We employed five different techniques to ensure comprehensive requirements coverage from multiple perspectives.

3.1 Brainstorming

Selection Rationale: Brainstorming was used to generate creative solution ideas through two strategic sessions: one before interviews to formulate questions, and one after interviews to develop solution ideas based on validated user pain points.

Execution:

- **Participants:** All four team members (Project Manager, Customer Coordinator, two Project Members)
- **Session 1 - Pre-Interview (90 min):** Generated 18 hypothetical pain points to inform interview questions around: "What problems might budget-conscious families face when planning meals and shopping for groceries?"
- **Session 2 - Post-Interview (90 min):** Generated 23 feature ideas based on five key user insights from interviews. Ideas were captured on whiteboard following classic brainstorming principles (no criticism during generation, encouraging wild ideas, building on others' suggestions)

Requirements Elicited:

Table 5: Requirements from Brainstorming

Requirement ID	Description
PR2	AI generated recipe recommendations using discounted ingredients
PR3	Recipe filtering (cuisine, dietary restrictions, cooking time, difficulty)
PR4	Automatic shopping list generation from selected recipes
PR8	Store recommendation based on total cost and location
PR9	Display detailed savings calculation
PR11	Show additional ingredients needed, prioritizing sale items
PR13	Store user preferences for items and recipes
QR1	Performance: Fast recipe generation and page loading
QR2	Usability: Easy first-time experience

Key Insight:
Comparing Session 1 hypotheses with interview findings revealed significant mismatches. The team initially assumed users wanted multi-store shopping optimization, but interviews showed strong preference for single-store convenience, validating the importance of user research before finalizing requirements.

3.2 Interviews

Selection Rationale: Semi-structured interviews were conducted early in the project to gain deep, qualitative understanding of users' grocery shopping journeys, pain points, and decision-making processes.

Execution:

- **Participants:** 10 individuals in Karlskrona (5 university students, 2 professionals, 3 household members)
- **Duration:** 5-10 minutes per interview
- **Method:** Semi-structured, in-person interviews focusing on recent shopping experiences
- **Topics Covered:** Meal planning habits, store selection criteria, price comparison behaviors, recipe discovery

Requirements Elicited:

Table 6: Requirements from Interviews

Requirement ID	Description
Domain Requirements	
DL3	Single store shopping constraint
DL4	Include all applicable promotions in basket price
Product Requirements	
PR2	AI recipe suggestions using discounted ingredients
PR8	Store recommendation based on total cost and distance
PR9	Display detailed savings breakdown
Quality Requirements	
QR1	Performance critical (time savings valued)
QR3	Reliability critical (price data accuracy)

Key Insights:
Interviews revealed two critical findings:

1. Users strongly prefer single-store shopping despite being price-conscious, validating DL3 as a fundamental constraint.
2. Time is valued more than money, with users spending 30+ minutes weekly manually matching sales to recipes, elevating QR1 (performance) to top priority.

3.3 Requirements Workshop

Selection Rationale: A requirements workshop was conducted to consolidate findings from brainstorming and interviews into structured, traceable requirements including domain models, refined specifications, and traceability documentation.

Execution:

- **Participants:** All four team members (Project Manager: Asif, Customer Coordinator: Letian Zhou, Project Members: Runlin, Oghenero)
- **Duration:** 4 hours in group room with large screen and whiteboard
- **Phase 1 - Domain Modeling (90 min):** Created Entity Relationship Diagram identifying core entities (User, Recipe, Ingredient, Store, Sale Item, Shopping List, Rating)
- **Phase 2 - Requirement Refinement (120 min):** Reviewed and refined all preliminary requirements for clarity, completeness, and traceability
- **Phase 3 - Traceability Mapping (30 min):** Linked requirements back to interview insights and business goals

Requirements Elicited:

Table 7: Requirements Elicited from Workshop

Req ID	Requirement Description
Data Requirements (from ERD creation)	
DR1	Core entity data models (User, Recipe, Ingredient, Rating, Store, Sale Item)
DR2	Relationship models (Recipe-Ingredient, Shopping List, Shopping List Item)
DR3	Data interchange formats (JSON for APIs, CSV/JSON for export, HTML parsing)
DR4	System states (user sessions, operational states, recipe generation, shopping list)
Refined Product Requirements	
PR1	External authentication (Google/Apple ID)
PR5	User ratings and feedback for recipes (added during workshop discussion)
PR6	Store AI generated recipes and their details
PR7	Display complete recipe details with step by step instructions
PR10	Search recipes by specific ingredients
PR12	Daily web scraping to gather and update item data from store websites
PR14	Display interactive map with store locations and navigation
PR15	Calculate and display average ratings for recipes
Domain Requirements (refined)	
DL1	Data sourced from store websites via web scraping (API partnerships future goal)
DL2	Compliance with data protection regulations (GDPR)
DL5	Rate limits and operational boundaries for external interfaces
DL6	Domain events triggered by user actions and system operations

Key Insights:

1. ERD creation revealed CookWise's value proposition depends on accurate, current sale price data, elevating QR3 (reliability/data accuracy) to top priority.

2. The workshop confirmed DL3 (single store shopping) as a fundamental domain constraint, reflected in the ERD by linking each shopping list to exactly one store.

3. Collaborative modeling revealed technical dependencies: PR1 (user accounts) must precede PR13 (preferences), and DL1 (sale data) plus DR1 (recipe database) must exist before PR2 (recipe suggestions) can function.

3.4 Questionnaire

Selection Rationale: Questionnaires were used to obtain statistical evidence (n=25) for key assumptions identified during interviews, using primarily closed questions (multiple choice, Likert scales) to quantify findings such as single store shopping preference, time burden, and interest in CookWise.

Execution:

- **Method:** Face-to-face questionnaire administered over a two week period
- **Locations:** BTH University campus and local gym in Karlskrona
- **Participants:** 25 individuals (12 university students, 3 professionals, 10 household members)
- **Questionnaire Design:** 15 questions (see Appendix A) organized into five sections: demographics, shopping behaviors, price sensitivity, meal planning habits, and interest in CookWise concept
- **Question Types:** Primarily closed questions (multiple choice, Likert scales 1-5) with limited open-ended questions
- **Duration:** 5-8 minutes per questionnaire

Requirements Validated:
The questionnaire provided quantitative validation for key requirements:

Table 8: Requirements Validated Through Questionnaire

Req ID	Validation Evidence
Domain Requirements	
DL3	Single store preference: 92% always shop at one store, 8% rarely visit multiple
Product Requirements	
PR2	High interest in sale based recipe suggestions: 77% rated 4 to 5 out of 5
PR3	Dietary filters important: 58% indicated dietary restrictions or preferences
PR9	Want to see savings details: 83% said transparency in savings calculation is important
Quality Requirements	
QR1	Time savings valued: 54% spend 30+ minutes weekly on meal planning
QR2	Usability important: Users want simple, intuitive interface

Key Findings:
Finding 1: Time Burden Confirmed
54% of respondents spend 30+ minutes weekly on meal planning and grocery list preparation (mean: 37 minutes/week). This validates the system's focus on reducing planning time by 60 to 80%.

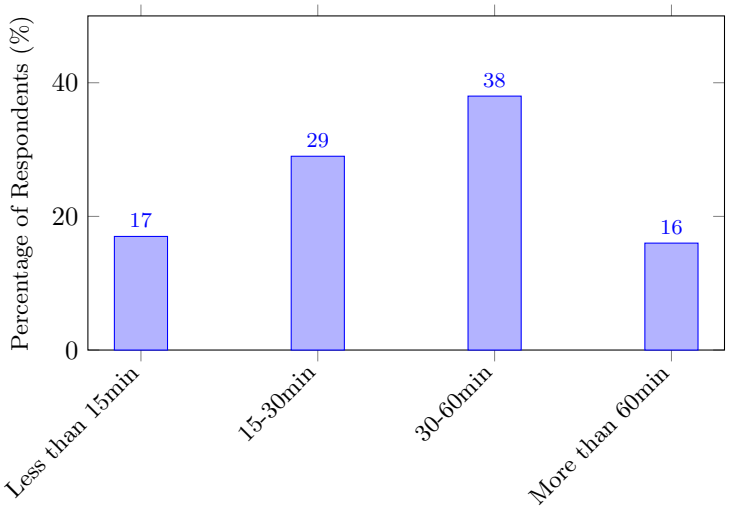

Figure 2: Time Spent Weekly on Meal Planning and Grocery List Preparation

Finding 2: Single Store Preference Validated

92% always shop at one store, 8% rarely visit multiple stores (only in rare cases), and 0% frequently shop at multiple stores. This strongly validates DL3 (single store constraint) as a fundamental requirement - virtually all users prefer single-store shopping, with no one regularly visiting multiple stores.

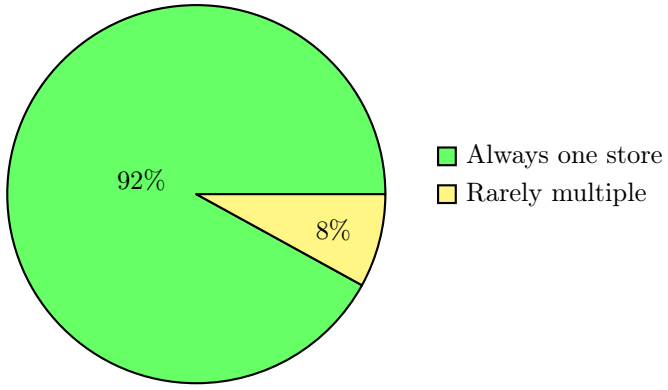

Figure 3: Shopping Behavior: Single Store vs Multiple Stores

Finding 3: High Interest in Sale Based Recipes

When asked about interest in an app that automatically suggests recipes based on current sales, 77% expressed high interest (rating 4 to 5 on a 5 point scale), with mean interest of 4.2/5.0. This validates PR2 (AI recipe suggestions based on sales) as a compelling core value proposition.

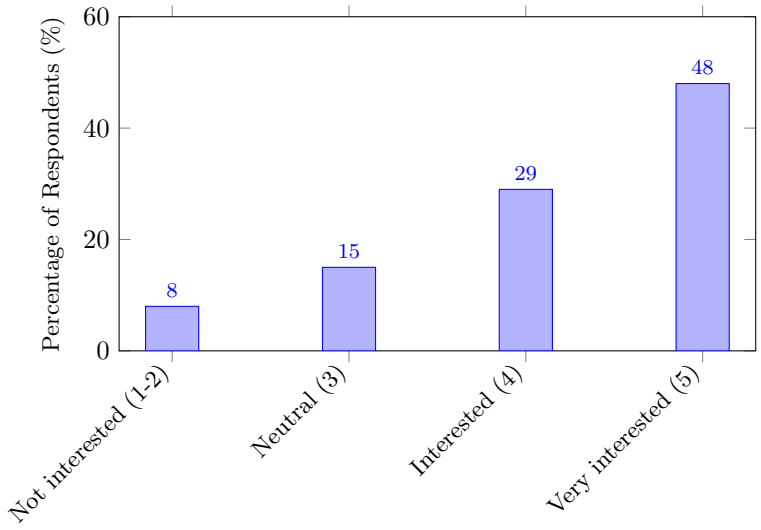

Figure 4: Interest Level in Sale Based Recipe Suggestions

Key Insights:
The questionnaire (n=25) quantitatively validated interview findings:

1. Time burden confirmed: 54% spend 30+ minutes weekly on meal planning.
2. Single store preference validated: 92% always shop at one store, confirming DL3 as fundamental constraint.
3. High interest in CookWise: 77% rated interest as 4 to 5 out of 5, demonstrating strong product market fit.

Limitations: Limited sample size (n=25), convenience sampling at university/gym locations, self-report bias, and geographic limitation to Karlskrona area.

The complete questionnaire with all 15 questions is provided in Appendix A.

3.5 Prototyping

Selection Rationale: Prototyping was used to visualize UI design, validate user workflows, and gather early feedback through two complementary approaches: (1) high-fidelity interactive Figma prototype for visual design testing, and (2) AI-generated functional prototype (Base64) for rapid design exploration and technical feasibility validation.

Execution:
Approach 1: High-Fidelity Interactive Prototype (Figma)
The team created a comprehensive high-fidelity interactive prototype using Figma, a collaborative interface design tool. The prototype was designed with a mobile-first approach, as most users are expected to access CookWise on smartphones. The Figma prototype covered the complete user journey across two main feature areas:

Recipe Features: The prototype included recipe browsing with multiple view options (all recipes, dessert recipes, low budget recipes), recipe filtering by cuisine type and dietary restrictions, and detailed recipe pages with ingredient lists, cooking instructions, and nutritional information. Figure 5 shows the recipe browsing interface with filtering capabilities, validating PR2 (AI recipe suggestions) and PR3 (recipe filtering).

