1. 요구사항 정의서 (Requirements Specification Document - RSD)

1.1 기능 요구사항 (Functional Requirements)

* 사용자 관리:
  + 앱 최초 실행 시 사용자 이름을 입력받고 UserPreferences에 저장해야 합니다.
  + 사용자 이름이 저장되어 있으면 MainMenuScreen으로 즉시 이동해야 합니다.
* 언어 및 학습 모드 선택:
  + 사용자는 타갈로그어 또는 세부아노어 중 학습 언어를 선택할 수 있어야 합니다.
  + 사용자는 문장 학습 또는 플래시카드 학습 모드를 선택할 수 있어야 합니다.
  + 플래시카드 학습 모드에서는 초급, 중급, 고급 레벨을 선택할 수 있어야 합니다.
* 학습 코어 기능:
  + assets/sentences\_final.csv에서 학습 문장 데이터를 로드하여 사용할 수 있어야 합니다.
  + assets/audio/에 있는 오디오 파일을 재생하여 문장/단어의 올바른 발음을 들려줄 수 있어야 합니다.
  + speech\_to\_text를 사용하여 사용자의 발음을 인식할 수 있어야 합니다.
  + PronunciationService를 사용하여 인식된 발음과 정답 발음의 유사도를 평가하고 점수를 제공해야 합니다.
  + 학습 결과에 따라 사용자에게 즉각적인 피드백(예: 정답/오답, 유사도 점수)을 제공해야 합니다.
* 데이터 관리 및 보고:
  + DBHelper를 사용하여 학습 세션 정보 및 개별 학습 결과를 SQLite 데이터베이스에 저장해야 합니다.
  + ReportService를 사용하여 학습 결과를 CSV 파일로 생성할 수 있어야 합니다.
  + 저장된 세션 목록을 SessionListScreen에서 조회할 수 있어야 합니다.
  + 특정 세션의 상세 보고서를 ReportScreen에서 조회할 수 있어야 합니다.

1.2 비기능 요구사항 (Non-Functional Requirements)

* 성능: 음성 인식 및 평가 과정은 지연 없이 실시간에 가까운 응답 속도를 보여야 합니다 (목표: 1초 이내).
* 확장성: 향후 학습 언어 및 콘텐츠 추가가 용이하도록 시스템이 설계되어야 합니다.
* 보안: 사용자 이름 및 학습 결과 데이터는 로컬 저장소에 안전하게 저장되어야 합니다.
* 사용성: 직관적이고 쉬운 사용자 인터페이스를 통해 모든 연령대의 사용자가 쉽게 학습할 수 있어야 합니다.
* 호환성: iOS 및 Android 모바일 운영체제에서 원활하게 작동해야 합니다.

2. 아키텍처 설계 문서 (Architecture Design Document - ADD)

2.1 시스템 개요 및 레이어 구조

앱은 Flutter의 일반적인 레이어 구조를 따르며, 서비스 지향 아키텍처를 채택하여 UI와 비즈니스 로직을 분리합니다.

* UI 레이어 (Screens): 사용자와 상호작용하는 화면. StatefulWidget 또는 상태 관리 패키지를 사용하여 상태를 관리하고 UI를 업데이트합니다.
* 서비스 레이어 (Services): 비즈니스 로직 및 외부 시스템(DB, 파일 시스템, 음성 인식 API 등)과의 상호작용을 처리합니다.
* 데이터 레이어 (Models): 앱의 데이터 구조를 정의합니다.

2.2 주요 컴포넌트 및 상호작용

* main.dart: 앱 초기 진입점 및 사용자 이름 기반 초기 라우팅.
* UserPreferences: shared\_preferences를 이용한 설정 관리.
* DBHelper: SQLite 데이터베이스 접근 및 세션/결과 데이터 관리.
* PronunciationService: 발음 전처리 및 유사도 비교.
* ReportService: CSV 보고서 생성.
* LearningScreen (FlashcardLearningScreen, SentenceLearningScreen):
  + UI 레이어에서 speech\_to\_text를 사용하여 음성 입력.
  + audioplayers를 사용하여 음성 출력.
  + csv를 사용하여 학습 데이터 로드.
  + PronunciationService로 발음 평가.
  + DBHelper로 학습 결과 저장.

