



PROGRAMMING FOR MOBILE DEVICES**CCS21503****ASSIGNMENT 3****BACK-END PROGRAMMING MOBILE INTERFACE DESIGN & PROTOTYPE
DEVELOPMENT**



Received Date : 29/9/2024
Submission Date : 12/12/2024
Weightage : 30%
Semester : September 2024
Lecturer : Dr. JAMAL ABDULLAHI NUH

Instruction to students:

- This is a GROUP assignment.
- Complete this cover sheet and attach it to your assignment (first page).

Student declaration:***I declare that:***

- *This assignment is my own work*
- *I understand what is meant by plagiarism*
- *My lecturer has the right to deduct my marks in the case of:*
 - *Late submission*
 - *Any plagiarism found in my assignment.*

Name	Student ID	Photo
ALFI ZAHRA NISYA SEKEDANG	012021090880	
MUHAMMAD NAIM MUSTAQIM BIN MD NIZAR	012022070592	

MARKS:	Total

TABLE OF CONTENT

INTRODUCTION.....	4
PROJECT GOALS OVERVIEW.....	4
FRONT-END SECTION.....	5
UI COMPONENTS.....	5
BACK-END SECTION.....	12
BACK-END ARCHITECTURE, TECHNOLOGIES USED & FRONT-END COMMUNICATION	
12	
DATA FLOW.....	12
CHALLENGES AND SOLUTIONS.....	14
CONCLUSION.....	15

LANGUAGE LEARNING APP

INTRODUCTION

In an increasingly globalized world, mastering new languages has become essential for personal and professional growth. This project focuses on developing an innovative language learning app designed to make language acquisition engaging, efficient, and accessible. By incorporating interactive quizzes, dynamic flashcards, and a personalized progress tracker, the app provides a comprehensive platform for learners to build and refine their language skills. Through a user-friendly interface and adaptive learning features, the app aims to cater to diverse learning styles and keep users motivated throughout their language-learning journey.

PROJECT GOALS OVERVIEW

The primary goals of the project include enhancing user engagement, promoting knowledge retention, and providing clear insights into progress. Interactive quizzes test users' understanding of vocabulary, grammar, and pronunciation in a fun and challenging way. Flashcards serve as an effective tool for memory reinforcement, offering customizable sets tailored to individual needs. Meanwhile, the progress tracker empowers learners to monitor their development, celebrate milestones, and identify areas for improvement. Together, these features aim to create a holistic language learning experience that is both enjoyable and results-driven.

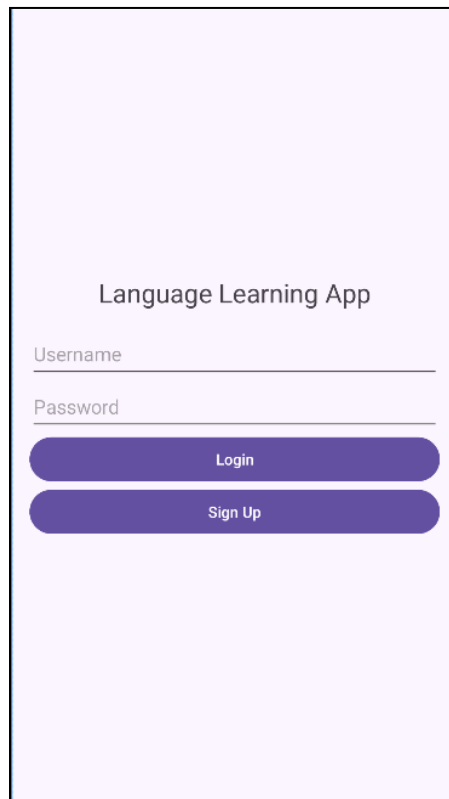
FRONT-END SECTION

This section discusses the design approach, tools and justification for the particular user interface (UI) components created for Language Learning App. The front-end design aims to make the application aesthetically pleasing, user-friendly and effective in carrying out its tasks.

- Tools used: XML for layout, Kotlin for activity logic and Android's native libraries for smooth interaction.
- Design: Clean and minimalistic design with large buttons and text for readability.
- UI Components:
 - RecyclerView for listing lessons.
 - ProgressBar to visually track learning progression.
 - TextView and Buttons to interact with flashcards.