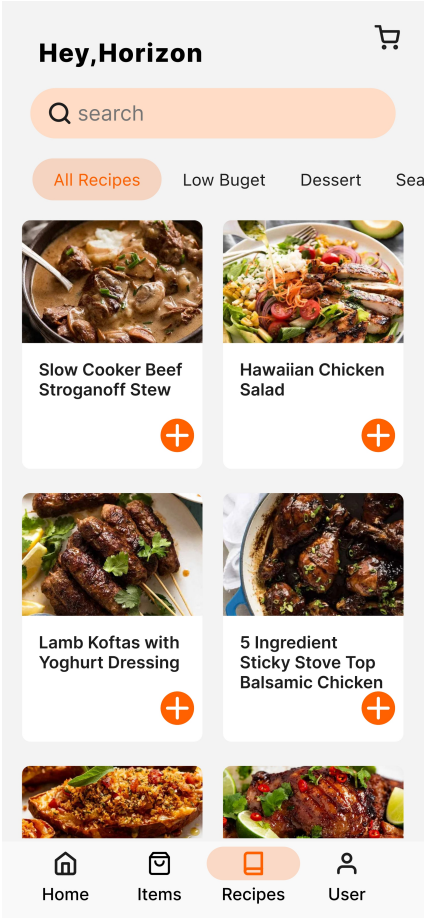

Figure 5: Recipe browsing with filtering options (Figma - PR2, PR3)

Items and Shopping Features: The prototype included product browsing pages organized by store (ICA, Willys, Coop) and by category (nuts, seafood, bakery, vegetables, fruits), shopping cart functionality with running totals and savings calculations, store location mapping for route optimization, and an "Everyday Low Price" section highlighting consistent deals. Figure 6 shows the home screen with store selection and promotional content, while Figure 7 demonstrates product browsing by store. Figure 8 displays the shopping cart with savings calculations.

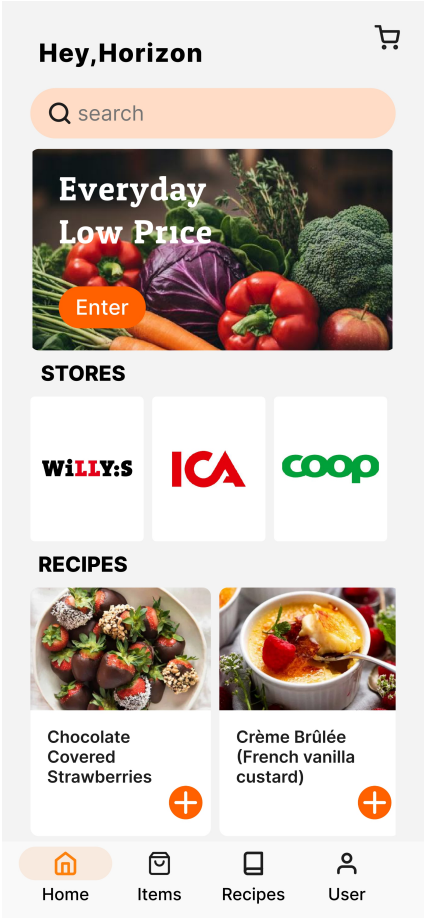

Figure 6: CookWise home screen with retailer selection (Figma)

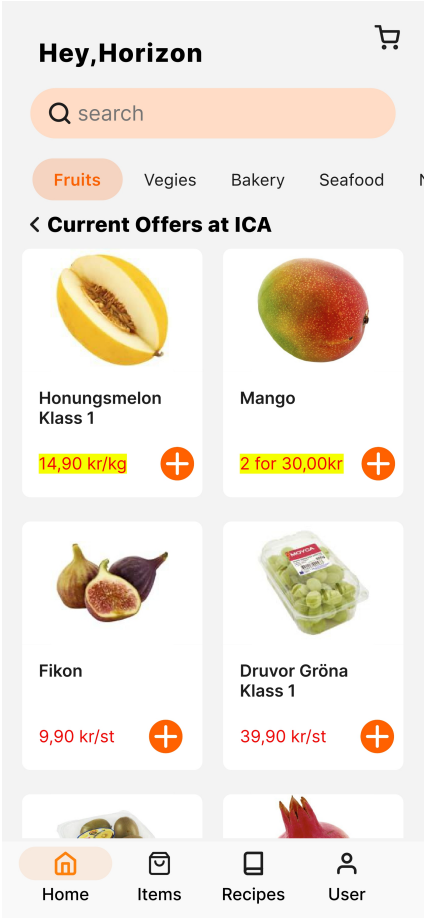

Figure 7: Product browsing with price display by store (Figma - PR8)

Figure 8: Shopping cart with savings calculation (Figma - PR4, PR9)

The Figma prototype underwent testing with two groups. First, the internal development team (four members) reviewed the prototype thoroughly, navigating through all interactive elements to identify usability issues and technical feasibility concerns. Second, the prototype was tested with a small group of seven potential users representing the target audience (the same individuals from the interview phase). Test participants were asked to complete specific tasks such as finding recipes based on current sales, adding items to shopping carts, comparing prices across stores, and evaluating the overall visual design. Testing sessions were conducted informally over one week, with participants encouraged to think aloud as they navigated the prototype.

Approach 2: AI-Generated Functional Prototype (Base64)

To complement the Figma prototype and explore alternative design approaches rapidly, the team created a functional prototype using AI-assisted development (Claude Code with Artifacts). The Base64 prototype was generated from a comprehensive prompt specifying all system requirements, Swedish market context, technical stack preferences, and UI/UX guidelines (see Appendix 10.4 for complete prompt).

The AI-generated prototype implemented working navigation, sample Swedish recipe data, realistic sale item displays, functional shopping list features, and responsive mobile design. Unlike the static Figma prototype, the Base64 version included functional CRUD operations and state management, allowing more realistic testing of user workflows. Figure 9 shows the AI-generated home screen design, while Figure 10 demonstrates the product browsing interface. Figure 11 displays the recipe detail page with ingredient optimization, and Figure 12 shows the automated shopping list with savings breakdown.

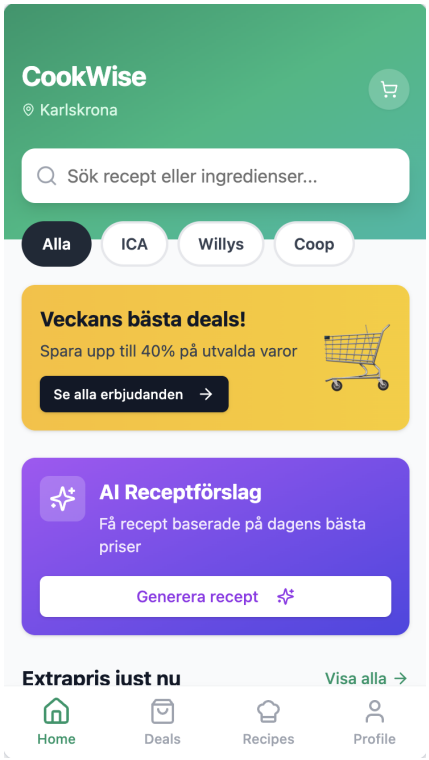

Figure 9: AI-generated home screen design with featured recipes and deals (Base64)

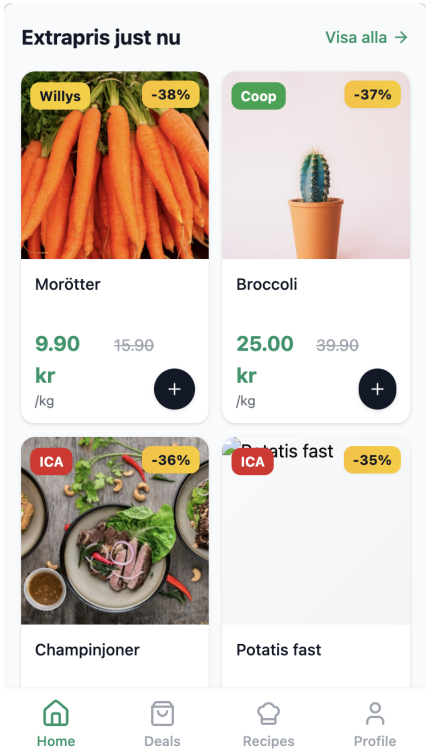

Figure 10: Product browsing interface with store filtering (Base64 - PR8)

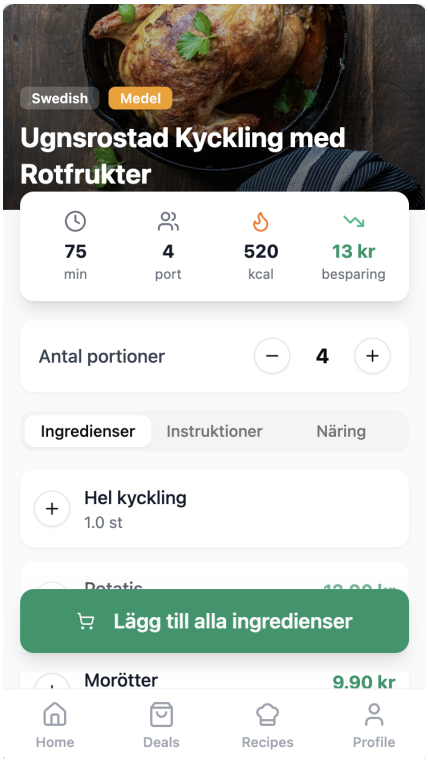

Figure 11: Recipe detail with ingredient list and price optimization (Base64 - PR2, PR4, PR7)

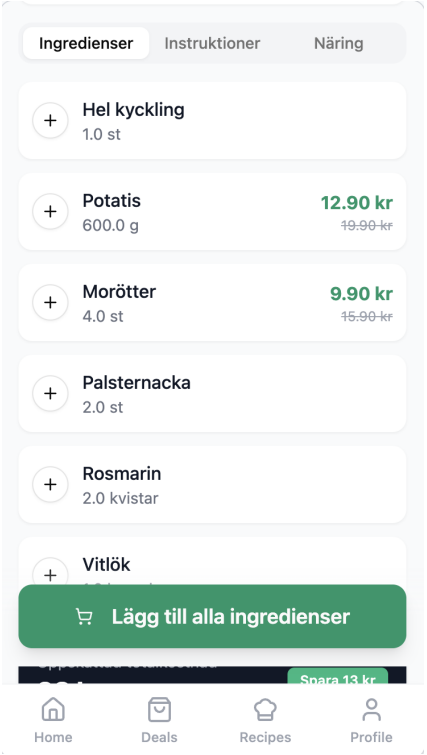

Figure 12: Automated shopping list with savings breakdown (Base64 - PR4, PR9)

The Base64 prototype was tested internally by the development team to evaluate technical feasibility, identify implementation challenges, and compare design approaches with the Figma version. The AI-generated code provided valuable insights into data structure design, component architecture, and potential technical constraints.

Requirements Elicited:
Table 9 summarizes the requirements validated or identified through both prototyping approaches.

Table 9: Requirements Validated and Identified Through Prototyping

Requirement	**Prototyping Outcome**
PR2	AI recipe suggestions based on sales validated as core value proposition through positive user response
PR3	Recipe filtering by cuisine and dietary restrictions found intuitive by users
PR4	Shopping list functionality demonstrated but users identified need for clearer recipe-to-list connection
PR7	Recipe detail pages with complete instructions and nutritional data validated
PR8	Store comparison display validated as highly valuable for decision-making
PR13	Users strongly expected ability to save favorite recipes and preferences
DR1	Prototype revealed need for data freshness indicator showing when prices were last updated
DR4	Users expected clear system state indicators and data validation
QR1	Image-heavy design raised performance concerns requiring optimization strategies
QR2	Mobile-first design with intuitive navigation validated positively
QR3	Users expressed high expectations for pricing accuracy, reinforcing reliability as critical

Key Insights:
The dual prototyping approach provided several valuable insights:

- **Price Comparison Drives Value**: The side-by-side price display on product detail pages was highlighted as the most valuable feature by users in both prototypes. The ability to see prices from all three retailers without visiting multiple apps directly addresses user pain points identified in interviews.
- **Visual Indicators Matter**: Users wanted more prominent visual indicators distinguishing sale items from regular-priced products. While sale prices were shown with yellow badges in both prototypes, additional icons or color coding would make sale items stand out more clearly in product listings.
- **Recipe-Shopping Integration Needs Clarity**: Users expected explicit visual affordances showing how selecting a recipe would automatically populate their shopping list with required ingredients. The connection between recipe browsing and shopping list generation (PR4) needs stronger visual design in both approaches.
- **Mobile-First Validated**: The mobile-optimized interface with bottom navigation and appropriately sized touch targets tested well with users in the Figma prototype. The design decision to prioritize smartphone usage over desktop was confirmed as aligned with user expectations.
- **Store Logo Trust**: Prominent display of recognizable retailer logos (Willys, ICA, Coop) on the home screen built immediate user confidence in the application's legitimacy and coverage.
- **AI Prototyping Accelerates Iteration**: The Base64 approach allowed rapid generation of functional prototypes with realistic data structures and working interactions in hours rather than days. This speed enabled the team to test multiple design variations and technical approaches efficiently.
- **Complementary Strengths**: The Figma prototype excelled at detailed visual design and user testing, while the Base64 prototype provided technical feasibility validation and rapid iteration. Using both approaches together provided more comprehensive insights than either method alone.

