**요구사항 명세서**

버전: 0 . 4

작성일: 2024.05.12

작성자: 김창현

목 차

[1. 개요 4](#_Toc206345667)

[2. 기능적 요구사항 5](#_Toc206345668)

[2.1. 시스템 기능 구조 5](#_Toc206345669)

[2.1.1. 유스케이스 패키지 구조도 5](#_Toc206345670)

[2.1.2. 유스케이스 패키지 개요 5](#_Toc206345671)

[2.2. 유스케이스 패키지 명세: 대출관리 6](#_Toc206345672)

[2.2.1. 유스케이스 다이어그램 6](#_Toc206345673)

[2.2.2. 액터 개요 7](#_Toc206345674)

[2.2.3. 유스케이스 개요 7](#_Toc206345675)

[2.2.4. 유스케이스 명세: 도서대출신청 8](#_Toc206345676)

[2.2.4.1. 개요 8](#_Toc206345677)

[2.2.4.2. 관련 액터 8](#_Toc206345678)

[2.2.4.3. 우선순위 8](#_Toc206345679)

[2.2.4.4. 선행 조건 8](#_Toc206345680)

[2.2.4.5. 후행 조건 9](#_Toc206345681)

[2.2.4.6. 시나리오 9](#_Toc206345682)

[2.2.4.7. 비기능적 요구사항 10](#_Toc206345683)

[2.2.5. 유스케이스 명세: 소장도서검색 10](#_Toc206345684)

[2.2.5.1. 개요 10](#_Toc206345685)

[2.2.5.2. 관련 액터 10](#_Toc206345686)

[2.2.5.3. 우선순위 10](#_Toc206345687)

[2.2.5.4. 선행 조건 10](#_Toc206345688)

[2.2.5.5. 후행 조건 10](#_Toc206345689)

[2.2.5.6. 시나리오 10](#_Toc206345690)

[2.2.5.7. 비기능적 요구사항 10](#_Toc206345691)

[*2.3.* 유스케이스 패키지 명세: *패키지2* 11](#_Toc206345692)

[2.3.1. 유스케이스 다이어그램 11](#_Toc206345693)

[2.3.2. 액터 개요 11](#_Toc206345694)

[2.3.3. 유스케이스 개요 11](#_Toc206345695)

[2.3.4. 유스케이스 명세: 유스케이스2-1 11](#_Toc206345696)

[2.3.5. 유스케이스명세: 유스케이스2-2 11](#_Toc206345697)

[3. 시스템 품질 요구사항 12](#_Toc206345698)

[3.1. 성능 12](#_Toc206345699)

[3.2. 신뢰도 12](#_Toc206345700)

[3.3. 확장성 12](#_Toc206345701)

[3.4. 보안성 12](#_Toc206345702)

[4. 개발 제약 사항 13](#_Toc206345703)

# 개요

|  |
| --- |
| *문서의 목적은 주문 관리 프로그램의 요구사항을 파악하고 문서화하여 프로젝트의 방향과 개발 과정을 작성하며, 제품의 품질을 향상시키는 데 있습니다.*  *문서의 내용에 대한 개요*   1. 프로그램 기능 2. 사용자 요구사항 3. 프로그램 사용 시나리오 4. 시스템 품질 요구사항   이 요구사항 명세서는 개발팀과 운영팀과의 소통을 위한 문서로 작용합니다.  명세서에는 시스템의 기능적, 비기능적 요구사항, 각 유스케이스의 명세, 시스템 품질 요구사항, 개발 제약 사항 등이 포함되어 있습니다. |

# 기능적 요구사항

## 시스템 기능 구조

|  |
| --- |
| *시스템의 기능에 대한 구조를 유스케이스 패키지를 이용하여 표현한다. 규모가 큰 시스템인 경우에는 계층적 구조가 될 수 있다.* |

