6/09
8조 **Fotential**

POTENTIAL

과 목 명 **:** 소프트웨어 공학

학 과 **:** 컴퓨터소프트웨어과

학 번 **: 2006094** 박기준
2106066 하종수
2006103 김강민
2106089 정승훈

목차

1.1 문서의 범위

재료 기반 레시피 추천: 사용자가 입력한 재료를 바탕으로 만들 수 있는 다양한 요리 레시피를 추천합니다. 사용자는 냉장고에 있는 재료를 입력하여, 그 재료로 만들 수 있는 요리를 쉽게 찾을 수 있습니다.

날씨 기반 레시피 추천: 기상청 API를 활용하여 현재 지역의 날씨 정보를 수집하고, 날씨에 적합한 요리 레시피를 추천합니다. 예를 들어, 비오는 날에는 따뜻한 국물 요리나 전을 추천하고, 더운 날에는 시원한 냉면이나 샐러드를 추천합니다.

로그인 및 즐겨찾기 기능: 사용자는 구글 또는 카카오 계정을 통해 앱에 로그인할 수 있으며, 마음에 드는 요리를 즐겨찾기에 추가하여 쉽게 접근할 수 있습니다. 즐겨찾기 기능을 통해 자주 찾는 레시피를 저장하고, 나중에 다시 확인할 수 있습니다.

앱의 주요 사용자는 선택장애가 있는 사람들이나 요리를 잘 하지 못하는 사람들입니다. 이 앱은 요리 선택의 어려움을 덜어주고, 사용자가 쉽게 요리를 준비할 수 있도록 돕는 것을 목표로 합니다. 또한, 요리에 대한 경험이 부족한 사용자도 쉽게 따라할 수 있는 직관적인 인터페이스를 제공합니다.

1.2 참고 자료

기상청 API 공식 문서
Firebase 개발자 가이드
국민건강영양조사 식생활 패턴 자료
공공데이터 사이트

1.3 정의**,** 약어

API: Application Programming Interface
DB: Database
UI: User Interface
UX: User Experience
Firebase: Google의 클라우드 기반 백엔드 서비스

2. 테스트 범위와 목표

2.1 프로젝트 단계**/**유형

설계 단계:

UI/UX 디자인 검토: 사용자 인터페이스가 직관적이고 사용하기 쉬운지 평가합니다. 디자인 시안은 Figma를 사용하여 검토하며, 사용자 피드백을 통해 개선점을 찾습니다.

개발 단계:

기능 구현 및 초기 테스트: 각 기능이 예상대로 작동하는지 확인합니다. 로그인, 재료 입력, 레시피 추천 기능을 구현하고, 기본적인 작동 여부를 테스트합니다.

테스트 단계:

기능 테스트: 모든 기능이 예상대로 작동하는지 확인합니다. 특히, 재료 기반 레시피 추천과 날씨 기반 메뉴 추천 기능의 정확성을 검증합니다.
성능 테스트: 앱을 사용하면서 버벅거림이 없는지, 로딩 시간과 반응 속도가 빠른지 확인합니다. 앱의 성능이 사용자 경험에 영향을 미치지 않도록 최적화합니다.
사용자 테스트: 실제 사용자 피드백을 통해 사용성을 평가합니다. 선택장애가 있는 사용자나 요리를 잘 하지 못하는 사용자가 앱을 쉽게 사용할 수 있는지 확인합니다.

배포 단계:

최종 검토 및 출시 준비: 모든 테스트가 완료되고, 발견된 문제점이 수정되었는지 확인합니다. 앱을 Google Play 스토어에 배포할 준비를 합니다.

2.2 테스트 범위

사용자 프로필 기반 추천 기능

프로필 입력 및 수정: 사용자가 성별, 나이, 식습관 등의 정보를 입력하고 수정할 수 있는지 확인합니다.
추천 알고리즘: 입력된 프로필 정보를 기반으로 적절한 메뉴를 추천하는 알고리즘의 정확성을 테스트합니다.

보유 재료 기반 레시피 검색 및 추천

재료 입력 및 관리: 사용자가 냉장고에 있는 재료를 입력하고 관리할 수 있는지 확인합니다.
레시피 추천: 입력된 재료를 기반으로 가능한 요리 레시피를 추천하는 기능의 정확성을 테스트합니다.

날씨 데이터 기반 오늘의 메뉴 추천

날씨 데이터 수집: 기상청 API를 통해 현재 지역의 날씨 정보를 정확하게 수집하는지 확인합니다.
날씨 기반 추천: 수집된 날씨 정보를 바탕으로 적절한 메뉴를 추천하는 기능의 정확성을 테스트합니다.

즐겨찾기 및 음식 기록 기능

즐겨찾기 추가 및 관리: 사용자가 마음에 드는 레시피를 즐겨찾기에 추가하고 관리할 수 있는지 확인합니다.
음식 기록: 사용자가 이전에 선택한 메뉴를 기록하고 다시 조회할 수 있는지 테스트합니다.

로그인/회원가입

계정 생성 및 로그인: 사용자가 구글 또는 카카오 계정을 통해 회원가입하고 로그인할 수 있는지 확인합니다.
자동 로그인: 사용자가 앱을 재실행할 때 자동으로 로그인되는지 테스트합니다.