4 System Requirements

This section specifies the system requirements at different levels: domain, product, data, and quality. The requirements have been elicited through the techniques described in Section 3 and are documented using multiple specification techniques to ensure comprehensive coverage and clarity for all stakeholders.

Specification Techniques Used:
The following specification techniques have been employed to document the CookWise system requirements:

1. **Context Diagram** (Section 1.5): Visual representation of the system boundaries and external entities, showing how CookWise interacts with users, stores, and external services.
2. **Use Case Diagrams** (Section 4.2): Visual representation of user interactions with the system, showing both the overall system use cases and a detailed view of the core recipe browsing workflow.
3. **Feature Requirements** (Textual): Product-level requirements (PR1-PR15) are specified as textual "shall" statements describing specific system capabilities and behaviors.
4. **Data Model (Entity-Relationship Diagram)** (Section 4.4): Visual representation of all data entities, their attributes, and relationships using an ERD to ensure database design clarity.
5. **Data Dictionary** (Section 4.4): Textual description of all data requirements (DR1-DR4) covering core entities, relationships, data formats, and system configuration.
6. **Domain Rules** (Section 4.1): Textual specification of business rules and constraints (DL1-DL6) that define the problem domain and operational boundaries.
7. **Quality Requirements Specification** (Section 4.5): Structured specification of non-functional requirements (QR1-QR5) with measurable quality attributes using the QUPER model.

This combination of techniques ensures that requirements are communicated effectively to different stakeholders: visual models for system architects and developers, textual specifications for business stakeholders and testers, and structured quality metrics for project managers and quality assurance teams.

4.1 Domain Level Requirements

DL1: The system's core product and pricing data shall be sourced from grocery store websites through web scraping.

DL2: The system operations shall comply with relevant data protection regulations (e.g., GDPR) regarding user and partner data.

DL3: The system shall ensure all shopping lists contain only items available at a single physical store location (single-store shopping constraint).

DL4: The calculation of a store's total basket price shall include all applicable promotions and discounts for the user.

DL5: The system shall enforce rate limits and operational boundaries for external interfaces (web scraping limited to once daily per retailer, AI API limited to 100 requests per hour per user, geolocation API limited to 50 requests per user per day, and session timeout set to 30 minutes).

DL6: The system shall process domain events triggered by user actions (login, search, filter, select), external service responses (AI recipe generation, geolocation queries, authentication validation), and scheduled operations (daily web scraping at 02:00 local time).

4.2 Use Case Specification

The following use case diagrams illustrate the functional interactions between users and the CookWise system. These diagrams complement the product requirements by visualizing the user's journey through the system.

Overall System Use Cases

Figure 13 presents an overview of all primary use cases available to users in the CookWise system. The diagram shows that recipe browsing is dependent on store selection («include» relationship), and shopping list creation follows from recipe browsing. The use cases are ordered from top to bottom following the typical user flow: login, store selection, recipe browsing, and supporting features.

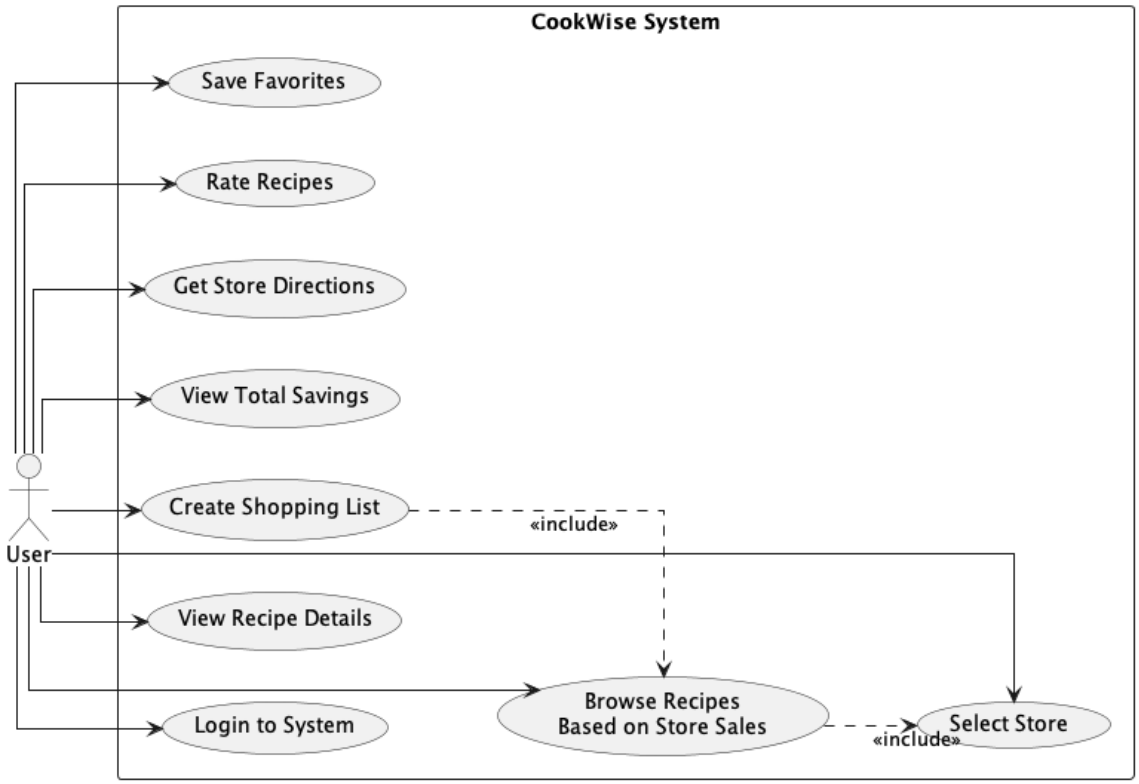

Figure 13: Overall System Use Case Diagram for CookWise

Detailed Use Case: Browse Recipes from Store Sale Items

Figure 14 provides a detailed view of the core use case "Browse Recipes from Store Sale Items." This use case represents the primary value proposition of CookWise: enabling users to discover recipes based on current sales at their selected grocery store. The diagram shows the main flow (blue) with required steps («include» relationships) and optional extensions (yellow) such as filtering, searching, and viewing details («extend» relationships). The PlantUML source code for both diagrams is provided in Appendix C.

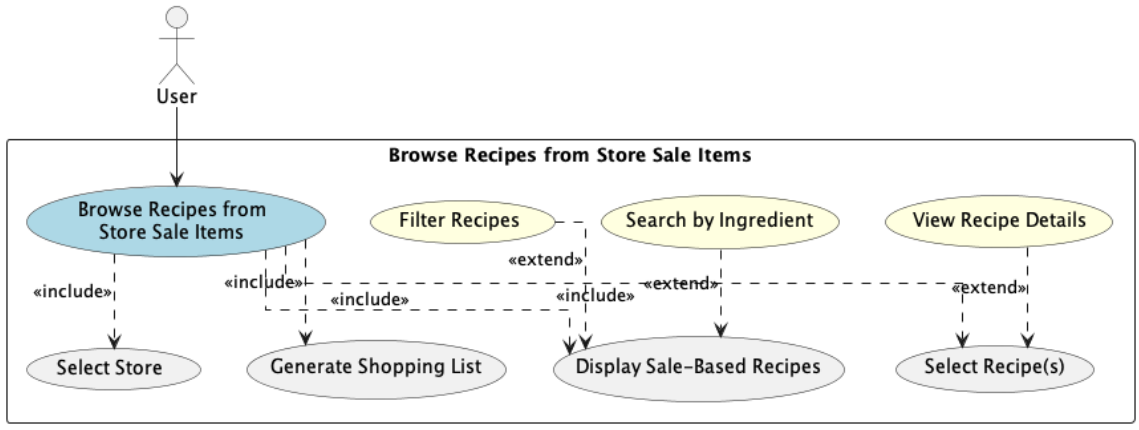

Figure 14: Detailed Use Case: Browse Recipes from Store Sale Items

4.3 Functional Product Level Requirements

PR1: The system shall authenticate users through external authentication services (e.g., Google/Apple ID).

PR2: The system shall invoke the AI Recipe Generation API with currently discounted ingredients, user dietary restrictions, and cuisine preferences to generate recipe suggestions.

PR2.1: The system shall display the top 5 AI-generated recipes ranked by cost savings.

PR3: The system shall filter recipe search results based on cuisine, dietary restrictions (e.g., vegan, gluten free), budget, cooking time, difficulty level, and user ratings.

PR4: The system shall automatically generate a consolidated shopping list when the user selects one or more recipes, aggregating ingredient quantities for duplicates and applying current sale prices where available.

PR5: The system shall record individual user ratings (1-5 scale) and text feedback submitted by users for recipes.

PR6: The system shall store recipes generated by AI and their details (e.g., id, ingredients, instructions, rating).

PR7: The system shall display recipe details including ingredients, cooking time, difficulty level, step by step cooking instructions, and nutritional data.

PR8: The system shall display store price comparisons and distances from user location for each store carrying the shopping list items.

PR9: The system shall display total savings amount and itemized price differences between regular and sale prices.

PR10: The system shall display recipes containing ingredients specified in the user's search query.

PR11: The system shall display all ingredients needed for selected recipes, highlighting which ingredients are currently on sale and listing sale items first in the display order.

PR12: The system shall gather and update product data (product name, regular price, sale price, discount percentage, sale validity dates, and product category) from grocery store websites through web scraping once per day at 02:00 local time (scheduled during low-traffic hours to minimize server load on retailer websites and ensure stores have updated their daily promotions).

PR13: The system shall store user preferences for ingredients and recipes.

PR14: The system shall provide store location information with "Get Directions" links that redirect users to their preferred external mapping application (Google Maps, Apple Maps, etc.) with the selected store address pre-populated.

PR15: The system shall calculate and display the average rating (aggregated from all user ratings per PR5) for each recipe, showing both the numerical average and the total number of ratings.

4.4 Data Requirements

Database Model Selection: The CookWise system will use a **relational database model** (SQL-based). This decision is based on the structured nature of the data, the need for complex relationships between entities (recipes, ingredients, stores, users), and the requirement for transactional integrity when managing shopping lists. A relational model ensures data consistency, supports complex joins for recipe recommendations, and provides ACID properties essential for data management operations.

