RANCANG BANGUN APLIKASI LATIHAN *TEST OF ENGLISH AS A FOREIGN LANGUAGE* (TOEFL) BERBASIS *MOBILE* DENGAN INTEGRASI WORKFLOW n8n

TUGAS AKHIR

Diajukan Oleh:

MAUDY AKMAL

NIM. 210705114

Mahasiswa Fakultas Sains dan Teknologi

Program Studi Teknologi Informasi

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI AR-RANIRY

BANDA ACEH

2026 M / 1447 H

LEMBAR PERSETUJUAN

RANCANG BANGUN APLIKASI LATIHAN TOEFL (*TEST OF ENGLISH AS A FOREIGN LANGUAGE*) BERBASIS *MOBILE* DENGAN INTEGRASI WORKFLOW n8n

TUGAS AKHIR

Diajukan kepada Fakultas Sains dan Teknologi UIN AR-Raniry Banda Aceh
Sebagai Salah Satu Persyaratan Penulisan Tugas akhir dalam Prodi Teknologi Informasi

Oleh:
MAUDY AKMAL
210705114
Mahasiswa Fakultas Sains dan Teknologi
Program Studi Teknologi Informasi

Disetujui Untuk Diseminarkan Oleh:

Pembimbing I	Pembimbing II
Mulkan Fadhli, M.T	Malahayati, M.T
NIP.198811282020121006	NIP. 198301272015032003

Mengetahui
Ketua Program Studi Teknologi Informasi

Malahayati, M T
NIP. 198301272015032003

LEMBAR PENGESAHAN

RANCANG BANGUN APLIKASI LATIHAN *TEST OF ENGLISH AS A FOREIGN LANGUAGE* (TOEFL) BERBASIS *MOBILE* DENGAN INTEGRASI WORKFLOW n8n

TUGAS AKHIR

Telah Diuji Oleh Panitia Ujian Munaqasyah Tugas Akhir
Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh dan Dinyatakan Lulus
Serta Diterima Sebagai Salah Satu Beban Studi Program Sarjana (S-1)
Dalam Ilmu Teknologi Informasi

Pada Hari/Tanggal: Selasa, 3 Februari 2026
15 Sya'ban 1447 H

Di Darussalam, Banda Aceh

Panitia Ujian Munaqasyah Tugas Akhir :

Ketua,	Sekretaris,
Mulkan Fadhli, M.T.	**Malahayati. M.T.**
NIP. 198811282020121006	**NIP. 198301272015032003**
Penguji I,	Penguji II,
Dr. Hendri Ahmadian, S.Si., M.I.M.	**Nizam Albar, ST., M.T**
NIP. 198301042014031002	

Mengetahui:
Dekan Fakultas Sains dan Teknologi
UIN Ar-Raniry Banda Aceh,

Prof. Dr. Ir. Muhammad Dirhamsyah, M.T., IPU
NIP. 196210021988111001

LEMBAR PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama	: Maudy Akmal
NIM	: 210705114
Program Studi	: Teknologi Informasi
Fakultas	: Sains dan Teknologi
Judul Tugas Akhir	: Rancang Bangun Aplikasi Test of English As A Foreign Languange (TOEFL) Berbasis Mobile Dengan Integrasi Workflow n8n

Dengan ini menyatakan bahwa dalam penulisan tugas akhir ini, saya:

1. Tidak menggunakan ide orang lain tanpa mampu mengembangkan dan mempertanggung jawabkan.
2. Tidak melakukan plagiasi terhadap naskah orang lain.
3. Tidak menggunakan karya orang lain tanpa menyebutkan sumber asli atau tanpa izin pemilik karya.
4. Tidak memanipulasi dan memalsukan data.
5. Mengerjakan sendiri karya ini dan mampu bertanggung jawab atas karya ini.

Bila dikemudian hari ada tuntutan dari pihak lain atas karya saya, dan telah melalui pembuktian yang dapat dipertanggung jawabkan dan ternyata memang ditemukan bukti bahwa saya telah melanggar pernyataan ini, maka saya siap dikenakan sanksi berdasarkan aturan yang berlaku di Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh.

Demikian Pernyataan ini saya buat dengan sesungguhnya dan tanpa paksaan dari pihak manapun.

Banda Aceh, 03 Februari 2026

Yang Menyatakan

(Maudy Akmal)

ABSTRAK

Nama	: Maudy Akmal
Nim	: 210705114
Program Studi	: Teknologi Informasi
Judul	: Rancang Bangun Aplikasi Latihan Test Of English As A Foreign Language (Toefl) Berbasis Mobile Dengan Integrasi Workflow N8n
Tanggal Sidang	: 3 Februari 2026 / 15 Sha'ban, 1447 H
Jumlah Halaman	: 96 Halaman
Pembimbing I	: Mulkan Fadhli, S.T., M.T.
Pembimbing II	: Malahayati, M.T.
Kata Kunci	: Aplikasi *Mobile*, TOEFL, Kotlin, n8n, Otomatisasi *Workflow*, *Middleware,* Kecerdasan Buatan.