사용자 프로필 입력

프로필 입력 화면: 사용자가 자신의 정보를 입력할 수 있는 화면의 사용성을 테스트합니다.

날씨 기반 메뉴 추천

메뉴 추천 정확성: 날씨에 따라 추천되는 메뉴가 적절한지 확인합니다.

레시피 검색 및 정보 조회

검색 기능: 사용자가 입력한 키워드로 레시피를 검색할 수 있는지 확인합니다.
정보 조회: 검색된 레시피의 상세 정보를 정확하게 조회할 수 있는지 테스트합니다.

예외 처리 **(API** 실패**,** 미입력 값 등**)**

API 오류 처리: API 호출 실패 시 사용자에게 적절한 대체 문구를 제공하는지 확인합니다.
미입력 값 처리: 필수 입력 값이 누락된 경우 사용자에게 입력 요청 팝업을 제공하는지 테스트합니다.

2.3 테스트 목표

안정성

다양한 환경에서의 작동: 앱이 다양한 Android 기기 및 운영체제 버전에서 안정적으로 작동하는지 확인합니다. 이를 위해 여러 기기에서 테스트를 수행하고, 앱이 크래시 없이 작동하는지 검증합니다.
네트워크 조건: 다양한 네트워크 조건(예: Wi-Fi, 4G, 5G)에서 앱의 성능을 평가하여, 네트워크 연결 상태에 따라 앱이 안정적으로 작동하는지 확인합니다.

사용성

직관적인 사용자 인터페이스: 앱의 UI가 사용자가 쉽게 이해하고 사용할 수 있도록 직관적으로 설계되었는지 평가합니다. 사용자 인터페이스의 각 요소(버튼, 메뉴, 입력 필드 등)가 명확하고 접근하기 쉬운지 확인합니다.
사용자 피드백: 실제 사용자 테스트를 통해 인터페이스의 사용성을 평가하고, 사용자 피드백을 바탕으로 개선점을 찾습니다. 사용자가 앱을 사용할 때 불편함을 느끼지 않도록 UI를 최적화합니다.

정확성

추천 기능의 정확성: 사용자 프로필 및 입력된 재료에 맞게 정확한 메뉴를 추천하는지 검증합니다. 추천 알고리즘이 사용자 요구에 맞게 작동하는지 확인하고, 추천 결과가 사용자 기대에 부합하는지 평가합니다.
날씨 기반 추천: 날씨 데이터에 기반한 메뉴 추천이 적절한지 확인합니다. 날씨 변화에 따라 추천되는 메뉴가 사용자에게 유용한지 검증합니다.

2.4 가정과 제약 사항

가정

기기 호환성: 모든 사용자가 Android 8.0 이상 버전을 사용하는 기기를 보유하고 있다고 가정합니다. 이는 앱의 기능이 최신 Android 환경에서 최적화되어 작동하도록 설계되었음을 의미합니다.
인터넷 연결: 사용자가 앱을 사용할 때 항상 인터넷에 연결되어 있다고 가정합니다. 이는 앱이 외부 API와 데이터베이스에 접근하여 실시간으로 데이터를 수집하고 처리하는 데 필수적입니다.

제약 사항

API 연동: 앱의 기능은 기상청 API, Firebase, 공공데이터 API와의 원활한 연동에 의존합니다. API 호출이 실패하거나 지연될 경우, 앱의 일부 기능이 제한될 수 있습니다. 따라서 API의 안정성과 응답 속도가 중요합니다.
데이터베이스 접근: 사용자 프로필, 재료 목록, 즐겨찾기 정보 등을 저장하고 조회하기 위해 Firebase 데이터베이스에 원활하게 접근할 수 있어야 합니다. 데이터베이스 연결이 불안정할 경우, 사용자 경험에 부정적인 영향을 미칠 수 있습니다.

기기 성능: 앱이 다양한 Android 기기에서 원활하게 작동하도록 최적화되어야 합니다. 특히, 저사양 기기에서도 앱이 버벅거림 없이 작동할 수 있도록 성능 최적화가 필요합니다.

2.5 이해 관계자

개발팀

역할: 앱의 설계, 개발, 테스트, 배포 및 유지보수를 담당합니다. 개발팀은 앱의 기능적 요구사항을 구현하고, 사용자 피드백을 반영하여 지속적으로 앱을 개선합니다.
구성: 개발자, 디자이너, 테스터 등으로 구성되며, 각 구성원은 자신의 전문 분야에서 프로젝트의 목표를 달성하기 위해 협력합니다.

사용자

역할: 앱을 사용하는 최종 소비자로서, 앱의 기능과 사용성을 평가합니다. 사용자의 피드백은 앱의 개선과 발전에 중요한 정보를 제공합니다.
특징: 선택장애가 있는 사람들, 요리를 잘 하지 못하는 사람들, 바쁜 일상 속에서 효율적인 식사 준비를 원하는 사람들이 주요 사용자입니다.