### 유스케이스 패키지 구조도

텍스트, 직사각형, 화이트, 폰트이(가) 표시된 사진

자동 생성된 설명

### 유스케이스 패키지 개요

|  |  |
| --- | --- |
| 패키지 명 | 설명 |
| 상품관리 | 카페 메뉴에 포함될 음료: 상품[이름, 재료, 가격, 등록, 수정, 삭제]를 조회할 수 있는 기능을 포함한다. |
| 결제관리 | 선택한 메뉴를 선정한 후 주문 시 결제기능을 내포한다. |
| 주문관리 | 사용자가 상품을 주문하는 과정부터 결과까지 결제[상품명, 수량, 금액, 일자]를 조회할 수 있는 기능을 포함한다. |

## 유스케이스 패키지 명세: 상품관리

### 유스케이스 다이어그램

*도표, 라인, 기술 도면, 종이접기이(가) 표시된 사진

자동 생성된 설명*

### 액터 개요

|  |  |  |
| --- | --- | --- |
| 액터 명 | 유형 | 설명 |
| 시스템 사용자 | 사용자 | 프로그램을 사용할 고객들 |
| 현장 관리자 | 사용자 | 카페에서 일하는 직원(소상공인) |

### 유스케이스 개요

|  |  |
| --- | --- |
| 유스케이스 명 | 설명 |
| 메뉴관리 | 카페 메뉴의 등록, 수정, 삭제 및 조회 기능 |
| 결제관리 | 선택한 메뉴 주문 시 결제 기능 |
| 주문관리 | 사용자의 주문 입력, 주문 내역 조회 및 주문 상태 관리 기능 |

### 유스케이스 명세: 주문관리

#### 개요

* 사용자는 메뉴 목록에서 주문할 항목을 선택합니다.
* 선택한 항목을 장바구니에 추가하고 주문을 요청합니다
* 사용자는 주문 수령을 위한 시간과 장소를 지정합니다.
* 시스템은 주문을 접수하고, 사용자에게 주문 완료 알림을 전송합니다.

#### 관련 액터

주액터: 주문 진행 (소비자, 카페 사장님)

보조액터: 없음

#### 우선순위

중요도: 상, 난이도: 상

#### 선행 조건

* 메뉴 정보 관리 및 시스템 구축
* 주문 및 결제 시스템 연동
* 장바구니 기능
* 사용자 경험 고려.

#### 후행 조건

* 앱 안정성 및 신뢰성 확보
* 보안 강화
* 성능 최적화
* 사용자 피드백 반영
* 운영 및 유지보수 계획 수립

#### 시나리오

**기본 시나리오:**

1. 주문하기 및 픽업 예약

* 메뉴 선택
* 장바구니 담기
* 주문 요청

1. 주문 상태 확인

* 주문 상태 조회
* 주문 완료

1. 음료, 음식 수령

* 픽업

1. 주문 이력 관리

* 주문 취소

**주문조회 및 취소 시나리오:**

1. 고객이 화면의 주문 조회를 클릭한다.
2. 주문 조회 결과가 표시된다. ( 주문의 상태, 주문한 메뉴 목록, 주문 총액, 주문 일시 등의 정보를 포함 )
3. 화면의 주문 취소를 선택한다. ( 취소하고자 하는 경우 )
4. 주문 취소를 하는 지 다시 한 번 확인한다. 확인을 누를 경우 주문 취소 프로세스를 시작한다.
5. 시스템은 데이터베이스에서 해당 주문을 찾아서 취소 처리한다. 주문된 메뉴의 재고를 복원한다.
6. 취소가 완료되었다는 메시지를 고객에게 표시한다.
7. 시스템은 주문 취소 내역을 기록하여 관리자나 직원이 추후에도 확인할 수 있도록 한다.

#### 비기능적 요구사항

해당 없음

### 유스케이스 명세: 메뉴 관리

#### 개요

시스템에 등록된 메뉴 (음식, 음료)를 관리하고 수정하는 기능을 제공합니다. 메뉴의 추가, 수정 삭제 조회 등을 포함합니다.