Keterbatasan akses media latihan dan tingginya biaya kursus konvensional menjadi kendala utama bagi calon peserta ujian TOEFL. Penelitian ini bertujuan merancang dan membangun aplikasi latihan TOEFL berbasis Android yang adaptif dengan integrasi *workflow* n8n sebagai *middleware* untuk otomatisasi pembuatan soal berbasis kecerdasan buatan (*Artificial Intelligence*). Sistem dikembangkan menggunakan metode *Waterfall*, di mana sisi *client* dibangun menggunakan bahasa Kotlin dengan *framework* Jetpack Compose serta basis data lokal Room untuk mendukung penggunaan secara *online* maupun *offline*. Pengujian fungsionalitas sistem dilakukan secara *end-to-end* menggunakan pendekatan *Black-Box Testing*. Hasil penelitian menunjukkan bahwa aplikasi berhasil menyajikan fitur simulasi ujian (*Listening, Structure, Reading*) secara dinamis. Integrasi n8n dan AI sukses mendukung fitur "Latihan Pintar" yang responsif terhadap input pengguna. Berdasarkan pengujian *Minimum Viable Product* (MVP), seluruh fitur utama aplikasi terbukti berjalan sangat baik, mudah digunakan, dan memperoleh penilaian rata-rata di atas skor 4.

ABSTRACT

Name	: Maudy Akmal
Nim	: 210705114
Department	: *Information Technology*
Title	: *Design and Development of a Mobile-Based TOEFL (Test of English as a Foreign Language) Practice Application Integrated with n8n Workflows*
Date	: *3 February 2026 / 15 Sha'ban, 1447 H*
Thesis Pages	: *96 Pages*
Supervisor I	: Mulkan Fadhli, S.T., M.T.
Supervisor II	: Malahayati, M.T.
Keywords	: *Mobile Application, TOEFL, Kotlin, n8n, Workflow Automation, Middleware, Artificial Intelligence.*

The limited accessibility of practice media and the high cost of conventional preparation courses remain the primary obstacles for TOEFL candidates. This research aims to design and develop an adaptive Android-based TOEFL practice application, integrating n8n workflow as a middleware to automate Artificial Intelligence (AI)-driven question generation. The system was developed using the Waterfall method, where the client side was built using the Kotlin programming language with the Jetpack Compose framework, alongside a Room local database to support both online and offline functionality. End-to-end system functionality testing was conducted using the Black-Box Testing approach. The results demonstrate that the application successfully delivers dynamic test simulation features covering Listening, Structure, and Reading. The integration of n8n and AI effectively supports the "Latihan Pintar" (Smart Practice) feature, which is responsive to user inputs. Based on the Minimum Viable Product (MVP) evaluation, all core features of the application proved to function exceptionally well, offer ease of use, and achieve an average satisfaction score above 4.

KATA PENGANTAR

بسم الله الرحمن الرحيم

Assalamu'alaikum Wr. Wb

Alhamdulillahirabbil'alamin, puja dan puji serta syukur kita ucapkan kehadirat Allah SWT, berkat nikmat dan karunia-Nya yang indah yang masih kita rasakan sampai pada saat ini, nikmat berupa iman, Islam, kesehatan, kesempatan, pengetahuan yang tentunya masih banyak lagi nikmat yang tidak dapat dijabar di atas seluruh kertas ini.

Dalam kesempatan ini penulis bersyukur kepada Allah SWT, berkat Ridho-Nya penulis mampu merampungkan proposal skripsi yang berjudul *"*Rancang Bangun Aplikasi Latihan *Test Of English As A Foreign Language* (*TOEFL*) Berbasis *Mobile* Dengan Integrasi *Workflow* n8n*"*.

Proposal skripsi ini disusun sebagai kewajiban penulis guna melengkapi tugas dan syarat untuk menyelesaikan pendidikan Strata-I (S1) Program Studi Teknologi Informasi Universitas Islam Negeri Ar-Raniry Banda Aceh, serta memperoleh gelar Sarjana Komputer Universitas Islam Negeri Ar-Raniry Banda Aceh.

Dalam proses penyusunan proposal ini, penulis telah mendapatkan banyak bantuan, bimbingan, dan dukungan dari berbagai pihak. Oleh karena itu, dengan segala hormat dan rasa Syukur, penulis menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Allah SWT yang telah memberikan nikmat, rahmat dan kemudahan dalam pengerjaan Tugas Akhir ini.
2. Yang teristimewa, Ayahanda Drs. martunis Alsa dan Ibunda Haniyah, serta keluarga besar penulis, khususnya saudara Yerni Hanimar, Rizky Herlianda dan Gema Maulida Silva, yang senantiasa menjadi sumber kekuatan, cinta, dan doa yang tiada henti. Semoga Allah senantiasa melimpahkan rahmat, kesehatan, dan kebahagiaan kepada mereka, serta menjadikan segala amal mereka sebagai jalan menuju ridha dan cinta-Nya.

3. Bapak Mulkan Fadhli, S.T., M.T. Selaku dosen pembimbing I Tugas Akhir yang telah dengan sabar membimbing, memberikan saran, kritik, dan masukan berharga selama proses penyusunan tugas akhir ini.
4. Ibu Malahayati, M.T. selaku pembimbing II Tugas Akhir yang telah banyak meluangkan waktu serta fikiran dalam memberikan bantuan, kritik, dan masukan berharga untuk membimbing penulis demi kesempurnaan Tugas Akhir ini.
5. Ketua dan Sekretaris Program Studi Teknologi Informasi, Ibu Malahayati, M.T dan Bapak Khairan Ar, M.Kom, serta Bapak dan Ibu dosen Program Studi Teknologi Informasi yang telah memberikan ilmu pengetahuan dalam bidang Teknologi Informasi.
6. Pembimbing Akademik, bapak Mulkan Fadhli, S.T., M.T. yang telah membimbing dan memberikan saran selama masa perkuliahan.
7. Staf Prodi Ibu Cut Ida Rahmadiana, S.Si., yang telah membantu penulis dalam pengurusan administrasi dan surat-menyurat.
8. Nazura Friza Alpinka, terima kasih karena sudah setia menemani, memberikan dukungan, dan motivasi kepada penulis sehingga penelitian Tugas Akhir ini dapat diselesaikan dengan baik.
9. Teman-teman seperjuangan saya, khususnya M. Dolyanda harialdy, Dani Rizqullah, Anggil Maulana dan Mindarina, terima kasih atas kebersamaan, dukungan, dan semangat yang telah kalian berikan sepanjang perjalanan perkuliahan ini.

