**4.2: Data Dictionary**

**Sub-topics:**

1. **Database Structure**
   * Tables for user profiles, language content, progress tracking, and AI model data.
   * Relationships between tables (e.g., User → Progress Tracking → Lessons).
2. **Fields and Attributes**
   * Fields for each table (e.g., User\_ID, Name, Preferred\_Language, Progress\_Level, etc.).
   * Data types and constraints for each field (e.g., User\_ID as Primary Key, Language\_Code as VARCHAR).
3. **Keys and Relationships**
   * Primary and foreign keys for tables.
   * Referential integrity constraints.
4. **Sample Data**
   * Example entries for key tables to showcase how data is stored.
5. **Indexing and Optimization**
   * Indexed columns for faster search and retrieval.
   * Database normalization level.

**4.3: Program Description**

**Sub-topics:**

1. **Overall Functionality**
   * Overview of how the app operates, including AI-based recommendations and language lessons.
2. **Modules and Components**
   * Explanation of each module:
     + **User Module**: Account creation, login, and preferences.
     + **Lesson Module**: Language content delivery.
     + **AI Recommendation Module**: Personalized content suggestions.
     + **Progress Tracker**: Tracks user learning progress.
3. **Algorithms and Logic**
   * AI-based logic for language learning suggestions.
   * Algorithms for evaluating quizzes and pronunciation assessments.
4. **Technology Stack**
   * Frontend: Android Studio.
   * Backend: Firebase/SQL Server.
   * AI: TensorFlow/Keras for machine learning.
5. **Integration and API Usage**
   * APIs used (e.g., Google Translate API, Speech-to-Text API).
   * Communication between modules.

**4.4: Naming Conventions**

**Sub-topics:**

1. **General Naming Standards**
   * Consistent use of camelCase for variables and methods.
   * PascalCase for classes.
2. **Database Naming**
   * Table names: Singular (e.g., User, Lesson, Progress).
   * Column names: Descriptive (e.g., Lesson\_Title, User\_Email).
3. **Code Naming**
   * Functions: Verb-noun structure (e.g., getUserProgress, loadLessonContent).
   * Constants: Uppercase with underscores (e.g., MAX\_ATTEMPTS, API\_KEY).
4. **UI Elements**
   * Activity and fragment IDs: Prefix activity\_ or fragment\_.
   * Widget IDs: Prefix by type (e.g., btn\_submit, txt\_username).
5. **File Naming**
   * Activity files: ActivityNameActivity.java.
   * XML layouts: activity\_name.xml.

**4.5: Validations**

**Sub-topics:**

1. **User Input Validations**
   * Registration fields (e.g., email format, strong password criteria).
   * Name and username validation (e.g., alphanumeric characters only).
2. **Data Integrity Validations**
   * Prevent duplicate user accounts using unique constraints.
   * Ensure proper language codes for lessons.
3. **AI and Learning Content Validations**
   * Validation of AI responses against predefined thresholds.
   * Content integrity checks during updates.
4. **Error Handling and Feedback**
   * Descriptive error messages for invalid inputs.
   * Retry mechanisms for failed API calls.
5. **Security Validations**
   * SQL injection prevention.
   * Secure storage of sensitive data (e.g., encrypted passwords).