#### 관련 액터

관리자, 시스템

#### 우선순위

#### 선행 조건

1. 시스템 구현 및 배포
2. 메뉴 정보 모델링 및 저장 구조 설계
3. 기본 메뉴 정보 등록
4. 보안 및 권한 관리 정책 준수

#### 후행 조건

1. 메뉴 정보의 정확성과 일관성 유지
2. 시스템의 안정성과 신뢰성 유지
3. 사용자에게 제공되는 메뉴 정보의 최신화
4. 정보 보안 및 개인정보 보호 정책 준수

시나리오

1. 메뉴추가
2. 메뉴 조회 및 수정
3. 메뉴 삭제
4. 작업 결과 확인

#### 비기능적 요구사항

해당 없음.

### 유스케이스 명세: 결제 관리

|  |
| --- |
| *시스템에서 주문한 상품에 대한 결제를 처리하고 관리하는 기능을 제공합니다. 주문 정보 확인, 결제 처리, 결제 상태 조회 등을 포함합니다.* |

#### 개요

* 권한 관리
* 세션 관리

#### 관련 액터

주액터: 결제 처리 (고객, 결제 서비스, 시스템)

보조액터: 없음

#### 우선순위

중요도: 상, 난이도: 상

#### 선행 조건

* 주문 정보 등록
* 결제 서비스 연동
* 결제 정보 입력 인터페이스 제공
* 보안 및 인증 메커니즘 구현
* 결제 관리 기능 활성화

#### 후행 조건

* 결제 완료 상태 유지
* 주문 처리 및 배송 진행
* 결제 정보의 안전한 보관
* 고객에게 결제 완료 메시지 제공
* 결제 실패 처리
* 결제 관련 로그 기록
* 시스템의 안정성 유지

#### 시나리오

**기본 시나리오:**

1. 결제 정보 입력
2. 결제 요청 및 처리
3. 결제 완료 상태 확인

**결제 취소 시나리오:**

1. 취소 요청
2. 취소 가능 여부 확인
3. 취소 처리
4. 취소 완료 상태 확인

#### 비기능적 요구사항

해당 없음

# 3.시스템 품질 요구사항

## 성능

시스템은 사용자의 요청을 신속하게 처리하여 뛰어난 사용자 경험을 제공해야 합니다. 이를 위해 다음과 같은 성능 요구사항을 충족해야 합니다:

* 응답 시간: 모든 사용자 요청에 대한 응답 시간은 최대 2초를 초과하지 않아야 합니다. 특히, 주문 처리 및 결제 과정에서는 1초 미만의 응답 시간을 목표로 합니다.
* 처리량: 시스템은 피크 타임 동안에도 최소 초당 100건 이상의 주문을 처리할 수 있는 능력을 갖추어야 합니다. 이를 위해 시스템은 부하 분산 및 효율적인 리소스 관리 메커니즘을 구현해야 합니다.
* 자원 사용률: 시스템의 자원 사용률은 효율적으로 관리되어야 하며, CPU 사용률과 메모리 사용률은 일반적인 운영 상황에서 70% 미만으로 유지되어야 합니다. 이를 위해 코드 최적화 및 리소스 관리 전략이 필요합니다.

## 신뢰도

시스템은 안정적인 운영을 보장하여 사용자와 운영자 모두에게 신뢰를 제공해야 합니다:

* 가용성: 연간 가용성은 99% 이상이 되어야 합니다. 이는 시스템이 연중 대부분의 시간 동안 안정적으로 운영됨을 의미합니다.
* 데이터 무결성: 데이터 처리 과정에서의 오류율은 0.01% 미만이어야 하며, 모든 데이터는 정확하고 완전한 상태로 유지되어야 합니다. 이를 달성하기 위해 데이터 검증 및 정정 메커니즘을 구현해야 합니다.
* 데이터 무결성: 데이터 처리 과정에서의 오류율은 0.01% 미만이어야 하며, 모든 데이터는 정확하고 완전한 상태로 유지되어야 합니다. 이를 달성하기 위해 데이터 검증 및 정정 메커니즘을 구현해야 합니다.