UI COMPONENTS

1. Login Screen - activity_login.xml:



Code Snippet:

```
<EditText
    android:id="@+id/etUsername"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Username" />

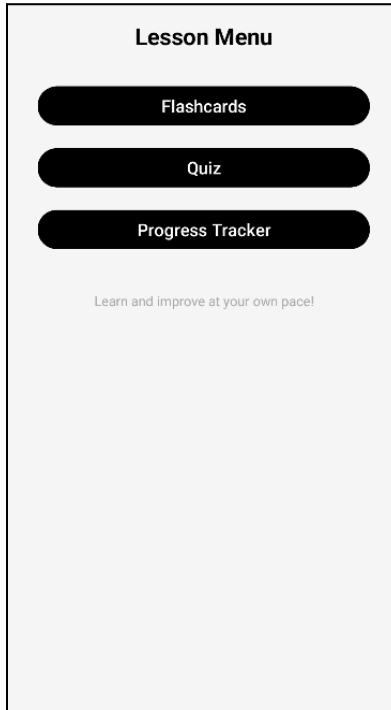
<EditText
    android:id="@+id/etPassword"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Password" />

<Button
    android:id="@+id/btnLogin"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Login" />
```

Explanation:

- This code defines the layout for the login screen.
- The LinearLayout arranges the child views (username field, password field and buttons) vertically.
- Two EditText fields allow the user to input their username and password, while the Button elements handle user actions for login and sign-up.
- The android:hint attributes provide placeholders in the input fields.

2. activity_lesson.xml:



Code Snippet:

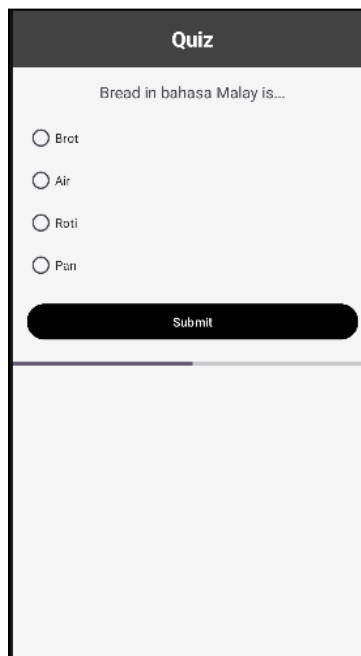
```
<Button
    android:id="@+id/btnFlashcards"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="16dp"
    android:backgroundTint="@color/teal_200"
    android:text="Flashcards"
    android:textColor="@android:color/white" />

<Button
    android:id="@+id/btnQuiz"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="16dp"
    android:backgroundTint="@color/purple_200"
    android:text="Quiz"
    android:textColor="@android:color/white" />
```

Explanation:

- These buttons are part of the Lesson screen and allow navigation to Flashcards and Quiz screens.
- Buttons are styled with backgroundTint to make them visually distinct.
- Margins and padding are applied to ensure a clean and organized UI layout.

3. activity_quiz.xml:



Code Snippet:

```
<TextView
    android:id="@+id/tvQuestion"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Bread in Bahasa Malay is..."
    android:gravity="center" />

<RadioGroup
    android:id="@+id/radioGroupOptions"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
```



```
<RadioButton
    android:id="@+id/radioOption1"
    android:text="Paris" />

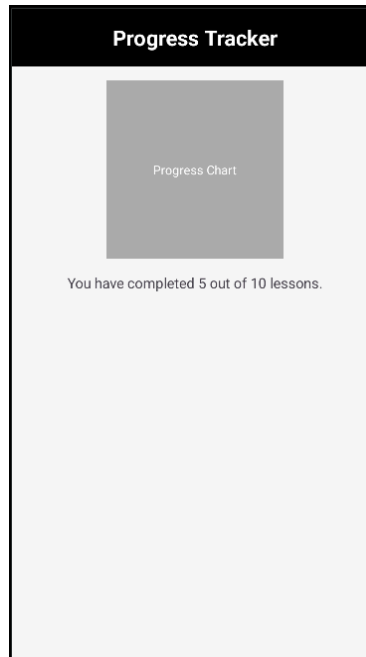
<RadioButton
    android:id="@+id/radioOption2"
    android:text="Roti" />
</RadioGroup>

<Button
    android:id="@+id/btnSubmitQuiz"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Submit" />
```

Explanation:

- TextView displays the quiz question.
- RadioGroup contains multiple-choice answers.
- Buttons allow the user to submit their answer.