DR1: Core Entity Data Models: The system shall implement the following core entities as specified in the Entity-Relationship Diagram (Figure 15):

- **User:** User ID, Email, Authentication Provider (Google/Apple), Geographic Location (Latitude and Longitude), and Dietary Restrictions
- **Recipe:** Recipe ID, Recipe Name, Cuisine Type, Cooking Time, Difficulty Level, Number of Servings, Step-by-step Instructions, and Image URL
- **Ingredient:** Ingredient ID, Ingredient Name, and Category
- **Rating:** Rating ID, User ID reference, Recipe ID reference, and Rating Score (1-5 scale)
- **Store:** Store ID, Store Name, and Geographic Coordinates (Latitude and Longitude)
- **Sale Item:** Sale ID, Store ID reference, Ingredient ID reference, Sale Price, and validity period (Valid From date and Valid To date) to track promotional pricing offered by grocery stores on specific ingredients

DR2: Relationship and Associative Entity Data Models: The system shall implement the following relationship tables to connect core entities:

- **Recipe-Ingredient:** Recipe ID reference, Ingredient ID reference, Quantity, and Unit to specify exactly how much of each ingredient is needed for a recipe
- **Shopping List:** List ID, User ID reference, Store ID reference to link each shopping list to a specific user and their chosen store, and Total Price to track the aggregate cost of all items
- **Shopping List Item:** List ID reference, Ingredient ID reference, Quantity to specify the amount needed, and Sale ID reference to track which specific sale promotion is applied to each item for accurate savings calculation

DR3: Data Interchange Formats and Communication Protocols: The system shall support the following data formats for external communication:

- **API Communications:** JSON format for all API communications with external services (AI recipe generation, geolocation, authentication)
- **Data Export:** CSV and JSON formats for user data export functionality
- **Web Scraping:** Structured HTML parsing for data extraction from grocery store websites
- **Encoding:** UTF-8 encoding for all API requests and responses following RESTful conventions

DR4: System States and Initial Data Configuration: The system shall maintain the following operational states and be initialized with seed data:

- **User Session States:** Anonymous, Authenticated, Active, Expired
- **System Operational States:** Idle, Scraping Data, Processing Requests, Maintenance Mode, Error State
- **Recipe Generation States:** Pending, In Progress, Completed, Failed
- **Shopping List States:** Draft, Finalized, Archived
- **Initial Seed Data:** Minimum of 100 recipes across various cuisines, complete store information for major ICA, Willys, and Coop locations in target regions, comprehensive ingredient database with at least 500 common ingredients, and default user preferences (Swedish language, metric measurements, no dietary restrictions, 30-minute default cooking time filter)

Entity Relation Diagram

The Entity Relation Diagram (ERD) below illustrates the data model for the CookWise system. The diagram shows the relationships between all entities and their attributes. The system uses a relational database model to ensure data integrity and support complex queries required for recipe recommendations and shopping list optimization.

Key Entities and Relationships:

The data model is organized around four core workflows:

1. Recipe Discovery and Rating: Users can browse recipes, view their ingredients through the RECIPE_INGREDIENT relationship, and provide ratings. This enables the system to recommend popular recipes and track user preferences.

2. Sale and Promotion Management: Stores offer sale items on specific ingredients, tracked through the SALE_ITEM entity. This connection between stores and ingredients enables the core functionality of suggesting recipes based on current promotions, with each sale item having a sale price and validity period.

3. Ingredient Management: The INGREDIENT entity serves as the central hub connecting recipes, shopping lists, and sales. Each ingredient has a name and category, enabling efficient search and organization across all system features.

4. Shopping List Management: Users create shopping lists that contain specific ingredients with quantities. Each shopping list is linked to a single store and tracks the total price. The SHOPPING_LIST_ITEM entity references which specific sales are applied to calculate accurate savings.

The model ensures single-store shopping (as per DL3) by linking each shopping list to one store through the STORE entity, while enabling price optimization by tracking which sale items are applied to each shopping list item.

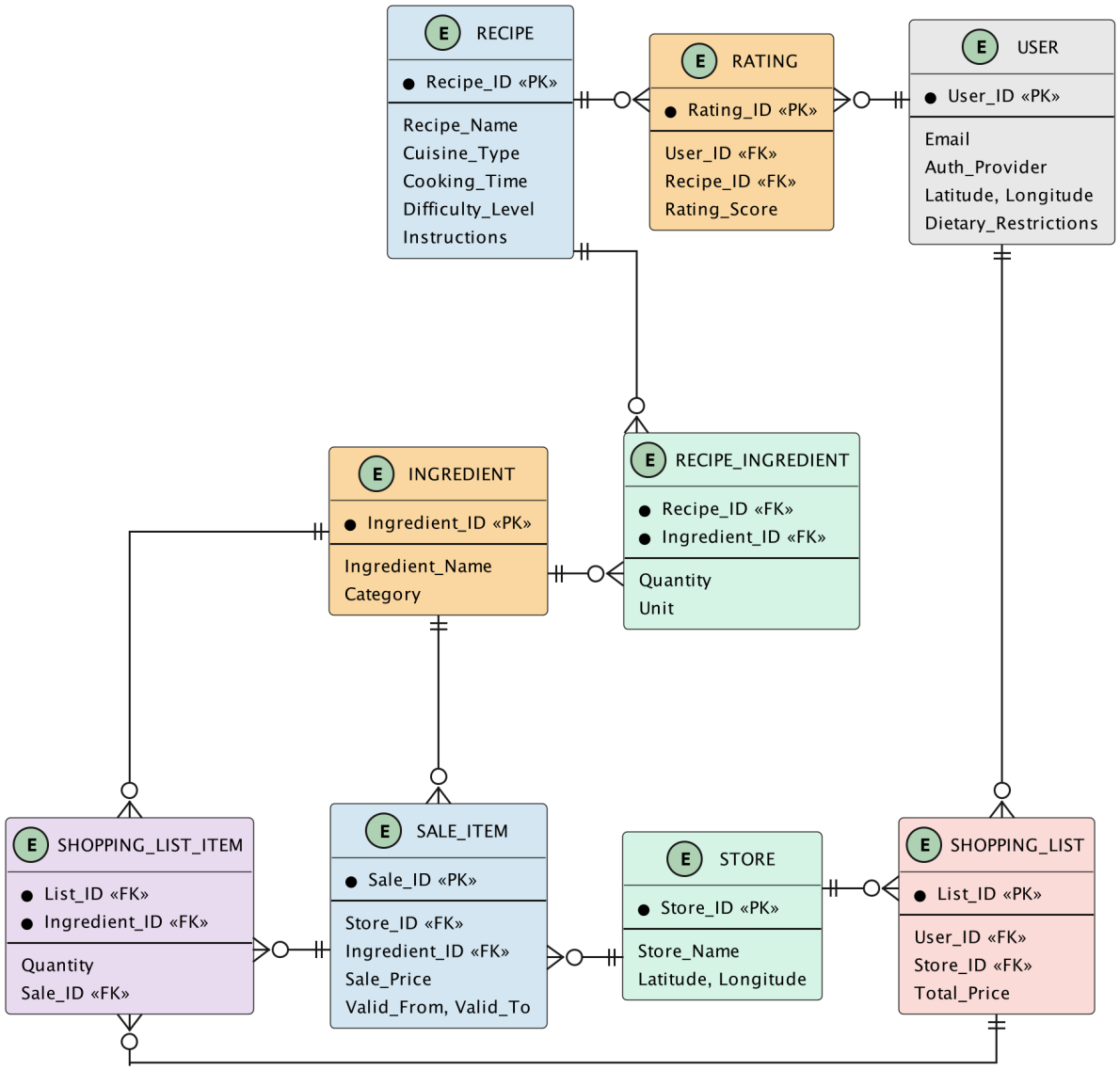

Figure 15: Entity Relation Diagram for CookWise System

The complete PlantUML source code for this diagram is provided in Appendix B.

Virtual Windows

Virtual windows illustrate how data will be presented to users through simplified screen mockups with realistic data. These windows demonstrate the key data flows and user interactions without detailed UI elements like buttons or menus.

Virtual Window 1: Recipe Detail View

This window shows how recipe data is presented to users, including ingredients with sale indicators, cooking details, and pricing information.

Swedish Meatballs with Cream Sauce

Cuisine: Swedish **Cooking Time:** 35 minutes **Difficulty:** Medium
Servings: 4 people **Average Rating:** 4.5/5 (24 ratings)

Ingredients:

- Ground beef, 500g **ON SALE** Regular: 89 kr → Sale: 59 kr
- Breadcrumbs, 100g
- Onion, 1 medium
- Heavy cream, 200ml **ON SALE** Regular: 28 kr → Sale: 19 kr
- Butter, 50g
- Beef broth, 250ml **ON SALE** Regular: 15 kr → Sale: 11 kr
- Salt and pepper to taste

Instructions:

1. Mix ground beef, breadcrumbs, chopped onion, salt and pepper. Form into small meatballs.
2. Fry meatballs in butter until golden brown on all sides (about 10 minutes).
3. Remove meatballs, add cream and beef broth to pan. Simmer for 5 minutes.
4. Return meatballs to sauce and cook for additional 10 minutes.

Your Savings:
Regular Total: 132 kr **Sale Total: 89 kr** **You save: 43 kr (33%)**

Table 10: Virtual Window 1: Recipe Detail with Sale-Based Ingredients

Virtual Window 2: Shopping List View

This window demonstrates how the shopping list aggregates ingredients from multiple selected recipes, applies sale prices, and calculates total savings.

My Shopping List

Selected Store: ICA Maxi Karlskrona
Recipes: Swedish Meatballs (4 servings), Pasta Carbonara (4 servings)

Dairy & Eggs
- Heavy cream, 400ml **SALE** 38 kr (Regular: 56 kr)
- Parmesan cheese, 100g 45 kr
- Eggs, 6 pieces **SALE** 25 kr (Regular: 32 kr)
- Butter, 100g 18 kr

Meat & Seafood
- Ground beef, 500g **SALE** 59 kr (Regular: 89 kr)
- Bacon, 200g **SALE** 35 kr (Regular: 48 kr)

Pantry & Dry Goods
- Spaghetti pasta, 500g 15 kr
- Breadcrumbs, 100g 12 kr
- Beef broth, 250ml **SALE** 11 kr (Regular: 15 kr)

Fresh Produce
- Onion, 2 medium 8 kr
- Garlic, 3 cloves 5 kr

Price Summary:
Regular Total Price: 338 kr
Your Total with Sales: 271 kr
Total Savings: 67 kr (20%)
Number of items on sale: 5 out of 11

Table 11: Virtual Window 2: Shopping List with Aggregated Ingredients and Pricing

Virtual Window 3: Store Price Comparison View

This window shows how the system compares total shopping list prices across different stores, helping users make informed decisions about where to shop.

Compare Stores for Your Shopping List

Shopping List: Swedish Meatballs + Pasta Carbonara (11 items)
Your Location: Karlskrona City Center

Store 1: ICA Maxi Karlskrona **BEST PRICE**
Distance from you: 2.3 km (7 min drive)
Items on sale: 5 out of 11
Regular total: 338 kr
Your total with sales: 271 kr
You save: 67 kr (20%)

Store 2: Willys Karlskrona
Distance from you: 1.8 km (6 min drive)
Items on sale: 3 out of 11
Regular total: 325 kr
Your total with sales: 289 kr
You save: 36 kr (11%)

Store 3: Coop Karlskrona
Distance from you: 3.1 km (9 min drive)
Items on sale: 4 out of 11
Regular total: 342 kr
Your total with sales: 298 kr
You save: 44 kr (13%)

Recommendation:
ICA Maxi offers the best price for your shopping list, saving you an additional 18 kr compared to Willys and 27 kr compared to Coop.

Table 12: Virtual Window 3: Store Price Comparison and Recommendation

These virtual windows demonstrate the core data presentation requirements for CookWise:

- **Recipe data integration:** Combining recipe details (DR1), ingredient information (DR1, DR2), and sale price data (DR1) to highlight cost-saving opportunities for users
- **Shopping list aggregation:** Consolidating ingredients from multiple recipes (DR2) with accurate quantity calculations and price totals including sale discounts
- **Multi-store comparison:** Presenting store information (DR1), location data, and total basket prices across different retailers to support informed shopping decisions
- **Visual sale indicators:** Clear marking of discounted items to immediately communicate savings opportunities to users
- **Savings calculations:** Transparent display of regular prices versus sale prices with percentage savings to demonstrate value proposition

4.5 Product Quality Requirements

This section defines the non-functional characteristics of CookWise through five quality requirements covering accuracy, reliability, performance, usability, and scalability. Each requirement is specified as a measurable "shall" statement with defined validation procedures.

4.5.1 Quality Requirements Summary

Table 13 presents all quality requirements for the CookWise system:

Table 13: CookWise Quality Requirements Summary

ID	Quality Aspect	Requirement	Target
QR1	Accuracy	The system shall achieve at least 95% accuracy in sale price data, measured as the percentage of sale prices that match actual retailer prices.	95%
QR2	Reliability	The system shall maintain at least 98% uptime during peak usage hours (17:00-20:00 local time), measured as the percentage of time the application is fully operational and accessible.	98%
QR3	Performance	The system shall generate recipe suggestions within 5 seconds for 95% of user requests, measured from the time a user submits search criteria to when results are displayed.	5 sec
QR4	Usability	The system shall enable new users to complete their first recipe selection within 8 minutes of account creation, measured from login to adding a recipe to their shopping list.	8 min
QR5	Scalability	The system shall support at least 300 concurrent users while maintaining acceptable performance (response time under 10 seconds), measured during peak usage hours.	300 users

QUPER Analysis Approach: Two quality requirements (QR1 and QR5) undergo detailed QUPER analysis due to their complexity and cost implications, while QR2, QR3, and QR4 have straightforward targets.