2.3 기술 스택

* 프론트엔드: Flutter (Dart)
* 상태 관리: StatefulWidget 또는 Provider/Riverpod (향후 도입)
* 로컬 데이터베이스: SQLite (sqflite)
* 로컬 설정: shared\_preferences
* 음성 인식: speech\_to\_text
* 오디오 재생: audioplayers

3. 데이터베이스 설계 문서 (Database Design Document - DDD)

3.1 SQLite 데이터베이스 스키마

DBHelper.dart에 정의된 대로, SQLite 데이터베이스는 sessions 테이블과 results 테이블을 가집니다.

* sessions 테이블
  + id: INTEGER, PRIMARY KEY, AUTOINCREMENT (세션 고유 ID)
  + user\_name: TEXT (사용자 이름)
  + selected\_language: TEXT (선택된 언어)
  + learning\_mode: TEXT (학습 모드: 'Sentence' or 'Flashcard')
  + level: TEXT (플래시카드 레벨: 'Beginner', 'Intermediate', 'Advanced')
  + session\_date: TEXT (세션 날짜)
  + session\_time: TEXT (세션 시간)
* results 테이블
  + id: INTEGER, PRIMARY KEY, AUTOINCREMENT (결과 고유 ID)
  + session\_id: INTEGER (세션 ID, sessions 테이블의 id를 참조하는 FOREIGN KEY)
  + sentence\_id: TEXT (학습된 문장 또는 단어 ID)
  + correctness: REAL (발음 유사도 점수)

3.2 데이터 무결성 및 관계

* results 테이블의 session\_id는 sessions 테이블의 id와 1:N 관계를 가집니다. 이는 results 테이블에 FOREIGN KEY(session\_id) REFERENCES sessions(id) 제약 조건으로 구현됩니다.

4. API 명세서 (API Specification)

이 앱은 주로 로컬 데이터와 서비스를 활용하며, 백엔드 API는 없습니다. 대신, 내부적으로 사용되는 서비스 모듈의 인터페이스를 명세합니다.

4.1 PronunciationService

* 기능: 발음 전처리 및 유사도 계산.
* preprocess(String text):
  + 설명: 입력된 문자열을 발음 유사성 기반으로 전처리합니다. (예: 오오와 오 통일)
  + 입력: String text (음성 인식 결과 또는 정답 발음)
  + 출력: String (전처리된 문자열)
* compareLevenshtein(String str1, String str2):
  + 설명: 두 문자열 간의 레벤슈타인 거리를 기반으로 유사도를 0.0에서 1.0 사이의 점수로 계산합니다.
  + 입력: String str1, String str2
  + 출력: double (유사도 점수)
* isPronunciationSimilar(String str1, String str2):
  + 설명: 유사도 점수가 0.7 이상인지 여부를 판단합니다.
  + 입력: String str1, String str2
  + 출력: bool

4.2 DBHelper

* 기능: 학습 세션 및 결과 데이터베이스 작업.
* insertSession(Map<String, dynamic> data): 세션 정보를 DB에 삽입.
* insertResult(Map<String, dynamic> data): 학습 결과를 DB에 삽입.
* getSessions(): 모든 세션 목록 조회.
* getResultsForSession(int sessionId): 특정 세션의 학습 결과 조회.

4.3 ReportService

* 기능: CSV 보고서 생성.
* generateCsvReport(List<LearningResult> results, ...): 학습 결과 리스트를 기반으로 CSV 파일 생성.

5. 개발 표준 및 가이드라인

5.1 코딩 스타일

* Dart 스타일 가이드 준수: Flutter 팀에서 권장하는 공식 Dart 코딩 스타일을 따릅니다.
* Linting: flutter\_lints 규칙을 활성화하고 모든 경고 및 에러를 해결합니다.
* 명명 규칙:
  + 클래스, Enum: PascalCase (MainMenuScreen)
  + 변수, 함수, 파일명: camelCase (userName, \_saveUserName, db\_helper.dart)
  + 상수: SCREAMING\_SNAKE\_CASE (필요시)

5.2 Flutter 위젯 가이드라인

* 관심사 분리: UI(위젯)는 비즈니스 로직(서비스)과 분리되어야 합니다.
* 상태 관리: 동적인 데이터 변화가 필요한 화면(LearningScreen,)은 StatefulWidget으로 구현하거나, Provider와 같은 상태 관리 패턴을 사용합니다.
* const 사용: 가능한 모든 Widget 인스턴스에 const를 사용하여 성능을 최적화합니다.