4. activity_progress_tracker.xml:

**Code Snippet:**

```
<TextView
    android:id="@+id/tvProgressSummary"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="You have completed 5 out of 10 lessons."
    android:gravity="center" />

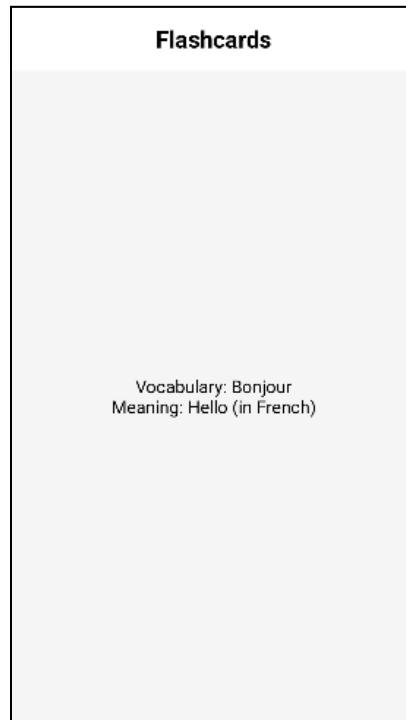
<ProgressBar
    android:id="@+id/progressBar"

style="@android:style/Widget.DeviceDefault.Light.ProgressBar.Horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:progress="50"
    android:max="100" />
```

Explanation:

- TextView displays a summary of user progress
- ProgressBar visually represents the user's learning progress.

5. activity_flashcards.xml:



Code Snippet:

```
<TextView
    android:id="@+id/tvFlashcard"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Sample Flashcard: Bonjour - Hello"
    android:gravity="center" />
```

Explanation:

- This TextView acts as a flashcard, displaying a vocabulary word and its meaning.
- The background attribute applies a custom drawable for rounded corners and padding.
- gravity="center" ensures that the text is centered within the view.

BACK-END SECTION

BACK-END ARCHITECTURE, TECHNOLOGIES USED & FRONT-END COMMUNICATION

Back-End Architecture:

- The back-end for the Language Learning App is structured using a lightweight approach, as the app primarily focuses on local functionality rather than server-based operations.
- It uses Java as the core programming language to manage logic and interactions between screens.

Technologies Used:

- Java: Handles core logic for navigation, user authentication and data updates.
- Android SDK: Provides APIs for interacting with user inputs, navigation and UI components.
- SharedPreferences: Temporarily stores user data like login credentials and progress without needing a server or database.
- Intents: Facilitates communication between front-end screens (activities) and back-end operations.

Front-End to Back-End Communication:

- The front-end communicates with the back end through activity lifecycles and direct method calls in Java.
- Example:
 - A button in the front end triggers a method (onClick) in the back end.
 - Data entered in UI components (e.g, username, password) is passed to the back end via EditText objects, validated and used for navigation or logic processing.

DATA FLOW

Since the application doesn't currently use a remote database or APIs, the data flow is simple and localized. Here's how the data flows:

Login/Sign-Up Process:

1. Input: The user enters their username and password into the EditText fields.
2. Validation: The entered values are compared with predefined credentials (hard coded or stored in SharedPreferences).
3. Response:
 - If the credentials match, the user is directed to the Lesson screen.
 - If the credentials don't match, an error message is displayed using a Toast.

Flashcards Feature:

1. Initialization: The flashcard text is hard coded or dynamically loaded into a TextView at runtime.
2. User Interaction: The user views the text on the screen (e.g, vocabulary, grammar tips).

Quiz Process:

1. Display question: The back end retrieves and displays a hard coded question and multiple-choice answers in the UI.
2. Validate Answer: When the user submits their answer, the back end checks the selected radio buttons against the correct answer.
3. Response: Displays a message indicating whether the answer is correct or incorrect.

Progress Tracker:

1. Calculate Progress: The back end calculates progress as a percentage based on the number of lessons completed.
2. Update UI: Updates the ProgressBar and a summary TextView in the UI dynamically.

CHALLENGES AND SOLUTIONS

Challenge 1: Managing Navigation Across Screens

- Problem: Ensuring smooth navigation between activities (e.g, login to lesson) while maintaining data consistency.
- Solution: Used Intent objects to transition between screens and passed any required data directly through Intent extras. The finish() method was used to prevent returning to the login screen after a successful login.

Challenge 2: Simplifying Authentication Logic

- Problem: Initial attempts at using SharedPreferences for user credentials required complex validation logic.
- Solution: Simplified the logic by hardcoding default credentials for testing purposes and storing temporary data in SharedPreferences. Future iterations can integrate real databases or authentication APIs.

Challenge 3: Dynamic Flashcard Updates

- Problem: Managing multiple flashcards without a swipeable interface was tedious and lacked user engagement.
- Solution: Created a simplified design where each flashcard is displayed as a single TextView. Dynamic updates were implemented using hardcoded values for simplicity.