투자자

역할: 앱 개발에 필요한 자금을 지원하는 이해 관계자입니다. 투자자는 프로젝트의 재정적 성공을 기대하며, 앱의 시장성과 수익성을 평가합니다.

관심사: 투자자는 앱의 시장 진입 전략, 경쟁력, 수익 모델 등에 관심을 가지며, 프로젝트의 진행 상황과 결과를 주기적으로 검토합니다.

3. 리스크

3.1 제품 품질 리스크

성능 저하

문제점: 앱이 느리게 작동하거나 응답 시간이 길어질 수 있습니다. 이는 사용자 경험에 부정적인 영향을 미치며, 사용자가 앱을 지속적으로 사용하는 것을 방해할 수 있습니다.
원인: 성능 저하는 주로 비효율적인 코드, 과도한 데이터 처리, 네트워크 지연 등으로 인해 발생할 수 있습니다.
관리 계획: 성능 최적화를 위해 코드 리뷰를 통해 비효율적인 부분을 개선하고, 데이터 처리 및 네트워크 요청을 최적화합니다. 또한, 성능 테스트를 통해 앱의 반응 속도를 지속적으로 모니터링하고 개선합니다.

UI/UX 문제

문제점: 사용자 인터페이스가 직관적이지 않거나 사용하기 어려울 수 있습니다. 이는 사용자가 앱을 이해하고 사용하는 데 어려움을 겪게 하며, 사용자 만족도를 저하시킬 수 있습니다.
원인: UI/UX 문제는 주로 복잡한 인터페이스 디자인, 일관성 부족, 사용자 피드백 미반영 등으로 인해 발생할 수 있습니다.
관리 계획: 사용자 인터페이스 디자인을 간소화하고, 사용자 피드백을 적극적으로 반영하여 UI/UX를 개선합니다. 사용자 테스트를 통해 인터페이스의 직관성을 평가하고, 필요한 경우 디자인을 수정합니다.

3.2 프로젝트 리스크

일정 지연

문제점: 개발 및 테스트 일정이 예상보다 길어질 수 있습니다. 이는 프로젝트의 전체 일정에 영향을 미치며, 앱의 출시가 지연될 수 있습니다.
원인: 일정 지연은 주로 예상치 못한 기술적 문제, 인력 부족, 의사소통 문제 등으로 인해 발생할 수 있습니다.
관리 계획: 프로젝트 관리 도구를 사용하여 일정과 작업 진행 상황을 지속적으로 모니터링합니다. 주기적인 팀 회의를 통해 문제를 조기에 식별하고, 필요한 경우 인력을 추가하거나 작업 우선순위를 조정합니다.

예산 초과

문제점: 개발 비용이 예산을 초과할 수 있습니다. 이는 프로젝트의 재정적 부담을 증가시키며, 추가 자금 조달이 필요할 수 있습니다.

원인: 예산 초과는 주로 예상치 못한 기술적 문제 해결 비용, 인력 추가 비용, 외부 서비스 사용 비용 증가 등으로 인해 발생할 수 있습니다.
관리 계획: 예산을 세부적으로 계획하고, 각 단계별 비용을 지속적으로 모니터링합니다. 비용 절감을 위한 대안을 모색하고, 필요시 투자자와 협력하여 추가 자금을 확보합니다.

4. 테스트 전략

4.1 테스트 설계 기법

기능 테스트

목적: 앱의 각 기능이 예상대로 작동하는지 확인합니다. 사용자 프로필 입력, 재료 기반 레시피 추천, 날씨 기반 메뉴 추천, 즐겨찾기 기능 등 모든 주요 기능을 테스트합니다.
방법: 각 기능에 대한 테스트 케이스를 작성하고, 기능이 올바르게 작동하는지 검증합니다. 예를 들어, 사용자가 입력한 재료에 따라 적절한 레시피가 추천되는지 확인합니다.

성능 테스트

목적: 앱의 반응 속도와 처리 속도를 평가하여, 사용자 경험에 영향을 미치지 않도록 최적화합니다.
방법: 앱의 로딩 시간, 메뉴 추천 출력 시간, 사용자 입력 반응 시간을 측정합니다. 다양한 기기와 네트워크 조건에서 테스트를 수행하여 성능을 평가합니다.

사용자 테스트

목적: 실제 사용자 피드백을 통해 앱의 사용성을 평가하고, 개선점을 찾습니다.
방법: 선택장애가 있는 사용자나 요리를 잘 하지 못하는 사용자를 대상으로 테스트를 수행합니다. 사용자가 앱을 쉽게 이해하고 사용할 수 있는지 확인하고, 사용자 피드백을 바탕으로 UI/UX를 개선합니다.

회귀 테스트

목적: 새로운 기능 추가나 기존 기능 수정 후, 앱의 다른 부분이 영향을 받지 않았는지 확인합니다.
방법: 기존 테스트 케이스를 반복 실행하여, 수정된 부분이 다른 기능에 영향을 미치지 않는지 검증합니다.

예외 처리 테스트

목적: API 호출 실패, 미입력 값 등 예외 상황에서 앱이 적절하게 대응하는지 확인합니다.
방법: 의도적으로 API 호출을 실패하게 하거나 필수 입력 값을 누락시켜, 앱이 사용자에게 적절한 대체 문구나 입력 요청을 제공하는지 테스트합니다.