5.3 Git 커밋 메시지 규칙

* 형식: type(scope): subject (예: feat(learning): add speech recognition integration)
* Type: feat(기능 추가), fix(버그 수정), docs(문서 변경), style(코드 스타일), refactor(리팩토링), test(테스트 코드), chore(기타).

5.4 에러 처리 및 로깅

* 모든 비동기 작업(DBHelper, UserPreferences, File I/O)에는 try-catch 블록을 포함하여 예외를 처리하고 적절한 사용자 피드백 또는 로깅을 수행합니다.
* 개발 단계에서는 print()나 debugPrint()를 사용하여 로그를 남기지만, 프로덕션 배포 전에는 삭제하거나 logger 패키지를 사용합니다.

6. 테스트 계획서 (Test Plan)

6.1 테스트 범위

* 단위 테스트 (Unit Test): PronunciationService, DBHelper, UserPreferences 등 서비스 로직의 정확성 테스트.
* 위젯 테스트 (Widget Test): screens 디렉토리의 각 화면 위젯이 의도한 대로 렌더링되고 사용자 상호작용에 응답하는지 테스트.
* 통합 테스트 (Integration Test): 앱의 주요 사용자 흐름(로그인 -> 언어 선택 -> 학습 모드 -> 학습 진행 -> 보고서 확인)이 전체적으로 올바르게 작동하는지 테스트.

6.2 테스트 환경

* 개발 환경: Flutter SDK, Dart SDK
* 도구: Flutter flutter\_test, mockito (필요 시)

6.3 주요 테스트 시나리오

* 사용자 관리: 이름 입력 및 shared\_preferences에 성공적으로 저장되는지 확인.
* 데이터베이스: DBHelper를 통해 세션 및 결과가 성공적으로 삽입되고 조회되는지 확인.
* 발음 평가: PronunciationService의 compareLevenshtein 및 preprocess 함수가 올바른 유사도 점수를 반환하는지 확인.
* 학습 화면:
  + CSV 파일 로딩이 정상적으로 이루어지는지.
  + 오디오 재생이 정상 작동하는지.
  + speech\_to\_text를 통해 음성이 성공적으로 인식되는지.
  + 발음 평가 결과에 따라 UI가 동적으로 업데이트되는지.

7. 배포 및 운영 가이드 (Deployment & Operations Guide)

7.1 빌드 및 배포 절차

1. 빌드: flutter build apk (Android) 또는 flutter build ipa (iOS) 명령어를 사용하여 프로덕션 빌드 생성.
2. 테스트: 테스트 계획서에 따라 QA 진행.
3. 스토어 업로드: Google Play Console 또는 Apple App Store Connect에 빌드 업로드.

7.2 운영 환경 설정

* 권한 관리: 음성 인식(speech\_to\_text) 및 로컬 파일 저장(path\_provider)을 위해 Android 및 iOS 권한 설정 (AndroidManifest.xml, Info.plist)이 올바르게 되어 있는지 확인.

7.3 모니터링 및 유지보수

* 크래시 리포팅: Firebase Crashlytics 또는 Sentry와 같은 도구를 연동하여 앱 크래시를 실시간으로 모니터링하고 추적.
* 정기 업데이트: 에러 수정 및 성능 개선을 포함한 정기적인 앱 업데이트를 계획.

## 8. 사용자 설명서 (User Documentation)

이 문서는 "StudyApp" 사용자를 위한 안내서입니다.

### 8.1 StudyApp 소개

"StudyApp"은 필리핀어(타갈로그어, 세부아노어)를 효과적으로 학습하고 발음 연습을 할 수 있도록 돕는 교육 앱입니다. 문장 학습과 플래시카드 학습 모드를 통해 재미있게 언어를 익히고, 자신의 학습 기록을 체계적으로 관리할 수 있습니다.