Peneliti menyadari bahwa Tugas Akhir ini masih jauh dari kata sempurna. Oleh karena itu, kritik dan saran yang membangun selalu peneliti harapkan, demi penyusunan Tugas Akhir yang baik. Peneliti berharap semoga tulisan ini dapat bermanfaat bagi perkembangan ilmu pengetahuan.

Banda Aceh, 10 Februari 2026

Maudy Akmal

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Bahasa Inggris merupakan bahasa internasional yang memiliki peran strategis dalam dunia pendidikan, pekerjaan, dan komunikasi global.(Siregar, 2023) Kemampuan berbahasa Inggris sering menjadi persyaratan utama dalam penerimaan beasiswa, melanjutkan pendidikan ke jenjang yang lebih tinggi, serta memasuki dunia kerja, baik di dalam maupun luar negeri. Salah satu instrumen yang digunakan secara luas untuk mengukur kemampuan bahasa Inggris adalah *Test of English as a Foreign Languag*e (TOEFL), yang menguji kemampuan peserta dalam aspek *listening*, structure, dan *reading comprehen*sion (Sakina & Khofifah, 2025).

Dalam praktiknya, banyak calon peserta TOEFL menghadapi berbagai kendala dalam mempersiapkan diri menghadapi ujian tersebut. Kendala yang umum ditemui antara lain keterbatasan akses terhadap media latihan, materi latihan yang kurang bervariasi, serta biaya kursus persiapan TOEFL yang relatif tinggi dan membutuhkan waktu khusus untuk mengikuti kelas. Kondisi ini menyebabkan proses persiapan TOEFL menjadi kurang efektif dan tidak merata bagi seluruh calon peserta.

Perkembangan teknologi informasi, khususnya teknologi *mobile*, memberikan peluang untuk mengatasi permasalahan tersebut. Aplikasi *mobile* memungkinkan proses pembelajaran dilakukan secara fleksibel, praktis, dan mandiri karena dapat diakses kapan saja dan di mana saja (Shabrina Ziha Fidela et al., 2023). aplikasi *mobile* memiliki keunggulan berupa kemudahan akses serta dukungan penggunaan secara *online* maupun *offline*, sehingga sesuai untuk digunakan sebagai media latihan TOEFL yang berkelanjutan.

Selain fleksibilitas akses, aplikasi *mobile* juga memungkinkan pengelolaan data latihan secara dinamis. Materi dan soal latihan dapat diperbarui serta disesuaikan dengan kebutuhan pengguna tanpa harus bergantung sepenuhnya pada metode pembelajaran konvensional. Dengan demikian, pengembangan aplikasi latihan TOEFL berbasis Android menggunakan bahasa pemrograman Kotlin menjadi solusi untuk menyediakan media latihan yang efektif dan efisien.

Lebih lanjut, pemanfaatan *Artificial Intelligence* (AI) dalam aplikasi mobile membuka peluang untuk meningkatkan kualitas latihan TOEFL. Integrasi AI melalui *workflow* n8n memungkinkan sistem untuk menghasilkan soal latihan secara otomatis, menganalisis pola kesalahan pengguna, serta memberikan rekomendasi latihan atau remedial yang sesuai(Widodo et al., 2024). Pendekatan ini diharapkan dapat membantu pengguna berlatih secara lebih terarah sesuai dengan kemampuan dan kebutuhan masing-masing.

Berdasarkan penelusuran pada Google Play Store dengan kata kunci "Latihan TOEFL" dan "TOEFL", ditemukan beberapa aplikasi sejenis, seperti TOEFL Academy dan TOEFL Prep & Practice App. Selain itu, terdapat pula penelitian terkait pengembangan aplikasi latihan TOEFL, di antaranya penelitian oleh Zainul Hasan (2024) berjudul "Implementasi Aplikasi TOEFL Test Pro Berbasis Android" serta penelitian oleh Astuti dan Yuli Yana (2024) berjudul "Rancang Bangun Aplikasi Simulasi Tes TOEFL Berbasis Website (Studi Kasus: Program Studi Teknik Informatika Universitas Mataram)".

Namun, aplikasi dan penelitian tersebut masih memiliki beberapa keterbatasan. Keterbatasan yang ditemukan antara lain jumlah soal yang terbatas, analisis hasil latihan yang masih sederhana, belum tersedianya dukungan penggunaan secara offline, serta minimnya fitur latihan yang bersifat adaptif dan interaktif. Kondisi ini menunjukkan bahwa kebutuhan akan aplikasi latihan TOEFL yang mampu menyediakan data latihan secara dinamis, mendukung penggunaan online dan offline, serta memanfaatkan AI secara optimal masih belum terpenuhi.