4.2 테스트 종료 기준

기능적 기준

모든 주요 기능이 예상대로 작동: 사용자 프로필 입력, 재료 기반 레시피 추천, 날씨 기반 메뉴 추천, 즐겨찾기 기능 등 모든 주요 기능이 테스트를 통해 예상대로 작동하는 것이 확인되었을 때 테스트를 종료합니다.

성능 기준

반응 시간 및 처리 속도: 앱의 로딩 시간, 메뉴 추천 출력 시간, 사용자 입력 반응 시간이 설정된 기준(예: 로딩 시간 3초 이내, 추천 메뉴 출력 시간 2초 이내, 사용자 입력 반응 시간 1초 이내)을 충족할 때 테스트를 종료합니다.

사용자 경험 기준

사용자 피드백: 사용자 테스트를 통해 수집된 피드백이 긍정적이며, 사용자가 앱을 쉽게 이해하고 사용할 수 있는 것으로 평가되었을 때 테스트를 종료합니다.

회귀 테스트 기준

기능 수정 후 안정성: 새로운 기능 추가나 기존 기능 수정 후, 회귀 테스트를 통해 다른 기능에 영향을 미치지 않는 것이 확인되었을 때 테스트를 종료합니다.

예외 처리 기준

예외 상황 대응: API 호출 실패, 미입력 값 등 예외 상황에서 앱이 적절하게 대응하는 것이 확인되었을 때 테스트를 종료합니다.

버그 수정 기준

모든 주요 버그 수정 완료: 테스트 과정에서 발견된 모든 주요 버그가 수정되고, 수정된 부분이 재 테스트를 통해 검증되었을 때 테스트를 종료합니다.

4.3 테스트 기능 요구사항

요구사항	테스트 방법
로그인	
로그인 기능	사용자가 로그인을 했을 경우 중복되어있는게 예외처리 됐는지 확인한다. 잘못된 아이디나 비밀번호를 작성했을경우 예외처리 되는지 확인한다.
자동 로그인 기능	앱을 종료하고 다시 앱 실행시 바로 메인화면으로 가는지 확인
비밀번호 찾기 기능	비밀번호 찾기를 할 때 이메일 이 유효한지 확인한다 해당 이메일로 비밀번호를 알려주는 메일이 도착 했는지 확인한다.
아이디 찾기 기능	로그인찾기를 할 때 이메일 이 유효한지 확인한다 해당 이메일로 아이디 를 알려주는 메일이 도착 했는지 확인한다.
회원가입 기능	이메일 형식 비밀번호 길이 및 복합성 등 유효성 검사가 정상적으로 이루어지는지 확인한다. 이메일,아이디,비밀번호 등 필수 입력 필드가 비어있을때 오류 메시지가 표시되는지 확인한다.

요구사항	테스트 방법
메인	
레시피 인기순 추천 기능	공공데이터 API 기반으로 레시피의 인기순으로 정상적으로 추천되는지 확인한다. 레시피의

	사진이 레시피와 매치되게 나오는지 확인한다.
하단 액션바 버튼 기능	하단 액션바의 재료입력,레시피 검색, 사용자 를 눌렀을 경우 화면전환이 이루어지는지 확인한다.
날씨별 레시피 추천 기능	기상청 API가 정상적으로 날씨를 알려주는지 확인한다. 날씨 상황에 맞게 정상적으로 레시피 추천하는지 확인한다.

요구사항	테스트 방법
재료 입력	
상단 재료 추가 하기 버튼 기능	재료 추가하기 를 눌렀을 경우 재료 추가하기 화면으로 이동이 되는지 확인한다.
재료 추가하기 화면에서 검색 기능	재료 추가하기 화면에서 검색을 했을 때 해당 재료가 검색에 보여지는지 확인한다.
재료 추가하기 화면에서 보기 기능	재료 추가하기 화면에서 재료를 보기형식으로 재료사진과 이름이 일치하는지 검색 했을때 해당 재료 사진이 보여지는지 확인한다.
재료가 없거나 있을경우 화면에 재료 사진과 재료 이름이 나오는 기능	재료가 없을경우 화면에 메시지가 나오는지 재료를 추가 했을경우 재료 사진과 이름이 나오는지 확인한다.

요구사항	테스트 방법
레시피 찾기	
레시피 검색 기능	API를 통해 레시피 데이터를 검색을 통해 정상적으로 불러와 지는지 확인 레시피 검색을 했을경우 옳바른 레시피가 검색되는지 확인한다.
레시피 카테고리별 나누는 기능	레시피 카테고리별 한식,중식,양식,일식 으로 나누어서 카테고리에 맞게 레시피 정보가 나오는지 확인한다.
레시피 정보 리스트 형식으로 보여지는 기능	검색 됬을떄 레시피 정보가 올바른지 확인한다.
표시할 때 있는 재료, 없는 재료 등 정보 표시	레시피 찾기를 클릭했을 때 화면에 보유한

해주는 기능	재료가 무엇인지 없는 재료가 무엇인지 구별할 수 있는지 확인한다.