### 8.2 주요 기능

* **사용자 이름 설정**: 나만의 학습 프로필을 만들고 학습 데이터를 관리합니다.
* **언어 선택**: 타갈로그어 또는 세부아노어 중 학습하고 싶은 언어를 선택합니다.
* **학습 모드 선택**:
  + **문장 학습**: 다양한 문장을 듣고 따라 말하며 발음을 연습합니다.
  + **플래시카드 학습**: 단어나 구문을 플래시카드 형태로 학습하고 발음을 교정합니다.
* **발음 평가**: 사용자의 발음을 인식하고 원어민 발음과 비교하여 유사도를 알려줍니다.
* **학습 기록 보기**: 과거 학습 세션의 기록을 확인하고 학습 성과를 추적할 수 있습니다.
* **학습 결과 보고서**: 학습 결과가 CSV 파일로 생성되어 자세한 분석이 가능합니다.

### 8.3 앱 사용 방법

#### 8.8.1 첫 시작 (이름 입력)

1. 앱을 처음 실행하면, 사용하실 이름을 입력하는 화면이 나타납니다.
2. 원하는 이름을 입력하고 **"저장"** 버튼을 누르세요. 이 이름은 학습 기록 관리에 사용됩니다.

#### 8.8.2 메인 메뉴

이름 입력 또는 재접속 후에는 메인 메뉴 화면이 나타납니다.

* **"학습 시작"**: 새로운 학습 세션을 시작합니다.
* **"학습 기록 보기"**: 이전 학습 기록을 확인합니다.
* **"앱 정보"**: 앱에 대한 기본 정보를 확인합니다.

#### 8.8.3 학습 시작하기

1. 메인 메뉴에서 **"학습 시작"** 버튼을 누릅니다.
2. **언어 선택**: 타갈로그어와 세부아노어 중 학습하고 싶은 언어를 선택합니다.
3. **학습 모드 선택**: "문장 학습" 또는 "플래시카드 학습" 중 하나를 선택합니다.
4. **플래시카드 학습 시 레벨 선택**: "플래시카드 학습"을 선택했다면, 초급, 중급, 고급 중 원하는 학습 레벨을 선택합니다.

#### 8.8.4 학습 진행 (문장/플래시카드 학습 화면)

* 화면에 제시된 문장이나 단어를 확인합니다.
* (미래 기능: 원어민 발음 듣기 버튼을 눌러 정확한 발음을 들어보세요.)
* (미래 기능: 마이크 버튼을 누르고 제시된 문장을 큰 소리로 따라 말해보세요.)
* (미래 기능: 앱이 사용자의 발음을 인식하고, 정답 발음과의 유사도를 %로 표시합니다. "정답!" 또는 "발음이 조금 달라요"와 같은 피드백을 통해 발음을 교정할 수 있습니다.)
* (미래 기능: 다음 문장/플래시카드로 이동하는 버튼을 눌러 계속 학습하세요.)

#### 8.8.5 학습 기록 확인하기

1. 메인 메뉴에서 **"학습 기록 보기"** 버튼을 누릅니다.
2. 이전 학습 세션 목록이 날짜와 함께 표시됩니다.
3. 특정 세션을 선택하면 해당 세션의 학습 결과 보고서가 나타납니다. (미래 기능: 보고서 내용을 자세히 확인하고 CSV 파일로 저장하거나 공유할 수 있습니다.)

### 8.4 문제 해결 (Troubleshooting)

* **음성 인식이 잘 안 돼요**:
  + 앱에 마이크 사용 권한이 부여되었는지 확인해주세요. (휴대폰 설정 > 앱 > StudyApp > 권한)
  + 주변이 조용한 곳에서 명확하게 발음해보세요.
  + 인터넷 연결 상태가 안정적인지 확인해주세요 (일부 음성 인식 기능은 인터넷 연결이 필요할 수 있습니다).
* **앱이 갑자기 종료돼요**: 앱을 다시 시작해 보세요. 문제가 계속 발생하면, 최신 버전으로 업데이트되었는지 확인하거나 개발자에게 문의해주세요.

### 8.5 문의 및 피드백

앱 사용 중 문의사항이나 개선 아이디어가 있다면 언제든지 알려주세요. 여러분의 소중한 의견은 앱 개선에 큰 도움이 됩니다.