4.5.2 QR1: Accuracy - Price Data Correctness (QUPER Analysis)

Requirement: The system shall achieve at least 95% accuracy in sale price data, measured as the percentage of ingredient sale prices that match actual current sale prices published by retailers.

Rationale: Accuracy is critical for user trust and cost savings. Different accuracy levels require different technical approaches, making QUPER analysis valuable for understanding cost benefit tradeoffs.

FEATURE: Sale Price Information **ID:** QR1 **QUALITY REQUIREMENT:** Accuracy of sale prices
DEFINITION: The percentage of ingredient sale prices in the application that match the actual current sale prices published by retailers, measured as (number of correct prices / total number of sale prices) × 100.
REFERENCE LEVELS **PRODUCT:** eTilbudsavis **LEVEL:** 95% accuracy **PRODUCT:** Price comparison websites **LEVEL:** 85-90% accuracy **PRODUCT:** Current CookWise **LEVEL:** N/A (new system)
QUALITY BREAKPOINTS **UTILITY:** 90% accuracy **RATIONALE:** Below 90%, users lose trust in recommendations. One in ten incorrect prices creates too many negative experiences and undermines cost savings value proposition. **SATURATION:** 100% accuracy **RATIONALE:** Perfect accuracy impossible with web scraping due to timing delays, retailer website changes, and regional variations. Diminishing returns beyond 98-99%. **DIFFERENTIATION:** 98% accuracy **RATIONALE:** Accuracy of 98%+ establishes application as highly reliable and trustworthy. Can be promoted as "verified sale prices" for strong competitive advantage.
COST BARRIERS AND TECHNIQUES **Qref:** 92% accuracy - Basic web scraping with simple validation **Q1:** 96% accuracy **RATIONALE:** Advanced parsing algorithms, automated validation checks, error detection and correction systems. Requires sophisticated data validation logic for multiple retailer formats. Moving beyond 96% requires switching to or supplementing with API integrations instead of web scraping alone. **C1:** 2 weeks **Q2:** 99% accuracy **RATIONALE:** Multi-source validation combining web scraping with retailer API access (where available), cross-verification against multiple data points, and machine learning for anomaly detection. Requires partnerships with retailers and sophisticated data reconciliation. **C2:** 16 weeks
TARGETS **GOOD:** 95% accuracy **RATIONALE:** Reliable enough for confident purchasing decisions, matches best available competitor (eTilbudsavis), achievable with robust web scraping implementation and validation checks. **STRETCH:** 97% accuracy **RATIONALE:** Provides differentiation through advanced web scraping techniques, automated error correction, and multi-source data validation systems.
VALIDATION PROCEDURE Accuracy validated through weekly manual verification tests. Random sample of 100 sale items selected from database and manually checked against current retailer websites and physical store flyers. Percentage of matching prices calculated. Additionally, users can report incorrect prices through application for ongoing accuracy monitoring.

Table 14: QR1: Accuracy Quality Requirement with QUPER Analysis

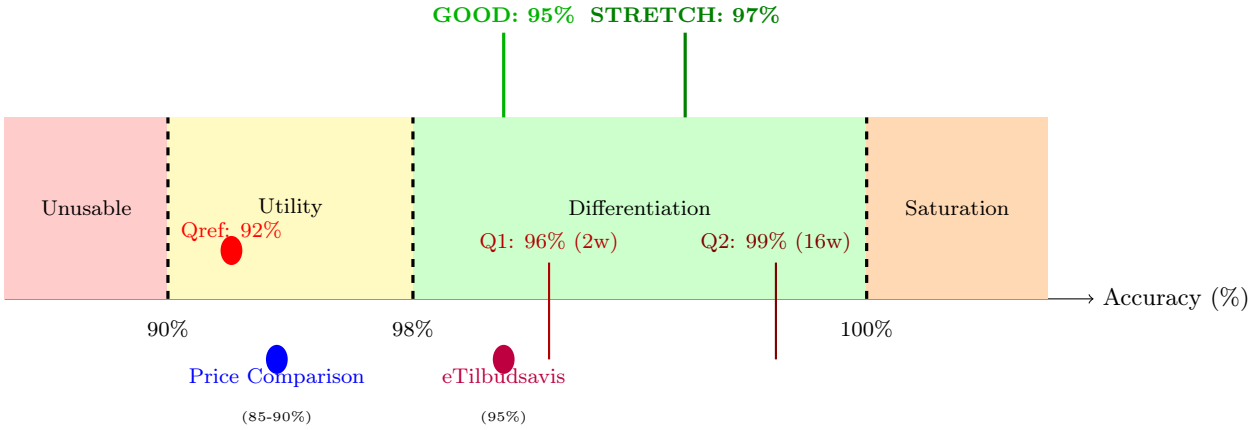

Figure 16: QUPER Roadmap for QR1: Accuracy (Price Data Correctness)

Saturation Interpretation: Beyond 98%, each additional percentage point requires exponentially increasing effort with minimal user perceived benefit.

4.5.3 QR5: Scalability - Concurrent User Capacity (QUPER Analysis)

Requirement: The system shall support at least 300 concurrent users while maintaining acceptable performance (response time under 10 seconds), measured during peak usage hours.

Rationale: Scalability requires QUPER analysis because infrastructure costs increase significantly at different capacity levels, and target selection involves substantial infrastructure investment decisions.

FEATURE: System Infrastructure **ID:** QR5 **QUALITY REQUIREMENT:** Concurrent user support
DEFINITION: The maximum number of users who can simultaneously use the application while maintaining acceptable performance (response time under 10 seconds), measured during peak usage hours.
REFERENCE LEVELS **PRODUCT:** Small recipe apps **LEVEL:** 100-500 concurrent users **PRODUCT:** eTilbudsavis scale **LEVEL:** 1000+ concurrent users **PRODUCT:** Current CookWise **LEVEL:** N/A (new system)
QUALITY BREAKPOINTS **UTILITY:** 50 concurrent users **RATIONALE:** Minimum viable capacity for initial launch in Swedish market. Below this, system cannot support even limited regional adoption and MVP launch becomes impractical. **SATURATION:** 5000 concurrent users **RATIONALE:** Capacity beyond 5000 exceeds realistic projections for initial Swedish market penetration. Infrastructure investment at this scale cannot be justified for early-stage product. **DIFFERENTIATION:** 500 concurrent users **RATIONALE:** Capacity of 500 supports significant market share in target regions (Karlskrona and surrounding areas) and enables positive growth trajectory.
COST BARRIERS AND INFRASTRUCTURE INVESTMENT **Qref:** 100 concurrent users - Basic single-server deployment **Q1:** 500 concurrent users **RATIONALE:** Load balancing, database connection pooling, query optimization, caching layer. Requires significant architecture improvements and database tuning. **Infrastructure cost implications:** Upgrading from shared hosting to dedicated server or basic cloud infrastructure (estimated additional €200-400/month operational costs). **C1:** 3 weeks **Q2:** 2000 concurrent users **RATIONALE:** Cloud infrastructure (AWS/Azure), CDN integration, distributed caching, auto-scaling. Requires migration to enterprise cloud platform and distributed system architecture. **Infrastructure cost implications:** Enterprise cloud tier with auto-scaling, load balancers, CDN (estimated additional €800-1500/month operational costs). **C2:** 10 weeks
TARGETS **GOOD:** 300 concurrent users **RATIONALE:** Realistic capacity for successful initial market launch, provides buffer for unexpected usage spikes, achievable with solid implementation and moderate infrastructure investment. **STRETCH:** 500 concurrent users **RATIONALE:** Provides differentiation and growth capacity. Reaching this level requires infrastructure investment (approximately €200-400/month additional operational costs) which must be evaluated against project budget and expected user growth trajectory.
VALIDATION PROCEDURE Scalability tested using load testing tools (Apache JMeter or Locust) simulating multiple concurrent users performing typical tasks (searching recipes, creating shopping lists). Tests gradually increase concurrent users until response time exceeds 10 seconds or error rate exceeds 1%. Maximum concurrent users before performance degradation recorded. Testing conducted in staging environment mirroring production configuration.

Table 15: QR5: Scalability Quality Requirement with QUPER Analysis

Note on Infrastructure Costs: Scaling from the Good level (300 users) to the Stretch level (500 users) involves both one time development costs and recurring monthly infrastructure expenses.

4.5.4 QR2, QR3, QR4: Additional Quality Requirements

The following quality requirements have straightforward targets based on industry standards and user expectations, and do not require detailed QUPER analysis:

QR2: Reliability - System Uptime

- **Requirement:** The system shall maintain at least 98% uptime during peak usage hours (17:00-20:00 local time).
- **Validation:** Uptime monitored using automated health checks every 60 seconds. Downtime incidents logged and analyzed. Monthly uptime percentage calculated as (total available minutes / total minutes in peak hours) × 100.
- **Rationale:** 98% uptime is acceptable for MVP launch while maintaining user trust.

QR3: Performance - Recipe Generation Response Time

- **Requirement:** The system shall generate recipe suggestions within 5 seconds for 95% of user requests.
- **Validation:** Response times logged for all recipe suggestion requests. 95th percentile response time calculated weekly from logs. Measured from user search submission to complete result display.
- **Rationale:** Users expect web applications to respond within 5 seconds for satisfactory engagement.

QR4: Usability - Time to First Recipe Selection

- **Requirement:** The system shall enable new users to complete their first recipe selection within 8 minutes of account creation.
- **Validation:** User session analytics track time from login to first recipe added to shopping list. Median completion time calculated monthly from new user cohorts. Users unable to complete task within 15 minutes flagged for usability investigation.
- **Rationale:** 8 minutes is a reasonable timeframe for new users to explore the interface and make their first selection.

5 Requirements Prioritization

This section applies three different prioritization techniques to the CookWise requirements identified in Section 4. The techniques used are the 100 Dollar Test (ratio scale), Ranking (ordinal scale), and Numerical Assignment (categorical scale). By comparing results across all three techniques, we establish a final consolidated priority ranking to guide the release planning process.

5.1 Prioritization Technique 1: 100 Dollar Test

Description of Technique

The 100 Dollar Test is a ratio scale technique where each stakeholder distributes 100 points across all requirements according to their perceived importance and value.

Application Process

Each team member independently allocated 100 points across requirements from Section 4. The allocations were then aggregated by calculating average points, considering factors such as business value, user impact, technical feasibility, and dependencies.

Results

Table 16 presents the results of the 100 Dollar Test, showing each requirement with its allocated points (averaged across team members) and sorted in descending order of priority.

Table 16: 100 Dollar Test Results for CookWise Requirements

ID	Requirement Description	Points
PR2	AI-generated recipe suggestions based on sales	12.0
DL1	Web scraping for product and pricing data	9.5
PR4	Automatic shopping list generation from recipes	7.5
PR1	User authentication via external services	6.0
QR3	Performance: Recipe generation response time	5.5
PR3	Recipe filtering by cuisine and dietary restrictions	4.5
QR4	Usability: Time to complete first recipe selection	4.0
DL3	Single-store shopping constraint	3.5
PR12	Daily web scraping for product data	3.5
DL2	GDPR compliance for user and partner data	3.0
QR1	Accuracy: Price data correctness	2.5
PR8	Store price comparisons and distances	2.5
PR13	Store user preferences for ingredients and recipes	2.0
PR7	Display recipe details with instructions	2.0
QR5	Scalability: Concurrent user capacity	1.5
PR9	Display total savings and price differences	1.5
PR5	Record user ratings and feedback	1.5
QR2	Reliability: System uptime during peak hours	1.0
PR6	Store AI-generated recipes and details	1.0
PR10	Display recipes by ingredient search	0.5
PR11	Display additional ingredients with sale items first	0.5
PR14	Interactive map with store locations and routes	0.5
PR15	Display average recipe ratings	0.5

PR2 (AI-generated recipe suggestions) received the highest allocation with 12 points, followed by DL1 (Web scraping) and PR4 (Automatic shopping list generation), reflecting the core value proposition and essential infrastructure.

5.2 Prioritization Technique 2: Ranking

Description of Technique

Ranking is an ordinal scale technique where requirements are arranged in strict order from highest to lowest priority, with each requirement receiving a unique rank number (1 = highest priority).

Application Process

After completing the 100 Dollar Test, the team reviewed point allocations and established a final ranking considering technical dependencies, implementation cost, and strategic importance for initial release.

Results

Table 17 presents the final ranking of CookWise requirements.