요구사항	테스트 방법
레시피 정보	
레시피 정보 화면 전환 기능	레시피 찾기에서 눌렀을 때 레시피 정보 화면으로 전환이 되는지 확인한다.
레시피 정보 불러들이는 기능	공공데이터 API를 사용해서 레시피 정보(이름, 설명, 몇 인분, 조리 시간, 재료, 도구, 순서)가 정상적으로 화면에 나타나는지 확인한다.
레시피 즐겨찾기 하는 기능	레시피 정보 화면에서 즐겨찾기 아이콘을 눌렀을 때 즐겨찾기 화면 에서 저장이 되는지 확인한다.

요구사항	테스트 방법
사용자 정보	
사용자 이름을 표시하는 기능	사용자가 ID나 구글, 카카오 등 로그인 했을 경우 사용자의 이름이 앱상단에 표시되는지 확인한다.
사용자 정보에서 즐겨찾기 화면으로 전환 하는 기능	사용자 정보에서 즐겨찾기를 눌렀을 때 화면이 정상적으로 전환이 되는지 확인한다.

요구사항	테스트 방법
즐겨찾기	
즐겨찾기한 레시피를 표시하는 기능	사용자 정보에서 레시피를 즐겨찾기 했을때 즐겨찾기 화면에서 올바르게 나타나는지 확인한다.

표시할 때 있는 재료, 없는 재료 등 정보 표시 해주는 기능	즐겨찾기를 클릭했을 때 화면에 보유한 재료가 무엇인지 없는 재료가 무엇인지 구별할 수 있는지 확인한다.
검색할 때 즐겨찾기한 레시피를 찾게 하는 기능	즐겨찾기 에서 내가 검색한 메뉴가 화면에 정확하게 나오는지, 내가 즐겨찾기에 넣어둔 메뉴가 맞는지 확인한다.
즐겨찾기한 레시피를 삭제하는 기능	즐겨찾기한 레시피를 삭제하고 새로고침을 하였을 때 즐겨찾기 내에서 삭제한 메뉴가 사라지는지 확인한다.

4.4 테스트 데이터 요구사항

사용자 프로필 데이터

다양한 사용자 프로필: 성별, 나이, 식습관 등 다양한 사용자 프로필을 기반으로 테스트를 수행합니다. 예를 들어, 다양한 연령대와 식습관을 가진 사용자 프로필을 생성하여 추천 기능의 정확성을 평가합니다.

재료 목록 데이터

다양한 재료 조합: 사용자가 입력할 수 있는 다양한 재료 목록을 준비합니다. 예를 들어, 일반적인 가정에서 자주 사용하는 재료(예: 계란, 밀가루, 우유 등)와 특수 재료(예: 아보카도, 퀴노아 등)를 포함하여 레시피 추천 기능을 테스트합니다.

날씨 데이터

다양한 날씨 조건: 기상청 API를 통해 수집한 다양한 날씨 데이터를 기반으로 테스트를 수행합니다. 예를 들어, 맑은 날, 비오는 날, 더운 날, 추운 날 등 다양한 날씨 조건에서 메뉴 추천 기능의 적절성을 평가합니다.

로그인 및 회원가입 데이터

사용자 계정 정보: 다양한 사용자 계정 정보를 생성하여 로그인 및 회원가입 기능을 테스트합니다. 예를 들어, 구글 계정, 카카오 계정 등을 사용하여 로그인 기능의 안정성을 평가합니다.

예외 처리 데이터

예외 상황 데이터: API 호출 실패, 미입력 값 등 예외 상황을 시뮬레이션하기 위한 데이터를 준비합니다. 예를 들어, 의도적으로 API 호출을 실패하게 하거나 필수 입력 값을 누락시켜 예외 처리 기능을 테스트합니다.

4.5 테스트 환경 요구사항

하드웨어 요구사항

Android 기기: 최소 Android 8.0 이상 버전을 지원하는 스마트폰이 필요합니다. 다양한 기기에서 테스트를 수행하여 앱의 호환성을 평가합니다.
다양한 기기 모델: 삼성, LG, Google Pixel 등 다양한 제조사의 기기에서 테스트를 수행하여 기기별 성능 차이를 평가합니다.

네트워크 요구사항

인터넷 연결: 앱의 기능이 외부 API와 데이터베이스에 접근하여 실시간으로 데이터를 수집하고 처리하기 위해 안정적인 인터넷 연결이 필요합니다.
다양한 네트워크 조건: Wi-Fi, 4G, 5G 등 다양한 네트워크 조건에서 테스트를 수행하여 네트워크 상태에 따른 앱의 성능을 평가합니다.

소프트웨어 요구사항

운영체제 버전: Android 8.0 이상 버전에서 테스트를 수행하여 최신 운영체제 환경에서 앱의 기능을 평가합니다.
API 연동: 기상청 API, Firebase, 공공데이터 API 등과의 원활한 연동을 위해 필요한 소프트웨어 환경을 설정합니다.