Berdasarkan permasalahan tersebut, penelitian ini berfokus pada perancangan dan pembangunan aplikasi latihan TOEFL berbasis Android menggunakan Kotlin yang dapat digunakan secara online dan offline dengan pengelolaan data yang dinamis. Selain itu, penelitian ini juga mengintegrasikan kecerdasan buatan melalui workflow n8n untuk membantu proses latihan TOEFL, khususnya dalam menghasilkan soal secara otomatis dan memberikan rekomendasi remedial berdasarkan pilihan dan hasil latihan pengguna.

Dengan demikian, penelitian ini mengangkat judul "Rancang Bangun Aplikasi Latihan TOEFL Berbasis Mobile dengan Integrasi Workflow n8n". Penelitian ini bertujuan untuk mengembangkan aplikasi latihan TOEFL berbasis

mobile yang praktis dan terjangkau, menerapkan integrasi AI untuk mendukung proses latihan secara adaptif, serta menyediakan dukungan mode offline agar proses pembelajaran tetap dapat berlangsung tanpa ketergantungan pada jaringan internet.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan oleh penulis maka rumusan masalah dari penelitian ini adalah sebagai berikut:

1. Bagaimana merancang dan membangun aplikasi latihan TOEFL berbasis Android menggunakan Kotlin, serta dapat digunakan secara online dan offline dengan data yang dinamis
2. Bagaimana cara mengintegrasikan *Artificial Intelligence* (AI) untuk membantu proses latihan TOEFL

1.3 Tujuan Penelitian

Berdasarkan latar belakang dan juga rumusan masalah yang telah diuraikan oleh penulis maka tujuan dari penelitian ini adalah sebagai berikut:

1. Mengembangkan aplikasi latihan TOEFL berbasis Android menggunakan Kotlin yang menyediakan latihan secara variatif dan dinamis.
2. Menerapkan integrasi AI melalui workflow n8n untuk menghasilkan soal otomatis dan memberi rekomendasi remedial sesuai kebutuhan pengguna.

1.4 Manfaat penelitian

Berdasarkan latar belakang, rumusan masalah dan juga tujuan penelitian yang telah diuraikan oleh penulis maka manfaat dari penelitian ini yaitu :

1. Penelitian ini diharapkan dapat memberikan kontribusi terhadap pengembangan literatur dan praktik dalam bidang teknologi pembelajaran bahasa, khususnya mengenai pemanfaatan *Artificial Intelligence* (AI) melalui *workflow n8n* dalam pembuatan soal dan analisis kemampuan pengguna.
2. Aplikasi yang dihasilkan dapat menjadi sarana alternatif yang efektif, efisien, dan terjangkau bagi calon peserta TOEFL dalam mempersiapkan ujian, tanpa harus bergantung pada kursus berbiaya tinggi.

3. penelitian ini diharapkan dapat meningkatkan aksesibilitas dan pemerataan kesempatan belajar bahasa Inggris, terutama bagi masyarakat dengan keterbatasan waktu, biaya, maupun akses jaringan internet.

1.5 Batasan Penelitian

Untuk memastikan kesesuaian pembahasan penelitian ini dengan judul dan latar belakang yang telah diuraikan, penulis memutuskan untuk mempersempit cakupan masalah yang akan dibahas dalam penelitian ini dengan menetapkan batasan sebagai berikut:

1. Penelitian ini hanya berfokus pada pengembangan aplikasi latihan TOEFL berbasis mobile pada Android menggunakan bahasa pemrograman Kotlin.
2. Penerapan AI dalam penelitian ini digunakan untuk menghasilkan soal otomatis dan memberikan variasi materi latihan.
3. Mode offline dalam aplikasi dibatasi pada penggunaan paket materi dan bank soal yang dapat diunduh, tidak mencakup fitur AI yang akan di buat.

BAB II
TINJAUAN PUSTAKA

2.1 Bahasa Inggris

Bahasa Inggris saat ini bukan sekadar materi sekolah, melainkan sebuah alat komunikasi global yang penting. Menurut (Tauhid & Lubis, 2024) menguasai bahasa Inggris menjadi sangat penting di era globalisasi karena memungkinkan seseorang menjangkau pengetahuan dan peluang internasional.

Meski begitu, meskipun bahasa Inggris diajarkan dari SD hingga perguruan tinggi, masih banyak mahasiswa yang merasa belum mampu menggunakan bahasa Inggris dengan baik dalam praktiknya (seperti berbicara atau menulis). Hal ini menunjukkan bahwa pengajaran bahasa Inggris perlu diarahkan tidak hanya pada teori, tetapi juga penggunaan nyata di lingkungan sosial (Zulfania Arrahma et al., 2022).

Dalam konteks penguasaan bahasa Inggris, khususnya untuk menghadapi tes standar internasional seperti TOEFL, strategi pembelajaran yang efektif sangat dibutuhkan. Penelitian (Zulfania Arrahma et al., 2022) menegaskan bahwa media pembelajaran yang interaktif dapat membantu meningkatkan pemahaman kosakata dan keterampilan berbahasa secara menyenangkan. Dalam hal ini, pemanfaatan teknologi digital, khususnya aplikasi mobile, menjadi solusi yang tepat karena mampu menyediakan latihan secara mandiri, fleksibel, dan berulang. juga menunjukkan bahwa keterbatasan lingkungan belajar dapat diatasi melalui aplikasi berbasis mobile yang memberi akses latihan bahasa Inggris kapan saja dan di mana saja. Dengan demikian, pengembangan aplikasi latihan TOEFL berbasis Kotlin diharapkan dapat menjadi sarana pembelajaran yang praktis, terstruktur, serta mampu menjawab kebutuhan mahasiswa dalam meningkatkan kemampuan bahasa Inggris untuk keperluan akademik maupun profesional.