Challenge 4: Visual Representation of Progress

- Problem: Initial attempts at creating a progress tracker were inconsistent and didn't provide clear feedback to the user.
- Solution: Implemented a combination of ProgressBar and TextView to show both a visual and textual summary of progress. Hardcoded percentages were used for demonstration.

CONCLUSION

The development of this language learning app represents a significant step toward making language acquisition more accessible, engaging, and effective for users. By addressing common challenges such as maintaining user motivation, managing content delivery, and ensuring reliable progress tracking, the app offers a seamless and user-friendly experience. Features like quizzes, flashcards, and personalized progress tracking not only make learning enjoyable but also empower users to achieve their goals at their own pace. With continuous updates and a focus on user feedback, the app is well-positioned to adapt to evolving needs and remain a valuable tool for learners worldwide.

RUBRIC FOR ASSIGNMENT 3: BACK-END PROGRAMMING & MOBILE INTERFACE DEVELOPMENT

Criteria	1-2 (Standards Not Met)	3-4 (Standards Barely Met)	5-6 (Meets Standards)	7-8 (Exceeds Standards)	9-10 (Outstanding)	Marks
Introduction (5)	Lacks clarity or relevance to the project goals.	Objectives vaguely mentioned with minimal project context.	Provides an adequate overview with some detail on goals.	Clear objectives, covering front and back-end development in context.	Highly detailed introduction with clear project goals and outcomes.	
Front-End Design (5)	UI design is incomplete, poorly structured, and not functional.	Minimal design with little attention to usability and consistency.	Functional UI, but lacks visual appeal or consistency.	Well-designed UI, user-friendly, responsive, and meets project requirements.	Exceptionally designed UI with excellent usability, responsiveness, and consistency.	
Back-End Functionality (10)	Basic or incomplete back-end with little or no CRUD functionality.	Some back-end functionality is present, but incomplete or faulty CRUD operations.	Functional back-end with all CRUD operations, but minor issues exist.	Full back-end implementation with working CRUD operations and minor issues.	Outstanding back-end with well-structured, flawless CRUD operations and additional features (e.g., user authentication).	
Data Handling & CRUD Operations (30)	No clear data management, incomplete CRUD methods, or poorly functioning.	CRUD methods implemented but lack optimization or clarity in handling data.	CRUD functions are correctly implemented but need improvements in structure or efficiency.	Clear and effective implementation of CRUD methods, ensuring efficient data handling.	Highly organized, efficient CRUD functions with clear separation of concerns, optimal data handling.	
Authentication System (5)	No authentication implemented or non-functional.	Basic authentication exists but has	Authentication implemented but may have minor	Fully functional user authentication	Secure and well-implemented authentication system	

		security issues or lacks key features.	issues (e.g., insecure, limited error handling).	(signup, login) with minor flaws.	with all required features and proper error handling.	
Back-End Integration (10)	No proper integration between the front and back end, or only partially working.	Integration is present but lacks efficiency or consistency.	Basic integration between the front and back end with some functionality issues.	Seamless integration of front-end and back-end with good functionality.	Excellent integration, with smooth communication between front-end and back-end, using best practices.	
Report Structure & Clarity (5)	Report is disorganized, lacks details, and is hard to follow.	Report is partially structured but lacks clarity in key sections.	Adequately structured report, covering key sections but lacking in-depth analysis.	Well-structured and clear report, covering all required sections with appropriate depth.	Exceptionally clear, well-organized, and in-depth report, covering all sections with comprehensive detail.	
Screenshots & Documentation (10)	No screenshots or poorly documented.	Screenshots are present but poorly explained.	Screenshots included with explanations but lacking detail.	Clear screenshots with explanations, well-documented process.	Detailed documentation with relevant, well-explained screenshots throughout.	
Challenges & Solutions (5)	No challenges mentioned, or no solutions provided.	Challenges mentioned but solutions are unclear or ineffective.	Some challenges described with adequate solutions.	Challenges well-documented with clear and practical solutions.	Comprehensive breakdown of challenges with innovative and effective solutions.	
Prototype Submission (5)	Project files are incomplete or not functional.	Project files are submitted but contain errors or lack functionality.	Project files are functional with some minor issues.	Complete project files, fully functional and meeting requirements.	All project files submitted are well-organized, fully functional, and exceed requirements.	
Team Member Evaluation (10%)						
1) Task Completion						

- 2) Quality of Work
- 3) Able to explain clearly
- 4) Problem-Solving & Technical skills
- 5) Explain things using Android terms

Total Marks: 100 (Weight 30%)

Members	Score
ALFI ZAHRA NISYA SEKEDANG	
MUHAMMAD NAIM MUSTAQIM BIN MD NIZAR	