테스트 도구

테스트 자동화 도구: 앱의 기능 테스트를 자동화하기 위해 필요한 도구를 사용합니다. 예를 들어, Appium, Espresso 등을 사용하여 테스트를 자동화합니다.
성능 모니터링 도구: 앱의 성능을 모니터링하기 위해 필요한 도구를 사용합니다. 예를 들어, Firebase Performance Monitoring을 사용하여 앱의 반응 속도와 처리 속도를 평가합니다.

4.5 테스트 도구 요구사항

기능 테스트 도구

Appium: 모바일 애플리케이션의 기능 테스트를 자동화하기 위한 도구입니다. 다양한 기기와 운영체제에서 테스트를 수행할 수 있으며, UI 요소의 작동 여부를 검증합니다.
Espresso: Android 앱의 UI 테스트를 자동화하기 위한 도구입니다. 사용자 인터페이스의 반응성과 사용성을 평가하는 데 사용됩니다.

성능 테스트 도구

Firebase Performance Monitoring: 앱의 성능을 모니터링하고 분석하기 위한 도구입니다. 앱의 로딩 시간, 메뉴 추천 출력 시간, 사용자 입력 반응 시간을 측정하여 성능을 최적화합니다.
JMeter: 네트워크 성능 테스트를 수행하기 위한 도구입니다. 다양한 네트워크 조건에서 앱의 성능을 평가하고, 네트워크 요청의 처리 속도를 분석합니다.

사용자 테스트 도구

UserTesting: 실제 사용자 피드백을 수집하고 분석하기 위한 도구입니다. 사용자가 앱을 사용하는 동안의 경험을 기록하고, 개선점을 찾습니다.

예외 처리 테스트 도구

Postman: API 호출을 테스트하고, 예외 상황을 시뮬레이션하기 위한 도구입니다. API 호출 실패 시 앱의 대응 방식을 검증합니다.

4.7 테스트 우선순위

평균 점수 7~9 : H(높음), 4~6: M(보통), 1~3 : L(낮음)

요구사항	우선순위	우선순위 분석				
		사용빈도	영향도	개발자 미흡도	요구사항 중요도	평균
	로그인					
로그인 기능	H	8	8	4	8	7
자동 로그인 기능	M	7	5	3	5	5
비밀번호 찾기 기능	L	2	2	6	5	3.9
아이디 찾기 기능	L	2	2	6	5	3.9
회원가입 기능	H	9	9	7	9	8.5

요구사항	우선순위	우선순위 분석				
		사용빈도	영향도	개발자 미흡도	요구사항 중요도	평균
	메인					
레시피 인기순 추천 기능	M	6	5	6	5	5.5
하단 액션바 버튼 기능	M	8	7	3	7	6.1
날씨별 레시피 추천 기능	M	5	5	8	5	5.9

요구사항	우선순위	우선순위 분석				
		사용빈도	영향도	개발자 미흡도	요구사항 중요도	평균
	재료 입력					
상단 재료 추가 하기 버튼 기능	M	7	8	3	8	6.5
재료 추가하기 화면에서 검색 기능	M	6	7	5	7	6.25
재료 추가하기 화면에서 보기 기능	M	5	6	4	5	5
재료 사진과 이름을 보여주는 기능	M	7	7	5	8	6.7

요구사항	우선순위	우선순위 분석				
		사용빈도	영향도	개발자 미흡도	요구사항 중요도	평균
	레시피 찾기					
레시피 검색 기능	H	8	7	7	7	7.25
레시피 카테고리 별 나누는 기능	H	9	9	6	6	7.5
레시피 정보를 보여주는 기능	H	8	7	6	9	7.5
보유한 재료를 표시해주는 기능	M	8	6	5	7	6.5

요구사항	우선순위	우선순위 분석

		사용빈도	영향도	개발자 미흡도	요구사항 중요도	평균
	레시피 정보					
레시피 정보를 불러들이는 기능	M	7	6	5	6	6
레시피 즐겨찾기 하는 기능	M	5	4	5	3	4.2

요구사항	우선순위	우선순위 분석				
		사용빈도	영향도	개발자 미흡도	요구사항 중요도	평균
	사용자 정보					
사용자 이름을 표시하는 기능	L	3	3	1	3	2.5

요구사항	우선순위	우선순위 분석				
		사용빈도	영향도	개발자 미흡도	요구사항 중요도	평균
	즐겨찾기					
즐겨찾기 한 레시피를 표시하는 기능	M	7	6	5	7	6.2
표시할 때	M	6	6	3	6	5.2

있는 재료, 없는 재료 등 정보 표시 해주는 기능						
검색할 때 즐겨찾기 한 레시피를 찾게 하는 기능	H	7	7	7	8	7.2
즐겨찾기 한 레시피를 삭제하는 기능	M	5	5	4	6	5

4.8 테스트 산출물

테스트 계획서

내용: 테스트의 목표, 범위, 방법, 일정 등을 상세히 설명하는 문서입니다. 테스트 계획서는 테스트 활동의 방향을 설정하고, 이해 관계자에게 테스트 전략을 명확히 전달합니다.

테스트 케이스 문서

내용: 각 기능에 대한 테스트 케이스를 상세히 설명하는 문서입니다. 테스트 케이스 문서는 테스트 수행 시 필요한 입력 데이터, 예상 결과, 테스트 절차 등을 포함합니다.