2.2 *Test of English as a Foreign Language* (TOEFL)

TOEFL adalah tes standar internasional yang dirancang untuk mengukur kemampuan bahasa Inggris seseorang yang bukan penutur asli. Tes ini sering dipakai oleh universitas dan lembaga di berbagai negara untuk menentukan

kesiapan akademik calon mahasiswa asing. Menurut (Sutrisno et al., 2023) TOEFL merupakan salah satu tes kecakapan bahasa Inggris yang paling banyak digunakan di Indonesia, baik untuk keperluan akademik maupun profesional. Dalam laporan resmi ETS, dijelaskan bahwa sejak awal pengembangannya, tim ahli dan penguji dari bidang linguistik dan evaluasi (psychometrics) berupaya memastikan bahwa TOEFL mampu mengukur kompetensi komunikatif dalam konteks akademik, bukan hanya kemampuan struktural bahasa.

Struktur tes TOEFL modern (umumnya versi iBT) meliputi empat aspek penting: mendengarkan (listening), membaca (reading), berbicara (speaking), dan menulis (writing). Dalam setiap bagian, peserta dihadapkan pada tugas-tugas yang menuntut pemahaman dan produksi bahasa dalam kondisi yang menyerupai konteks akademik. Penelitian evaluatif terhadap TOEFL ITP menyatakan bahwa tanggapan peserta (test-takers) penting sebagai bagian dari bukti validitas: misalnya, menunjukkan bahwa masukan dari peserta uji turut digunakan untuk memperbaiki desain dan validitas tes berbicara (speaking) dalam TOEFL ITP. Selain itu, dalam publikasi TOEFL Research Insight Series, ETS menyebutkan bahwa dalam perancangan tes, pendekatan "evidence-centered design" digunakan agar setiap soal relevan dengan kemampuan bahasa yang ingin diukur serta adil bagi semua pengguna tes.

Namun, meskipun banyak kelebihan, keberlakuan dan efektivitas TOEFL juga kerap menjadi bahan perdebatan dan kajian ilmiah. Misalnya, dalam jurnal (Dewi et al., 2023) disebutkan bahwa menurut TOEFL iBT adalah "alat penilaian yang sangat baik" untuk mengevaluasi kemampuan bahasa Inggris dalam konteks akademik karena mencakup semua aspek (listening, reading, speaking, writing) secara terpadu. Sebaliknya, ada kritik terkait soal penilaian otomatis atau algoritmis dalam bagian menulis: penelitian terkini pada tugas *writing* TOEFL menunjukkan bahwa fitur-fitur linguistik seperti frekuensi *lexical bundle* dapat memperbaiki keselarasan antara skor mesin dan penilaian manusia dalam sistem penilaian otomatis esai TOEFL. Temuan ini memperlihatkan bahwa desain dan evaluasi TOEFL terus berkembang dan perlu disesuaikan agar tetap adil, valid, dan relevan bagi berbagai latar belakang pengguna.

2.3 Perancangan Aplikasi *Mobile*

Perancangan adalah suatu sekumpulan aktivitas yang menggambarkan secara rinci bagaimana sistem akan berjalan (Fauzi et al., 2022). Perancangan juga dapat diartikan sebagai proses menggambarkan rencana dan membuat sketsa atau menyusun berbagai elemen terpisah menjadi satu kesatuan yang berfungsi dengan baik. Sementara itu, aplikasi adalah kumpulan perintah yang dibuat untuk menjalankan tugas-tugas tertentu. Menurut (Putri Utami et al., n.d.) Aplikasi *mobile*, atau yang sering disebut *mobile apps*, merupakan jenis perangkat lunak yang dirancang untuk dijalankan pada perangkat bergerak seperti *smartphone* maupun tablet, dan beroperasi secara mandiri melalui sistem operasi yang mendukungnya. Pengguna dapat memperoleh aplikasi-aplikasi tersebut dengan mengunduhnya melalui platform distribusi resmi yang sesuai dengan sistem operasi perangkat, seperti Google Play Store untuk Android atau App Store untuk *Iphone Operating System* (IOS). Sedangkan aplikasi pembelajaran adalah media yang dapat digunakan perangkat *mobile* untuk menyampaikan isi materi yang melibatkan perangkat bergerak seperti ponsel berbasis android.

2.4 Android

Menurut (Ditha et al., 2024) Android merupakan sistem operasi berbasis Linux yang dirancang khusus untuk perangkat bergerak, seperti *smartphone* dan tablet, dengan mengintegrasikan sistem operasi, *middleware*, serta berbagai aplikasi pendukung. Sistem ini memberikan platform terbuka bagi para pengembang untuk merancang dan mengembangkan aplikasi secara bebas. Android sendiri menggunakan versi modifikasi dari kernel Linux serta komponen perangkat lunak sumber terbuka lainnya.

2.4.1 Sejarah Singkat Android

Android awalnya dikembangkan oleh Android Inc., sebuah perusahaan yang kemudian diakuisisi oleh Google pada tahun 2005. Pada tahun 2007, Google bersama dengan beberapa perusahaan teknologi terkemuka membentuk *Open Handset Alliance* (OHA), sebuah konsorsium yang bertujuan untuk mengembangkan standar terbuka bagi perangkat *mobile*. Anggota awal OHA

termasuk perusahaan-perusahaan besar seperti Texas Instruments, Intel, LG, Motorola, Samsung Electronics, Qualcomm, dan HTC. Pada tanggal 9 Desember 2008, proyek Android semakin berkembang dengan bergabungnya 14 anggota baru, termasuk Sony Ericsson, Toshiba, Vodafone, dan Garmin Ltd.. Dengan dukungan dari berbagai perusahaan teknologi, Android berkembang menjadi sistem operasi *mobile* yang dominan di dunia, dikenal karena sifatnya yang terbuka dan fleksibel bagi pengembang serta produsen perangkat (J. D. Ditha et al., 2024).