## 확장성

시스템은 비즈니스 성장 및 기술 변화에 유연하게 대응할 수 있어야 합니다:

* 시스템 확장: 새로운 기능 추가나 사용자 증가에 따른 시스템 확장이 용이해야 합니다. 모듈식 아키텍처와 마이크로서비스 디자인 패턴을 적용하여, 개별 서비스나 컴포넌트의 업데이트 및 확장이 다른 시스템 부분에 영향을 주지 않도록 설계해야 합니다.
* 데이터베이스 확장성: 데이터베이스는 증가하는 데이터 볼륨과 요청 처리 요구사항을 수용할 수 있도록 설계되어야 합니다. 이를 위해 수평적, 수직적 확장이 가능한 데이터베이스 솔루션을 선택하고, 필요에 따라 샤딩, 리플리케이션, 클러스터링 전략을 적용해야 합니다.

## 보안성

시스템은 사용자 데이터 보호 및 비즈니스 운영의 안전을 보장하는 높은 수준의 보안성을 유지해야 합니다:

* 데이터 암호화: 모든 민감한 데이터는 전송 및 저장 시점에서 강력한 암호화 알고리즘을 사용하여 암호화되어야 합니다. SSL/TLS를 사용한 데이터 전송 암호화 및 AES 같은 알고리즘을 사용한 저장 데이터 암호화가 필수적입니다.
* 접근 제어: 시스템은 역할 기반 접근 제어(RBAC)를 구현하여, 사용자와 관리자가 자신의 권한에 맞는 데이터와 기능에만 접근할 수 있도록 해야 합니다. 이는 불필요한 데이터 노출 위험을 최소화합니다.
* 시큐어 코딩: 개발 과정에서는 시큐어 코딩 원칙을 준수하여, SQL 인젝션, 크로스 사이트 스크립팅(XSS), 크로스 사이트 요청 위조(CSRF) 등의 보안 취약점을 방지해야 합니다. 이를 위해 정기적인 코드 리뷰와 보안 취약점 스캔을 실시해야 합니다.

# 개발 제약 사항

|  |
| --- |
| *운영체제, 프레임웍, 연동 시스템, 개발 방법론 등 소프트웨어 설계 측면의 제약할 수 있는 사항을 기록한다.* |

시스템 설계 및 개발 과정에서 다음과 같은 제약 사항을 고려해야 합니다.

운영 체제

* 지원 운영 체제 : windows와 macOS 운영 체제에서 호환될 수 있도록 개발되어야 합니다. 대학 내에서 가장 많이 사용되는 운영체제를 고려하여 개발해야 합니다.

프레임워크

* 제약 사항: 네이티브 개발 툴셋(Swift for iOS, Kotlin for Android) 사용을 고려할 때, 각 운영 체제에 맞는 개발 환경과 전문 지식이 요구됩니다.

인터커넥션 시스템

* API 및 데이터 포맷: 서버와 애플리케이션 간의 통신은 RESTful API 또는 GraphQL API를 사용하며, JSON 또는 XML 포맷으로 데이터를 교환합니다.
* 제약 사항: API 설계와 구현은 보안, 데이터 효율성 및 플랫폼 간 호환성을 고려해야 하며, 이는 시스템의 전체적인 아키텍처에 영향을 줍니다.

개발 방법론

* 적용 방법론: Agile 개발 방법론과 그 하위 집합인 Scrum 또는 Kanban을 프로젝트 관리에 적용하여, 변화하는 요구사항에 대응하고 효율적인 작업 진행을 도모해야 합니다. 2주 간격의 스프린트, 지속적인 통합(CI), 지속적인 배포(CD) 등의 실천이 포함되어야 합니다.
* 제약 사항: 전통적인 워터폴 모델이나 다른 개발 방법론의 적용은 프로젝트의 유연성과 반응성을 저하시킬 수 있으며, Agile 방법론에 익숙하지 않은 팀원들은 초기 학습 과정을 거쳐야 할 수 있습니다.