테스트 결과 보고서

내용: 테스트 수행 후 결과를 기록한 보고서입니다. 테스트 결과 보고서는 각 테스트 케이스의 성공 여부, 발견된 문제점, 수정 상태 등을 포함하며, 앱의 품질을 평가하는 데 사용됩니다.

버그 목록

내용: 테스트 과정에서 발견된 버그를 기록한 목록입니다. 버그 목록은 각 버그의 상세 설명, 발생 조건, 우선순위, 수정 상태 등을 포함하며, 개발팀이 문제를 해결하는 데 도움을 줍니다.

성능 분석 보고서

내용: 앱의 성능을 분석한 보고서입니다. 성능 분석 보고서는 앱의 로딩 시간, 메뉴 추천 출력 시간, 사용자 입력 반응 시간 등을 측정하여 성능 최적화의 방향을 설정합니다.

사용자 피드백 보고서

내용: 사용자 테스트를 통해 수집된 피드백을 기록한 보고서입니다. 사용자 피드백 보고서는 앱의 사용성을 평가하고, 개선점을 찾는 데 사용됩니다.

4.9 재 테스트와 리그레션 테스트

재 테스트

목적: 수정된 기능이나 버그가 올바르게 해결되었는지 확인합니다. 재 테스트는 특정 문제를 해결한 후, 해당 기능이 예상대로 작동하는지 검증하는 데 중점을 둡니다.

절차:

수정된 기능에 대한 테스트 케이스를 다시 실행하여, 문제 해결 여부를 확인합니다.

수정된 부분이 다른 기능에 영향을 미치지 않는지 검증합니다.

재 테스트 결과를 기록하고, 필요한 경우 추가 수정 작업을 수행합니다.

리그레션 테스트

목적: 새로운 기능 추가나 기존 기능 수정 후, 앱의 다른 부분이 영향을 받지 않았는지 확인합니다. 리그레션 테스트는 전체 시스템의 안정성을 보장하는 데 중점을 둡니다.

절차:

기존 테스트 케이스를 반복 실행하여, 수정된 부분이 다른 기능에 영향을 미치지 않는지 검증합니다.

모든 주요 기능이 예상대로 작동하는지 확인합니다.

리그레션 테스트 결과를 기록하고, 필요한 경우 추가 수정 작업을 수행합니다.

테스트 주기

주기: 재 테스트와 리그레션 테스트는 기능 수정 및 추가 후, 주기적으로 수행됩니다. 특히, 주요 릴리스 전에는 반드시 리그레션 테스트를 통해 전체 시스템의 안정성을 검증합니다.

5. 테스트 규모 산정

시간 산정

기능 테스트: 각 기능에 대한 테스트 케이스 작성 및 실행에 필요한 시간을 평가합니다. 예를 들어, 사용자 프로필 입력, 재료 기반 레시피 추천, 날씨 기반 메뉴 추천, 즐겨찾기 기능 등 주요 기능에 대한 테스트 시간을 산정합니다.
성능 테스트: 앱의 반응 속도와 처리 속도를 평가하는 데 필요한 시간을 산정합니다. 다양한 기기와 네트워크 조건에서 테스트를 수행하여 성능을 최적화합니다.
사용자 테스트: 실제 사용자 피드백을 수집하고 분석하는 데 필요한 시간을 평가합니다. 사용자 테스트는 선택장애가 있는 사용자나 요리를 잘 하지 못하는 사용자를 대상으로 수행됩니다.
회귀 테스트: 기능 수정 및 추가 후, 기존 기능이 영향을 받지 않았는지 확인하는 데 필요한 시간을 산정합니다.

인력 산정

개발자: 기능 개발 및 수정 작업을 수행하는 개발자의 수와 역할을 평가합니다.
테스터: 테스트 케이스 작성, 실행, 결과 분석을 수행하는 테스터의 수와 역할을 평가합니다.
디자이너: 사용자 인터페이스 디자인 및 개선 작업을 수행하는 디자이너의 수와 역할을 평가합니다.

자원 산정

테스트 도구: 테스트 자동화 도구, 성능 모니터링 도구, 사용자 테스트 도구 등 필요한 도구의 수와 비용을 평가합니다.
기기 및 환경: 다양한 기기 모델과 네트워크 조건에서 테스트를 수행하기 위한 자원을 평가합니다.

예산 산정

총 비용: 테스트 활동에 필요한 총 비용을 평가합니다. 인력 비용, 도구 비용, 기기 및 환경 비용 등을 포함하여 예산을 산정합니다.

6. 인력

6.1 업무와 책임

개발자

기능 구현: 앱의 주요 기능(예: 사용자 프로필 입력, 재료 기반 레시피 추천, 날씨 기반 메뉴 추천, 즐겨찾기 기능 등)을 설계하고 구현합니다.
버그 수정: 테스트 과정에서 발견된 버그를 수정하고, 기능의 안정성을 보장합니다.
코드 리뷰: 팀 내 다른 개발자와 협력하여 코드 리뷰를 수행하고, 코드 품질을 유지합니다.
성능 최적화: 앱의 성능을 최적화하여 사용자 경험을 개선합니다.