Table 17: Ranking Results for CookWise Requirements

Rank	ID	Requirement Description
1	PR1	User authentication via external services
2	DL1	Web scraping for product and pricing data
3	PR2	AI-generated recipe suggestions based on sales
4	PR4	Automatic shopping list generation from recipes
5	PR3	Recipe filtering by cuisine and dietary restrictions
6	PR12	Daily web scraping for product data
7	QR3	Performance: Recipe generation response time
8	DL3	Single-store shopping constraint
9	QR4	Usability: Time to complete first recipe selection
10	QR1	Accuracy: Price data correctness
11	DL2	GDPR compliance for user and partner data
12	PR8	Store price comparisons and distances
13	PR9	Display total savings and price differences
14	PR13	Store user preferences for ingredients and recipes
15	PR7	Display recipe details with instructions
16	QR5	Scalability: Concurrent user capacity
17	DL4	Promotions and discounts in basket price
18	QR2	Reliability: System uptime during peak hours
19	PR5	Record user ratings and feedback
20	PR6	Store AI-generated recipes and details
21	PR10	Display recipes by ingredient search
22	PR11	Display additional ingredients with sale items first
23	PR14	Interactive map with store locations and routes
24	PR15	Display average recipe ratings
25	DL5	Rate limits for external interfaces
26	DL6	Domain events processing

The ranking shows some differences from the 100 Dollar Test results. Notably, PR1 (User authentication) is ranked first because it is a foundational requirement that must be implemented before most other features can function. Although it received fewer points in the 100 Dollar Test, its technical dependency elevated its rank. Similarly, DL1 (Web scraping) ranks second as it provides the essential data foundation for the entire application.

5.3 Prioritization Technique 3: Numerical Assignment

Description of Technique

Numerical Assignment categorizes requirements into three priority groups: Critical (essential for system to function and deliver core value), Standard (important enhancements but not necessary for initial launch), and Optional (desirable features with limited impact if omitted).

Application Process

After completing the 100 Dollar Test and Ranking exercises, each team member independently categorized requirements from Section 4 into three priority groups based on technical foundation, core value proposition support, user experience enhancement, and deferability. The team then reached consensus through structured discussion.

Results

Table 18 presents the Numerical Assignment categorization of CookWise requirements.

Table 18: Numerical Assignment Categorization for CookWise Requirements

ID	Requirement Description	Category
CRITICAL		
PR1	User authentication via external services	Critical
DL1	Web scraping for product and pricing data	Critical
PR2	AI-generated recipe suggestions based on sales	Critical
PR3	Recipe filtering by cuisine and dietary restrictions	Critical
PR4	Automatic shopping list generation from recipes	Critical
PR12	Daily web scraping for product data	Critical
QR3	Performance: Recipe generation response time	Critical
QR1	Accuracy: Price data correctness	Critical
STANDARD		
DL2	GDPR compliance for user and partner data	Standard
DL3	Single-store shopping constraint	Standard
PR7	Display recipe details with instructions	Standard
PR8	Store price comparisons and distances	Standard
PR9	Display total savings and price differences	Standard
PR13	Store user preferences for ingredients and recipes	Standard
QR4	Usability: Time to complete first recipe selection	Standard
QR5	Scalability: Concurrent user capacity	Standard
OPTIONAL		
DL4	Promotions and discounts in basket price	Optional
DL5	Rate limits for external interfaces	Optional
DL6	Domain events processing	Optional
PR5	Record user ratings and feedback	Optional
PR6	Store AI-generated recipes and details	Optional
PR10	Display recipes by ingredient search	Optional
PR11	Display additional ingredients with sale items first	Optional
PR14	Interactive map with store locations and routes	Optional
PR15	Display average recipe ratings	Optional
QR2	Reliability: System uptime during peak hours	Optional

The Numerical Assignment categorization identified 8 Critical requirements (31% of total), 8 Standard requirements (31%), and 10 Optional requirements (38%). This distribution shows a reasonable balance, with a focused set of Critical requirements deemed absolutely essential for the initial release, an equally sized group of Standard enhancements, and a larger set of Optional features that can be deferred to future releases.

Note that PR13 (Store user preferences) was classified as Standard rather than Critical, as while it enhances personalization significantly, the core functionality can operate without it for the initial MVP release.

5.4 Comparison and Discussion

Comparison Across Techniques

The three prioritization techniques provided complementary perspectives on requirement priorities. Table 19 presents a consolidated view of the top requirements across all three techniques.

Table 19: Comparison of Top Requirements Across Three Techniques

Requirement ID	100 Dollar Rank	Ranking Pos.	Numerical Assign.
PR2	1	3	Critical
DL1	2	2	Critical
PR4	3	4	Critical
PR1	4	1	Critical
QR3	5	7	Critical
PR3	6	5	Critical
QR4	7	9	Standard
DL3	8	8	Standard
PR12	9	6	Critical
DL2	10	11	Standard

Key Observations

Several important observations emerge from comparing the three techniques:

1. **Strong Agreement on Core Requirements:** All three techniques consistently identified PR1, DL1, PR2, PR3, PR4, PR12, and QR3 as highest priority, indicating clear consensus on core functionality.
2. **Impact of Technical Dependencies:** Ranking elevated PR1 (User authentication) to top position despite fewer points in 100 Dollar Test, recognizing it as a foundational requirement that must be implemented first.
3. **Categorization vs. Continuous Scales:** Numerical Assignment grouped 8 requirements as Critical, while 100 Dollar Test and Ranking provided more granular differentiation within the high priority group.
4. **Quality Requirements:** QR3 (Performance) and QR1 (Accuracy) were consistently prioritized highly, while QR4, QR5, and QR2 were ranked lower, suggesting they can be addressed incrementally after launch.
5. **Data Foundation Requirements:** DL1 and PR12 were both prioritized highly, reflecting that accurate, up to date pricing data is fundamental to CookWise's value proposition.

Addressing Disagreements

The primary disagreement concerned PR1 versus PR2 ordering. The 100 Dollar Test ranked PR2 first based on user value, while Ranking placed PR1 first based on technical necessity. Both are equally critical but serve different purposes: PR1 must be implemented first for technical reasons, while PR2 represents the primary value proposition. Both are classified as Critical in Numerical Assignment, confirming their equal importance.

Final Consolidated Priority

Based on the analysis of all three techniques, the team established a final consolidated priority for the CookWise requirements:

Must Have - Release 1.0 MVP (Critical for launch):

- PR1: User authentication via external services
- DL1: Web scraping for product and pricing data
- PR2: AI-generated recipe suggestions based on sales
- PR3: Recipe filtering by cuisine and dietary restrictions
- PR4: Automatic shopping list generation from recipes
- PR12: Daily web scraping for product data
- QR3: Performance - Recipe generation response time

- QR1: Accuracy - Price data correctness

Should Have - Release 1.0 if resources permit or Release 2.0 (Standard):

- DL2: GDPR compliance for user and partner data
- DL3: Single-store shopping constraint
- PR7: Display recipe details with instructions
- PR8: Store price comparisons and distances
- PR9: Display total savings and price differences
- PR13: Store user preferences for ingredients and recipes
- QR4: Usability - Time to complete first recipe selection
- QR5: Scalability - Concurrent user capacity

Could Have - Future releases (Optional):

- DL4: Promotions and discounts in basket price calculation
- DL5: Rate limits for external interfaces
- DL6: Domain events processing
- PR5: Record user ratings and feedback
- PR6: Store AI-generated recipes and details
- PR10: Display recipes by ingredient search
- PR11: Display additional ingredients with sale items first
- PR14: Interactive map with store locations and routes
- PR15: Display average recipe ratings
- QR2: Reliability - System uptime during peak hours

Implications for Release Planning

The 8 Must Have requirements form the minimum viable product, while the 8 Should Have requirements should be included in Release 1.0 if resources permit, or deferred to Release 2.0. The 10 Could Have requirements can be deferred to subsequent releases. This prioritized list will guide the release planning process described in Section 6.

6 Release Planning

The release planning uses Sprint Based Agile Release Planning, organizing prioritized requirements into release backlogs and development sprints for iterative delivery.

6.1 Release Planning Technique: Sprint Based Agile Planning

6.1.1 Description

Sprint Based Agile Release Planning is a technique commonly used in agile software development methodologies such as Scrum. The process involves:

1. **Product Backlog**: A prioritized list of all requirements for the product.
2. **Release Backlog**: A subset of the product backlog selected for a specific release, based on business value, dependencies, and capacity.
3. **Sprint Planning**: Requirements from the release backlog are allocated to fixed length development sprints (typically 1 to 4 weeks).
4. **Sprint Execution**: The development team implements the requirements assigned to each sprint.

Each sprint produces a potentially shippable product increment, allowing for early validation and feedback.

6.1.2 Planning Approach and Assumptions

The release planning for CookWise follows Sprint-Based Agile principles with the following approach:

Sprint Planning Process:

1. **Backlog Refinement**: Prior to each sprint, the team reviews the prioritized product backlog (from Section 5), breaks down high-level requirements into implementable tasks, and estimates effort using story points.
2. **Velocity Estimation**: Team velocity is estimated at 12-16 story points per sprint based on a 4-person team. Initial sprints may have lower velocity (8-10 points) as the team establishes development workflows.
3. **Requirement Selection**: Requirements are selected for each sprint based on: (a) business priority from 100-dollar test, (b) technical dependencies, (c) team capacity, and (d) balanced mix of domain, product, data, and quality requirements.
4. **Sprint Review and Adaptation**: At the end of each sprint, delivered functionality is reviewed. If velocity differs from estimates, subsequent sprint plans are adjusted accordingly.

Planning Assumptions:

- **Sprint Duration**: Each sprint is 2 weeks long (10 working days).
- **Sprints per Release**: Each release consists of 3 sprints (6 weeks total per release).
- **Team Composition**: Development team consists of 4 members (2 full-stack developers, 1 backend specialist, 1 frontend specialist).
- **Team Capacity**: The team can implement 3 to 4 requirements per sprint, depending on requirement complexity and technical dependencies. Complex requirements (e.g., web scraping, AI integration) count as 2-3 simpler requirements in terms of effort.
- **Velocity Tracking**: Story points assigned to each requirement type: Domain requirements (2-3 points), Product requirements (2-4 points), Data requirements (1-2 points), Quality requirements (3-5 points for implementation and testing).
- **Release Frequency**: Release 1.0 is the minimum viable product (MVP) targeted for initial market launch. Release 2.0 follows 6 weeks after Release 1.0 launch, adding enhanced functionality.
- **Technical Dependencies**: Requirements with dependencies must be implemented in the correct order (foundational requirements before dependent features). Dependency analysis performed during backlog refinement ensures proper sequencing.
- **Testing Integration**: Each sprint includes unit testing, integration testing, and quality validation (for QRs). No separate testing phase - testing is integrated throughout each sprint.

6.2 Release 1.0: Minimum Viable Product

6.2.1 Release Goals

Release 1.0 delivers the core value proposition: sale based recipe suggestions and automatic shopping list generation. This release implements the 9 Must Have requirements identified in Section 5, establishing foundational infrastructure, implementing core functionality, and ensuring acceptable performance and data accuracy.

6.2.2 Release 1.0 Backlog

Table 20 shows the requirements selected for Release 1.0, organized by priority.

Table 20: Release 1.0 Backlog

ID	Requirement	Type
DL1	Web scraping for product and pricing data	Domain
DL2	GDPR compliance for user and partner data	Domain
DR1	Core entity data models	Data
DR2	Relationship and associative entity data models	Data
PR1	Sale based recipe suggestions	Product
PR2	Recipe search and filtering	Product
PR3	Automatic grocery shopping list generation	Product
QR1	Performance: Recipe generation response time	Quality
QR3	Accuracy: Price data correctness	Quality

6.2.3 Sprint Allocation for Release 1.0

The 9 requirements in Release 1.0 are distributed across 3 sprints, considering technical dependencies and implementation effort. Table 21 presents the sprint allocation.

Table 21: Release 1.0 Sprint Allocation

Sprint	Requirements	Focus
Sprint 1 (Weeks 1-2)	DL1: Web scraping DL2: GDPR compliance DR2: Relationship data models	Foundation
Sprint 2 (Weeks 3-4)	DR1: Core entity data models PR2: Recipe search and filtering QR3: Accuracy validation	Core Data
Sprint 3 (Weeks 5-6)	PR1: Sale based recipe suggestions PR3: Shopping list generation QR1: Performance optimization	Core Features

Sprint 1 (Foundation): Establishes foundational infrastructure with DL1 (Web scraping), DL2 (GDPR compliance), and DR2 (Relationship data models).