2.4.2 Kelebihan Android

Menurut (Navintino et al., 2025) Android mempunyai berbagai macam keunggulan yaitu :

a. Open Source

Android merupakan sistem operasi berbasis Linux yang berarti android menyediakan sumber daya yang dapat dikembangkan bagi siapa saja yang ingin mengembangkannya, hal dapat kita lihat dengan banyaknya versi android yang di modifikasi tersebar di internet.

b. Dukungan Google

Dikarenakan android merupakan salah satu aset yang di miliki oleh *Google*, hal ini membuat semua hal yang berhubungan dengan *Google* dengan otomatis tersinkronisasi *device* android contohnya seperti Youtube, Chrome, *Google Maps* dan lain-lain.

c. Modifikasi

Karakteristik *open-source* yang dimiliki oleh sistem operasi Android memberikan keleluasaan bagi para pengembang dalam melakukan kustomisasi dan pengembangan lanjutan. Hal ini memungkinkan Android untuk diadaptasi sesuai dengan kebutuhan spesifik.

d. User Friendly

Sistem operasi Android dikenal memiliki antarmuka yang intuitif dan *user-friendly*, sehingga memudahkan proses adaptasi bahkan bagi pengguna yang belum

memiliki pengalaman sebelumnya. Kemudahan navigasi serta desain yang responsif memungkinkan pengguna baru untuk memahami dan mengoperasikan perangkat berbasis Android dalam waktu yang relatif singkat.

2.5 Kotlin

Kotlin adalah bahasa pemrograman yang berjalan di atas *Java Virtual Machine* (JVM). Dan bersifat statis artinya bahasa pemrograman kotlin dapat menggunakan paradigma *Object Orirented Programming* (OOP) dan Fungsional. Kotlin sendiri merupakan bahasa pemrograman *open-source* yang dikembangkan oleh Jet Brains untuk berbagai macam platform akan tetapi bahasa pemrograman kotlin sendiri saat ini sangat populer digunakan untuk membangun aplikasi Android (Pratama et al., 2023).

Bahasa pemrograman kotlin merupakan bahasa pemrograman yang tergolong relatif mudah untuk dipelajari, hal ini karena sebagian besar kodenya mirip dengan java, bahkan lebih ringkas, bahasa pemrograman kotlin juga dapat dengan mudah diintegrasikan dengan IDE seperti NetBeans, IntelliJ, Visual Studio Code, dan juga Android Studio(Diantoni et al., 2024)

2.6 Jetpack

Jetpack adalah kumpulan *library*, *tools*, dan panduan pengembangan yang disediakan oleh *Google* untuk membantu pengembang membangun aplikasi Android berkualitas tinggi dengan lebih mudah dan efisien. Jetpack dirancang untuk mengurangi kode *boilerplate* yang berlebihan, menyederhanakan pekerjaan kompleks, dan memungkinkan pengembang untuk lebih fokus pada logika bisnis aplikasi (Diantoni et al., 2024)

Jetpack terdiri dari berbagai komponen modular yang terbagi ke dalam empat kategori utama, yaitu *Architecture*, UI, *Behavior*, dan *Foundation*. Setiap kategori memiliki peran yang berbeda dalam membantu pengembangan aplikasi. Misalnya, komponen *Architecture* seperti *ViewModel*, *LiveData*, dan *Room* membantu pengembang dalam mengelola data dan siklus hidup aplikasi secara efisien. Sementara itu, komponen UI seperti *Navigation Component* dan *Data Binding* mempermudah pembuatan antarmuka pengguna yang responsif dan

dinamis. Dengan struktur yang modular, Jetpack memungkinkan pengembang menggunakan hanya komponen yang dibutuhkan tanpa harus mengimpor keseluruhan pustaka, sehingga proyek menjadi lebih ringan dan terstruktur.

Selain itu, Jetpack terus dikembangkan secara berkelanjutan oleh Google untuk mendukung praktik *modern Android development (*MAD*)*. Pendekatan ini mencakup penggunaan bahasa *Kotlin*, arsitektur berbasis komponen, dan integrasi dengan alat pengujian serta sistem build yang lebih efisien. Jetpack juga memastikan kompatibilitas yang tinggi antar versi Android, sehingga aplikasi tetap berjalan dengan stabil di berbagai perangkat dan versi sistem operasi. Menurut (Alpian & Muharom, n.d.) penggunaan Jetpack tidak hanya meningkatkan produktivitas pengembang, tetapi juga mempercepat proses pengembangan aplikasi dengan hasil yang lebih konsisten, aman, dan mudah dipelihara dalam jangka panjang.

2.7 n8n

n8n merupakan sebuah platform *workflow automation* yang memungkinkan pengembang menghubungkan berbagai aplikasi, layanan, dan API tanpa harus menulis kode secara penuh, meskipun tetap mendukung penggunaan skrip kustom. Platform ini bekerja dengan konsep "node" sebagai unit tindakan (*action*) atau pemicu (*trigger*) yang dihubungkan dalam satu alur kerja (*workflow*). Menurut (Aditia Ramadhani et al., 2025)pendekatan berbasis visual ini membuat n8n mudah digunakan baik oleh pengguna teknis maupun non-teknis, sekaligus mendukung integrasi dengan lebih dari 300 layanan populer seperti *Google Sheets*, *Slack*, dan *GitHub*.