테스터

테스트 케이스 작성: 각 기능에 대한 테스트 케이스를 작성하고, 테스트 계획에 따라 테스트를 수행합니다.
테스트 실행 및 결과 분석: 테스트를 실행하고, 결과를 분석하여 발견된 문제점을 기록합니다.
버그 리포트 작성: 발견된 버그를 상세히 기록하고, 개발팀에 전달하여 수정 작업을 지원합니다.
회귀 테스트 수행: 기능 수정 및 추가 후, 기존 기능이 영향을 받지 않았는지 확인합니다.

디자이너

UI/UX 디자인: 사용자 인터페이스를 설계하고, 사용자가 쉽게 이해하고 사용할 수 있도록 UX를 최적화합니다.
프로토타입 제작: Figma 등 디자인 도구를 사용하여 앱의 프로토타입을 제작하고, 개발팀과 협력하여 구현을 지원합니다.
사용자 피드백 반영: 사용자 테스트를 통해 수집된 피드백을 반영하여 디자인을 개선합니다.

프로젝트 매니저

프로젝트 계획 및 관리: 프로젝트의 전체 계획을 수립하고, 일정과 자원을 관리합니다.
팀 커뮤니케이션: 팀 내 커뮤니케이션을 조율하고, 이해 관계자와의 소통을 담당합니다.
리스크 관리: 프로젝트 진행 중 발생할 수 있는 리스크를 식별하고, 대응 전략을 수립합니다.

7. 일정 계획

7.1 일정

설계 단계

기간: 7월~8월
활동:
UI/UX 디자인: 사용자 인터페이스를 설계하고, Figma를 사용하여 제작합니다.
기술 스택 결정: 앱 개발에 필요한 기술 스택(Java, Firebase, Git 등)을 결정합니다.

개발 단계
기간: 8월~9월말
활동:
기능 구현: 사용자 프로필 입력, 재료 기반 레시피 추천, 날씨 기반 메뉴 추천, 즐겨찾기 기능 등 주요 기능을 구현합니다.
코드 리뷰: 팀 내 코드 리뷰를 통해 코드 품질을 유지하고, 성능 최적화를 수행합니다.
초기 테스트: 구현된 기능에 대한 초기 테스트를 수행하여 안정성을 검증합니다.

테스트 단계
기간: 10월~11월
활동:
기능 테스트: 모든 주요 기능이 예상대로 작동하는지 확인합니다.
성능 테스트: 앱의 반응 속도와 처리 속도를 평가하여 최적화합니다.
사용자 테스트: 실제 사용자 피드백을 수집하고, 사용성을 평가합니다.
회귀 테스트: 기능 수정 및 추가 후, 기존 기능이 영향을 받지 않았는지 확인합니다.

배포 단계
기간: 11월 ~ 11월 중순
활동:
최종 검토: 모든 테스트가 완료되고, 발견된 문제점이 수정되었는지 확인합니다.
앱 배포 준비: 앱을 Google Play 스토어에 배포할 준비를 합니다.
사용자 지원 계획: 앱 출시 후 사용자 지원을 위한 계획을 수립합니다.

마일스톤
설계 완료: 설계 단계 종료 시, 모든 요구사항과 디자인이 확정됩니다.
기능 구현 완료: 개발 단계 종료 시, 모든 주요 기능이 구현됩니다.
테스트 완료: 테스트 단계 종료 시, 모든 테스트가 완료되고, 앱의 안정성이 검증됩니다.
앱 출시: 배포 단계 종료 시, 앱이 공식적으로 출시됩니다.

7.2 중단과 재개 기준

중단 기준

주요 기능의 심각한 결함 발견: 테스트 과정에서 주요 기능에 심각한 결함이 발견되어 사용자 경험에 큰 영향을 미칠 경우, 프로젝트를 일시 중단하고 문제 해결에 집중합니다.
예산 초과: 프로젝트 진행 중 예산이 예상보다 크게 초과되어 추가 자금 조달이 필요할 경우, 프로젝트를 일시 중단하고 재정적 계획을 재조정합니다.
일정 지연: 프로젝트 일정이 예상보다 크게 지연되어 주요 마일스톤을 달성할 수 없을 경우, 프로젝트를 일시 중단하고 일정 계획을 재조정합니다.
기술적 문제: 예상치 못한 기술적 문제가 발생하여 프로젝트 진행이 불가능할 경우, 프로젝트를 일시 중단하고 문제 해결을 위한 기술적 지원을 요청합니다.

재개 기준

결함 수정 완료: 발견된 결함이 수정되고, 재 테스트를 통해 안정성이 검증되었을 때 프로젝트를 재개합니다.
추가 자금 확보: 필요한 추가 자금이 확보되어 재정적 계획이 재조정되었을 때 프로젝트를 재개합니다.
일정 재조정 완료: 일정 계획이 재조정되어 주요 마일스톤을 달성할 수 있는 상태가 되었을 때 프로젝트를 재개합니다.
기술적 문제 해결: 기술적 문제가 해결되고, 프로젝트 진행이 가능한 상태가 되었을 때 프로젝트를 재개합니다.