Sprint 2 (Core Data): Implements database structure with DR1 (Core entity data models), PR2 (Recipe search and filtering), and QR3 (Accuracy validation).

Sprint 3 (Core Features): Delivers primary value proposition with PR1 (Sale based recipe suggestions), PR3 (Shopping list generation), and QR1 (Performance optimization).

6.2.4 Release 1.0 Dependencies

Figure 17 illustrates the key dependencies between requirements in Release 1.0. Arrows indicate that one requirement depends on another being implemented first.

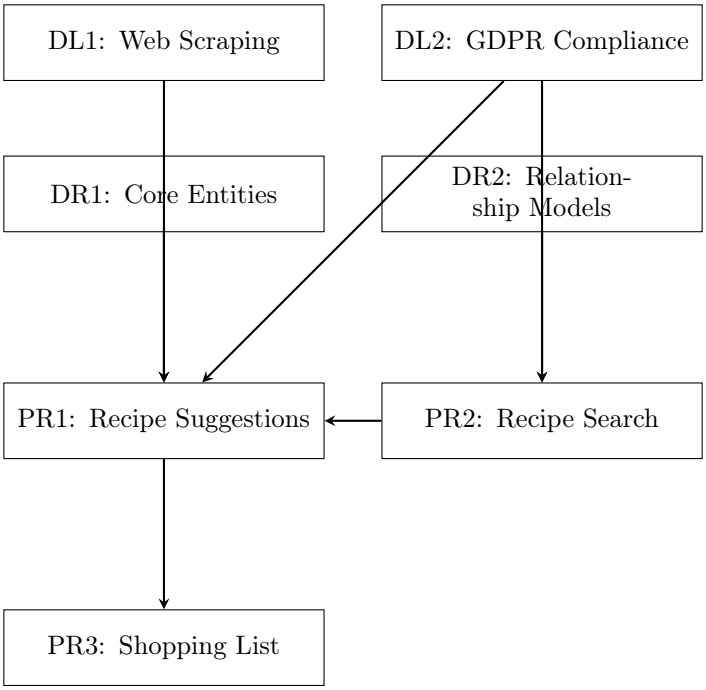

Figure 17: Release 1.0 Requirement Dependencies

6.3 Release 2.0: Enhanced User Experience

6.3.1 Release Goals

Release 2.0 enhances user experience through personalization, usability improvements, and expanded data features. This release implements 10 Should Have requirements from Section 5, differentiating CookWise from basic recipe apps and increasing user engagement.

6.3.2 Release 2.0 Backlog

Table 22 shows the requirements selected for Release 2.0.

Table 22: Release 2.0 Backlog

ID	Requirement	Type
DL3	Shopping list and meal planning system	Domain
DR3	User profile and preference data	Data
DR4	Recipe nutritional information	Data
DR5	Retailer store location data	Data
PR4	Personalized recipe recommendations	Product
PR5	Dietary preference and restriction filters	Product
PR6	Price comparison across multiple retailers	Product
PR7	Shopping list modification and sharing	Product
QR2	Usability: First recipe selection time	Quality
QR4	Scalability: Concurrent user capacity	Quality

6.3.3 Sprint Allocation for Release 2.0

The 10 requirements in Release 2.0 are distributed across 3 sprints. Table 23 presents the sprint allocation.

Sprint 4 (Personalization): Introduces personalization with DR3 (User profiles), PR5 (Dietary filters), and DR4 (Nutritional information).

Sprint 5 (Enhanced Features): Implements advanced features with PR4 (Personalized recommendations), DL3 (Meal planning), PR7 (List modification and sharing), and QR2 (Usability improvements).

Table 23: Release 2.0 Sprint Allocation

Sprint	Requirements	Focus
Sprint 4 (Weeks 7-8)	DR3: User profiles and preferences PR5: Dietary filters DR4: Nutritional information	Personalization
Sprint 5 (Weeks 9-10)	PR4: Personalized recommendations DL3: Meal planning system PR7: List modification and sharing QR2: Usability improvements	Enhanced Features
Sprint 6 (Weeks 11-12)	DR5: Store location data PR6: Price comparison QR4: Scalability improvements	Data Expansion

Sprint 6 (Data Expansion): Expands available data with DR5 (Store location data), PR6 (Price comparison), and QR4 (Scalability improvements).

6.4 Product Backlog and Future Releases

After Release 2.0, the remaining 14 requirements from Section 5 form the backlog for future releases. Table 24 categorizes these requirements.

Table 24: Future Release Backlog

Priority	Requirements	Count
Could Have (Release 3.0)	DL4, DR6, DR7, DR8, PR8, PR9, PR12, QR5	8
Won't Have (Release 4.0+)	DR9, DR10, PR10, PR11, PR13, PR14	6

The Could Have requirements would be considered for Release 3.0, while Won't Have requirements are deferred to Release 4.0 or later based on market feedback and available resources.

6.5 Release Timeline and Milestones

Figure 18 presents the overall release timeline for CookWise, showing the progression from initial development through Release 2.0.

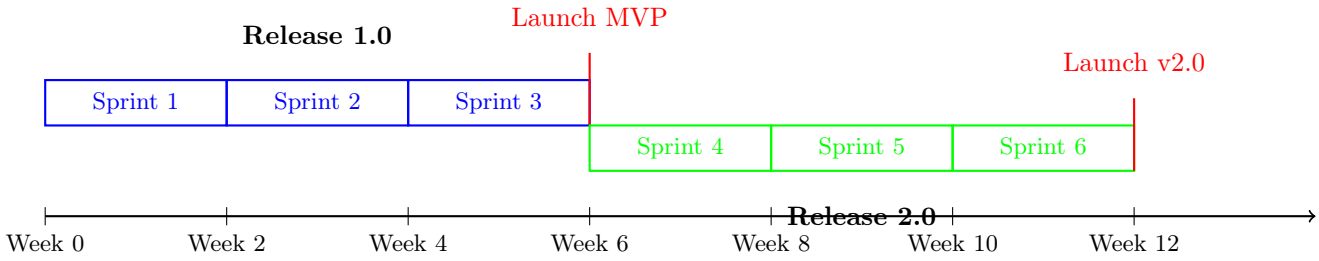

Figure 18: CookWise Release Timeline

The complete development and launch cycle spans 12 weeks: Release 1.0 MVP launches at Week 6, followed by Release 2.0 at Week 12.

6.6 Risk Management and Mitigation

Several risks could impact the release plan, and the following mitigation strategies have been identified:

- **Web Scraping Reliability**: Retailers may change website structure. Mitigation includes robust error handling and planned API transition.

- **Recipe Quality and Coverage**: Initial database may lack variety. Mitigation includes partnerships with food bloggers and user testing.
- **Performance**: Recipe suggestions may not meet 5 second target. Mitigation includes dedicated optimization time and caching strategies.
- **User Adoption**: MVP may not provide sufficient value. Mitigation includes user testing and marketing materials highlighting cost savings.

6.7 Success Metrics and Evaluation

The success of each release will be evaluated using the following metrics:

Release 1.0 Success Metrics:

- **User Acquisition**: 1,000 registered users within first month
- **User Engagement**: 60% of users generate at least one shopping list within first week
- **Performance**: 95% of recipe suggestions return within 5 seconds
- **Data Quality**: 95% accuracy in price data as validated weekly
- **Technical Stability**: Less than 5% error rate in web scraping operations

Release 2.0 Success Metrics:

- **User Growth**: 3,000 total registered users by end of Release 2.0
- **Feature Adoption**: 40% of users set dietary preferences within first week
- **Personalization Impact**: 30% increase in recipe selection rate for users with completed profiles
- **User Satisfaction**: Average rating of 4.0 out of 5.0 for usability
- **System Scalability**: Support 300 concurrent users with acceptable performance

These metrics will be monitored continuously after each release launch. If metrics fall short of targets, the team will analyze user feedback and adjust the release plan accordingly. The flexibility of the sprint based approach allows for rapid iteration and course correction based on real world data.

7 Policy and Regulation Requirements

CookWise must comply with various laws, regulations, and standards applicable to consumer applications operating in Sweden and the EU market.

Key Regulatory Requirements:

- **GDPR Compliance**: Lawful data processing, user consent, data minimization, user rights (access, erasure, portability, rectification), breach notification within 72 hours, privacy by design
- **Swedish Data Protection Act**: Registration as data controller with Swedish Authority for Privacy Protection
- **ePrivacy Directive**: Cookie consent banners, granular cookie controls, opt-in for non-essential cookies
- **EU Food Information Regulation**: Clear allergen labeling, accurate nutritional information in EU format, ingredient listings, disclaimers for AI-generated recipes
- **WCAG 2.1 Accessibility**: Semantic HTML, alt text for images, keyboard navigation, color contrast (4.5:1), screen reader support
- **Web Scraping Compliance**: Respect robots.txt, rate limiting, review retailer Terms of Service, plan transition to official APIs through partnerships

- **Intellectual Property**: AI-generated recipe ownership, proper attribution for third-party recipes, compliance with OpenAI/AI service terms
- **Terms of Service**: User responsibilities, liability limitations, acceptable use policy, service availability disclaimers
- **Security Standards**: ISO/IEC 27001 alignment, OWASP Top 10 mitigation, encryption of data at rest and in transit, secure authentication (OAuth 2.0)
- **Compliance Monitoring**: Quarterly regulatory reviews, annual privacy audits, security testing, documentation of data processing activities

8 References

[Lau] Søren Lauesen, *Software Requirements: Styles and Techniques*, Addison Wesley, ISBN 0-201-74570-4, 2002.

[QUPER] Björn Regnell, Richard Berntsson Svensson, and Thomas Olsson, "Supporting Roadmapping of Quality Requirements," *IEEE Software*, vol. 25, no. 2, pp. 42-47, 2008.

[GDPR] Regulation (EU) 2016/679, *General Data Protection Regulation*, European Parliament and Council, April 27, 2016.

9 Document Revision History

Version	Date	Name	Description
1	23/11/2025	Software Specification Release 1	Initial release with stakeholder analysis, elicitation techniques, requirements documentation, and data models.
2	14/12/2025	Software Specification Release 2	Release 2 with supervisor feedback from Release 1 addressed, quality requirements defined using the QUPER model, requirements prioritization, sprint-based planning, and policy and regulatory considerations.
3	10/01/2026	Software Specification Release 3	Release 3 with supervisor feedback from Release 2 addressed, condensed verbose sections throughout document.

10 Appendices

10.1 Appendix A: CookWise User Survey Questionnaire

This appendix contains the complete questionnaire administered to 25 participants as described in Section 3.4. The questionnaire was designed to quantitatively validate findings from interviews.
Section 1: Demographics Q1. What is your age group?

☐ 18-24 years

☐ 25-34 years

☐ 35-44 years

☐ 45-54 years

☐ 55+ years

Q2. What is your primary role in household grocery shopping?

☐ Primarily responsible

☐ Shared responsibility

☐ Occasionally help

☐ Not responsible

Q3. How many people live in your household?

☐ 1 person (living alone)

☐ 2 people

☐ 3-4 people

☐ 5+ people

Section 2: Shopping Behaviors

Q4. How often do you grocery shop?

☐ Multiple times per week

☐ Once per week

☐ Once every 2 weeks

☐ Less frequently

Q5. Do you typically shop at one store or multiple stores?

☐ Always at one store

☐ Rarely visit multiple stores (only in rare cases)

☐ Frequently shop at multiple stores

Q6. Which grocery stores do you use most frequently? (Select all that apply)

☐ ICA

☐ Willys

☐ Coop

☐ Lidl

☐ Hemköp

☐ Other: _____

Section 3: Price Sensitivity Q7. How often do you check for sales or discounts before shopping?

☐ Always

☐ Often

☐ Sometimes

☐ Rarely

☐ Never

Q8. How much money would you need to save to consider switching to a different store?

☐ Less than 25 SEK

☐ 25-50 SEK

☐ 50-100 SEK

☐ More than 100 SEK

☐ Would not switch regardless of savings

Q9. On a scale of 1-5, how important is it to see a detailed breakdown of your savings? (1 = Not important, 5 = Very important)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

Section 4: Meal Planning Habits Q10. How much time do you spend per week planning meals and creating shopping lists?

☐ Less than 15 minutes

☐ 15-30 minutes

☐ 30-60 minutes

☐ More than 60 minutes

Q11. Please rank the following factors in order of importance when planning meals: (1 = most important, 5 = least important)

Cost/Budget _____
Health/Nutrition _____
Time/Convenience _____
Family preference _____
Variety _____

Q12. Do you have any dietary restrictions or preferences?