Keunggulan utama n8n terletak pada fleksibilitasnya yang dapat dijalankan secara *self-hosted* di server pribadi maupun melalui layanan cloud, serta menggunakan lisensi *fair-code*. Artinya, pengguna memiliki kebebasan untuk mengakses dan memodifikasi sebagian besar kode sumbernya. Menurut *DataScientest* (2023), kemampuan ini memberikan kendali penuh terhadap keamanan data dan infrastruktur, yang sangat penting dalam pengembangan sistem yang membutuhkan privasi tinggi. Selain itu, n8n juga mendukung logika alur yang kompleks seperti kondisi (*if/else*), *loop*, serta manipulasi data antar-node,

menjadikannya alat otomasi yang tidak hanya sederhana tetapi juga sangat kuat dan dapat diadaptasi untuk berbagai kebutuhan integrasi (Riza et al., n.d.).

Dalam konteks pengembangan aplikasi *mobile*, n8n berperan penting sebagai penghubung antara backend dan layanan eksternal tanpa perlu membangun API dari nol. Misalnya, pengembang dapat menggunakan n8n untuk mengatur otomatisasi notifikasi push, sinkronisasi data pengguna antar server, atau integrasi dengan layanan pihak ketiga seperti Firebase dan Supabase. Menurut *ServerSpace* (2024), penggunaan n8n dalam proyek mobile memungkinkan pengembang mempercepat proses pengujian serta meminimalkan kesalahan manual pada proses *deployment* dan *data flow*. Dengan demikian, n8n tidak hanya mendukung efisiensi pengembangan aplikasi mobile, tetapi juga meningkatkan skalabilitas dan kestabilan sistem secara keseluruhan.

2.7.1 Kelebihan n8n

Menurut (Aditia Ramadhani et al., 2025) n8n mempunyai berbagai macam keunggulan yaitu :

a. Bersifat Open-Source dan Self-Hosted

n8n dapat dijalankan secara mandiri (*self-hosted*) di server pengguna sehingga memberikan kendali penuh terhadap data dan infrastruktur. Keunggulan ini menjadikan pengguna tidak bergantung pada penyedia layanan pihak ketiga dan dapat menyesuaikan konfigurasi sistem sesuai kebutuhan organisasi.

b. Antarmuka Visual Berbasis Node

n8n menawarkan tampilan editor visual yang memungkinkan pengguna membangun alur kerja otomatis dengan sedikit atau tanpa penulisan kode. Fitur ini mempermudah pengguna non-teknis untuk membuat otomasi sederhana, sekaligus tetap mendukung pengembang dalam menambahkan logika kustom melalui skrip.

c. Fleksibilitas Integrasi Aplikasi dan API

Platform ini mendukung integrasi dengan ratusan layanan, aplikasi, serta API eksternal, memungkinkan proses otomasi lintas sistem yang efisien.

Fleksibilitas ini membuat n8n dapat digunakan untuk berbagai kebutuhan, mulai dari manajemen data, notifikasi otomatis, hingga integrasi antar-platform.

d. Efisiensi

Dengan mengotomasi proses rutin, n8n membantu menghemat waktu kerja, mengurangi kesalahan manusia, serta meningkatkan konsistensi hasil. Hal ini mendukung terciptanya efisiensi operasional dalam lingkungan kerja digital.

2.7.2 Node node n8n untuk integrasi AI

a. Webhook

Node ini berfungsi sebagai pemicu (*trigger node*) yang mengawali seluruh alur kerja n8n. Ketika pengguna mengakses fitur *Latihan Pintar* di aplikasi, sistem mengirimkan permintaan (*request*) ke server n8n yang berisi parameter seperti jumlah soal, tingkat kesulitan, dan topik tertentu. Node ini memastikan bahwa setiap permintaan pengguna dapat ditangkap secara otomatis untuk diproses lebih lanjut tanpa intervensi manual.

b. Read data from spreadsheeet

Node ini berfungsi sebagai pemicu (*trigger node*) yang mengawali seluruh alur kerja n8n. Ketika pengguna mengakses fitur *Latihan Pintar* di aplikasi, sistem mengirimkan permintaan (*request*) ke server n8n yang berisi parameter seperti jumlah soal, tingkat kesulitan, dan topik tertentu. Node ini memastikan bahwa setiap permintaan pengguna dapat ditangkap secara otomatis untuk diproses lebih lanjut tanpa intervensi manual.

c. Generate Question with AI

Node ini merupakan inti dari alur kerja karena bertugas menghubungkan n8n dengan layanan AI. Berdasarkan parameter dan data pengguna yang telah diperoleh, node ini menginstruksikan model AI untuk menghasilkan kumpulan soal baru secara otomatis. Proses ini memungkinkan sistem menciptakan soal yang bervariasi, adaptif, dan sesuai dengan tingkat kemampuan masing-masing pengguna.

d. Information Extractor

Node ini digunakan untuk mengekstrak dan menyeleksi hasil keluaran dari model AI. Data mentah yang dihasilkan AI akan diperiksa kembali agar memenuhi struktur yang benar, seperti kelengkapan pertanyaan, opsi jawaban, kunci jawaban, serta tingkat kesulitan. Node ini juga memastikan bahwa setiap butir soal tidak duplikat dan sesuai dengan format standar TOEFL.