☐ None

☐ Vegetarian

☐ Vegan

☐ Gluten-free

☐ Lactose-free

☐ Other: _____

Section 5: Interest in CookWise Concept

Q13. On a scale of 1-5, how interested would you be in an app that automatically suggests recipes based on current sales at your local stores? (1 = Not interested, 5 = Very interested)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

Q14. Which of the following features would be most valuable to you? (Select top 3)

☐ Recipe suggestions based on current sales

☐ Automatic shopping list generation

☐ Price comparison across stores

☐ Dietary restriction filters

☐ Meal planning calendar

☐ Recipe ratings and reviews

☐ Savings calculator

Q15. What would prevent you from using such an app? (Open-ended)

10.2 Appendix B: Entity Relation Diagram - PlantUML Source Code

This appendix contains the PlantUML source code for the Entity Relation Diagram shown in Section 4.3. This code can be used to regenerate or modify the diagram using any PlantUML renderer.

```
@startuml CookWise_ERD

skinparam dpi 300

skinparam linetype ortho
skinparam nodesep 40
skinparam ranksep 45
skinparam defaultFontSize 9
skinparam padding 2
skinparam entity {
  FontSize 9
  AttributeFontSize 8
}

together {
  entity "USER" as USER #E8E8E8 {
    * User_ID <<PK>>
    --
    Email
    Auth_Provider
    Latitude, Longitude
    Dietary_Restrictions
  }

  entity "RATING" as RATING #FAD7A0 {
    * Rating_ID <<PK>>
    --
    User_ID <<FK>>
    Recipe_ID <<FK>>
    Rating_Score
  }

  entity "RECIPE" as RECIPE #D4E6F1 {
    * Recipe_ID <<PK>>
    --
    Recipe_Name
    Cuisine_Type
    Cooking_Time
    Difficulty_Level
    Instructions
  }
}

USER -[hidden]right- RATING
RATING -[hidden]right- RECIPE

together {
  entity "RECIPE_INGREDIENT" as RI #D5F4E6 {
    * Recipe_ID <<FK>>
    * Ingredient_ID <<FK>>
    --
    Quantity
    Unit
  }
```

```
  entity "INGREDIENT" as ING #FAD7A0 {
    * Ingredient_ID <<PK>>
    --
    Ingredient_Name
    Category
  }
}

RI -[hidden]right- ING

together {
  entity "SHOPPING_LIST" as SL #FADBD8 {
    * List_ID <<PK>>
    --
    User_ID <<FK>>
    Store_ID <<FK>>
    Total_Price
  }

  entity "STORE" as STORE #D5F4E6 {
    * Store_ID <<PK>>
    --
    Store_Name
    Latitude, Longitude
  }

  entity "SALE_ITEM" as SALE #D4E6F1 {
    * Sale_ID <<PK>>
    --
    Store_ID <<FK>>
    Ingredient_ID <<FK>>
    Sale_Price
    Valid_From, Valid_To
  }

  entity "SHOPPING_LIST_ITEM" as SLI #E8DAEF {
    * List_ID <<FK>>
    * Ingredient_ID <<FK>>
    --
    Quantity
    Sale_ID <<FK>>
  }
}

SL -[hidden]right- STORE
STORE -[hidden]right- SALE
SALE -[hidden]right- SLI

USER -[hidden]down- RI
RATING -[hidden]down- RI
RECIPE -[hidden]down- ING
RI -[hidden]down- SL
ING -[hidden]down- STORE

USER ||--o{ RATING : ""
USER ||--down-o{ SL : ""
```

```
RATING }o--|| RECIPE : ""
RECIPE ||--down-o{ RI : ""
RI }o--|| ING : ""

SL ||--right-o{ SLI : ""
SL }o--right-|| STORE : ""

ING ||--down-o{ SALE : ""
ING ||--down-o{ SLI : ""

STORE ||--down-o{ SALE : ""
SALE ||--left-o{ SLI : ""

@enduml
```

To regenerate the diagram:

1. Copy the code above
2. Visit https://www.plantuml.com/plantuml/uml/ or use a local PlantUML installation
3. Paste the code and generate the diagram
4. Export as PNG or SVG as needed

10.3 Appendix C: Use Case Diagrams - PlantUML Source Code

This appendix contains the PlantUML source code for the use case diagrams shown in Section 4.2. These codes can be used to regenerate or modify the diagrams using any PlantUML renderer.
Overall System Use Case Diagram

```
@startuml CookWise_Overall
left to right direction
skinparam packageStyle rectangle
skinparam linetype ortho

actor "User" as User

rectangle "CookWise System" {
  usecase "Login to System" as UC1
  usecase "Select Store" as UC2
  usecase "Browse Recipes\nBased on Store Sales" as UC3
  usecase "View Recipe Details" as UC4
  usecase "Create Shopping List" as UC5
  usecase "View Total Savings" as UC6
  usecase "Get Store Directions" as UC7
  usecase "Rate Recipes" as UC8
  usecase "Save Favorites" as UC9
}

User --> UC1
User --> UC2
User --> UC3
User --> UC4
User --> UC5
User --> UC6
User --> UC7
User --> UC8
User --> UC9
```

```
UC3 ..> UC2 : <<include>>
UC5 ..> UC3 : <<include>>

@enduml
```

Detailed Use Case: Browse Recipes from Store Sale Items

```
@startuml Browse_Recipes_Detail
skinparam packageStyle rectangle
skinparam linetype ortho

actor "User" as User

rectangle "Browse Recipes from Store Sale Items" {

  usecase "Browse Recipes from\nStore Sale Items" as Main #LightBlue

  usecase "Select Store" as SelectStore
  usecase "Display Sale-Based Recipes" as ShowRecipes
  usecase "Select Recipe(s)" as SelectRecipe
  usecase "Generate Shopping List" as GenList

  usecase "Filter Recipes" as Filter #LightYellow
  usecase "Search by Ingredient" as Search #LightYellow
  usecase "View Recipe Details" as ViewDetails #LightYellow
}

User --> Main

Main ..> SelectStore : <<include>>
Main ..> ShowRecipes : <<include>>
Main ..> SelectRecipe : <<include>>
Main ..> GenList : <<include>>

Filter ..> ShowRecipes : <<extend>>
Search ..> ShowRecipes : <<extend>>
ViewDetails ..> SelectRecipe : <<extend>>

@enduml
```

10.4 Appendix D: AI-Generated Prototype Prompt (Base64)

This appendix contains the complete prompt used to generate the functional Base64 prototype described in Section 3.5. The prompt was provided to Claude AI (with Artifacts capability) to create a working interactive prototype of the CookWise application.

```
I need you to create a functional prototype for CookWise, a meal planning
and grocery shopping optimization app for the Swedish market. This is based
on our comprehensive System Requirements Document.

Project Overview
CookWise helps budget-conscious Swedish families save money by suggesting
recipes based on current sales at major grocery stores (ICA, Willys, Coop)
and generating optimized shopping lists.

Core Features to Implement (MVP - Release 1.0)

User Authentication
- External authentication via Google/Apple ID
```

```
- User profile with dietary restrictions (vegetarian, vegan, gluten-free,
  lactose-free)
- Location-based services (Swedish cities, default: Karlskrona)

Recipe System
- Database of Swedish recipes with local ingredients
- AI-generated recipe suggestions based on current sales
- Recipe search and filtering by:
  * Cuisine type (Swedish, Mediterranean, Asian, etc.)
  * Cooking time
  * Difficulty level
  * Dietary restrictions
  * Budget level (Low Budget, Medium, High)
- Recipe detail pages showing:
  * Ingredients with quantities
  * Step-by-step instructions
  * Nutritional information
  * Estimated cost
  * Potential savings

Store & Price Data
- Support for three major Swedish retailers: ICA, Willys, Coop
- Display current sale items by category (Fruits, Vegetables, Bakery,
  Seafood, Nuts)
- Price comparison across retailers for same products
- Store location data with map integration

Shopping List Generation
- Automatic shopping list creation from selected recipes
- Single-store optimization (recommend best store based on total price
  + distance)
- Display of savings calculation
- Quantity adjustment controls
- Running total display

Navigation & UI
- Bottom navigation: Home, Items, Recipes, User
- Swedish language interface
- Mobile-first responsive design
- Clean, modern aesthetic with food photography

Key Screens to Implement
1. Home Screen: Store selection, search bar, promotional banner,
   featured recipes
2. Store Pages: Category tabs, product grid with prices, "Add to cart"
   buttons
3. Recipe Browse: Category filters, recipe cards with images
4. Recipe Detail: Hero image, ingredients list, instructions, price
   breakdown
5. Shopping Cart: Item list, quantities, running total, store display
6. User Profile: Settings, dietary preferences, purchase history


Specific Implementation Requests
- Create a mock database with 20-30 Swedish recipes and sample sale data
- Implement the recipe suggestion algorithm that matches recipes with
  current sales
- Build the shopping list optimizer that recommends the best single store
```

```
- Include realistic Swedish product names and prices in SEK
- Add sample store locations in Karlskrona and surrounding areas
- Implement basic search and filtering functionality
- Show savings calculations clearly to users

Context
- Target market: Sweden (Swedish language, SEK currency, local stores)
- Target users: Budget-conscious families, busy professionals, cooking
  enthusiasts
- Key value proposition: Save money by cooking with sale items
- Competitive advantage: Automatic sale-to-recipe matching

Please create a working prototype that demonstrates the core user flow:
Browse sale items → Get recipe suggestions → Generate shopping list →
See savings. Focus on making it functional and visually appealing rather
than production-ready. Use placeholder/mock data where real integrations
would be required.
```

10.5 Appendix E: Complete Requirements Checklist

This appendix provides a master checklist of all CookWise system requirements organized by type. This checklist can be used for tracking implementation progress, testing coverage, and release planning.

ID	Requirement Description	Priority	Release
Domain Level Requirements (DL)			
DL1	Web scraping for product and pricing data from grocery stores	Must Have	1.0
DL2	GDPR compliance for user and partner data	Must Have	1.0
DL3	Single-store shopping constraint (all items from one location)	Should Have	2.0
DL4	Total basket price calculation includes promotions and discounts	Must Have	1.0
DL5	Rate limits and operational boundaries for external interfaces	Should Have	2.0
DL6	Domain event processing (user actions, external responses, scheduled operations)	Could Have	3.0
Product Requirements (PR)			
PR1	External authentication (Google/Apple ID)	Must Have	1.0
PR2	AI Recipe Generation API integration	Must Have	1.0
PR2.1	Display top 5 AI-generated recipes ranked by cost savings	Must Have	1.0
PR3	Recipe search filtering (cuisine, dietary, budget, time, difficulty, ratings)	Must Have	1.0
PR4	Automatic consolidated shopping list generation	Must Have	1.0
PR5	Record individual user ratings (1-5 scale) and text feedback	Should Have	2.0
PR6	Store AI-generated recipes and details	Must Have	1.0
PR7	Display recipe details (ingredients, time, difficulty, instructions, nutrition)	Must Have	1.0
PR8	Display store price comparisons and distances from user location	Should Have	2.0
PR9	Display total savings amount and itemized price differences	Must Have	1.0
PR10	Display recipes containing specified ingredients	Should Have	2.0
PR11	Display all ingredients with sale items highlighted and listed first	Must Have	1.0
PR12	Daily web scraping at 02:00 local time	Must Have	1.0

ID	Requirement Description	Priority	Release
PR13	Store user preferences for ingredients and recipes	Should Have	2.0
PR14	Provide "Get Directions" links to external mapping applications	Could Have	3.0
PR15	Calculate and display average ratings for each recipe	Should Have	2.0
Data Requirements (DR)			
DR1	Core entity data models (User, Recipe, Ingredient, Rating, Store, Sale Item)	Must Have	1.0
DR2	Relationship and associative entity data models (Recipe-Ingredient, Shopping List, Shopping List Item)	Must Have	1.0
DR3	Data interchange formats (JSON for APIs, CSV/JSON for export, HTML parsing, UTF-8 encoding)	Should Have	2.0
DR4	System states and initial seed data (100 recipes, store info, 500 ingredients)	Should Have	2.0
Quality Requirements (QR)			
QR1	Accuracy: 95% accuracy in sale price data	Must Have	1.0
QR2	Reliability: 98% uptime during peak hours (17:00-20:00)	Should Have	2.0
QR3	Performance: 5 seconds recipe generation response time (95th percentile)	Must Have	1.0
QR4	Usability: 8 minutes for first recipe selection (new users)	Should Have	2.0
QR5	Scalability: Support 300 concurrent users with acceptable performance	Should Have	2.0