e. Merge Data

Node ini berfungsi untuk menggabungkan data yang telah dihasilkan oleh AI dengan data pengguna yang diambil dari basis data. Proses penggabungan ini menghasilkan satu paket latihan lengkap yang memuat soal, tingkat kesulitan, topik, serta waktu pengerjaan. Dengan adanya tahap ini, sistem dapat menyajikan latihan yang personal dan sesuai dengan kemampuan individu pengguna.

f. Edits Field

Node ini bertugas melakukan penyempurnaan hasil akhir sebelum dikirim ke aplikasi. Beberapa proses yang dilakukan antara lain pengacakan urutan soal, pengaturan kembali opsi jawaban, serta penyesuaian format tampilan agar lebih menarik dan tidak monoton. Tahapan ini berperan penting dalam menjaga variasi dan keseimbangan tingkat kesulitan pada setiap sesi latihan.

g. Save Data to Memory

Node ini digunakan untuk menyimpan seluruh hasil latihan ke dalam basis data atau penyimpanan server. Data yang disimpan mencakup identitas pengguna, daftar soal, jawaban benar, serta metadata lainnya. Penyimpanan ini berguna untuk analisis performa pengguna dan pengembangan rekomendasi latihan di masa mendatang.

2.8 Android Studio

Android Studio merupakan sebuah *Integrated Development Environment* (IDE) yang dirancang dan dikembangkan oleh Google khusus untuk pengembangan aplikasi berbasis Android. IDE ini menggabungkan berbagai fitur dan komponen yang mendukung proses pengembangan, pengujian, serta penyebaran aplikasi

Android secara terpadu. Sejak dirilis secara resmi pada tahun 2013, Android Studio langsung menarik perhatian para pengembang karena menawarkan fitur yang lengkap dan antarmuka yang ramah pengguna, menjadikannya sebagai alat utama dalam pengembangan aplikasi Android di berbagai kalangan, mulai dari akademik hingga industri (Rahman et al., 2025)

Android Studio memiliki sejumlah fitur unggulan yang mendukung produktivitas pengembang. Fitur-fitur tersebut antara lain *Intelligent Code Editor* yang mampu memberikan saran otomatis dan deteksi kesalahan secara real-time, *Android Virtual Device (AVD) Manager* untuk menguji aplikasi di berbagai versi perangkat Android secara virtual, serta *Gradle Build System* yang memudahkan proses kompilasi dan pengelolaan dependensi proyek. Selain itu, integrasi dengan *SDK Tools* dan *Layout Editor* memungkinkan pengembang merancang antarmuka pengguna secara visual tanpa harus menulis kode secara manual, sehingga mempercepat proses pembuatan dan pengujian aplikasi (Anggraini & Maiyana, 2025)

Keunggulan utama Android Studio terletak pada kemampuannya dalam memberikan pengalaman pengembangan yang terpusat, efisien, dan selalu diperbarui. Dengan dukungan penuh dari Google, Android Studio secara rutin mendapatkan pembaruan agar tetap kompatibel dengan berbagai versi Android terbaru serta mendukung penggunaan bahasa pemrograman seperti Java, Kotlin, dan C++. Pembaruan berkelanjutan ini menjadikan Android Studio sebagai *standard development environment* bagi pengembang aplikasi Android modern, yang digunakan secara luas dalam penelitian, pendidikan, maupun industri teknologi informasi.

2.9 Metode Waterfall

Waterfall merupakan salah satu model pengembangan perangkat lunak yang paling klasik dan banyak digunakan dalam penelitian akademik. Model ini bersifat sekuensial, artinya setiap tahap harus diselesaikan terlebih dahulu sebelum melanjutkan ke tahap berikutnya. Tahapan tersebut meliputi analisis kebutuhan, perancangan sistem, implementasi kode, pengujian perangkat lunak, dan

pemeliharaan. Keunggulan dari metode Waterfall adalah dokumentasi yang jelas, alur kerja yang terstruktur, serta memudahkan peneliti menjelaskan proses penelitian secara sistematis (Mahardika et al., 2023). Selain itu, metode ini juga meminimalkan terjadinya kesalahan dalam pengembangan karena setiap tahap harus ditinjau dan disetujui terlebih dahulu sebelum berlanjut ke tahap berikutnya. Hal ini membuat Waterfall sangat cocok digunakan pada proyek dengan kebutuhan yang sudah pasti, stabil, dan tidak banyak mengalami perubahan selama proses pengembangan. Metode Waterfall mencakup beberapa tahapan, yaitu:

1. ***Requirements***

Pada tahap ini, seluruh kebutuhan sistem yang akan dikembangkan dianalisis dan didokumentasikan secara rinci. Pengembang melakukan pengumpulan data melalui studi literatur, wawancara, atau observasi untuk mengetahui apa saja yang dibutuhkan oleh pengguna. Hasil dari tahap ini adalah dokumen kebutuhan (requirement document) yang menjadi acuan pada tahap berikutnya.

2. ***System design***

Tahap desain bertujuan untuk mengubah kebutuhan yang telah diperoleh menjadi rancangan sistem yang lebih terstruktur. Rancangan ini mencakup desain arsitektur perangkat lunak, desain antarmuka pengguna, basis data, hingga alur logika program. Desain yang baik akan mempermudah proses implementasi, serta meminimalkan kesalahan dalam pengkodean.

3. ***Implementation***

Pada tahap ini, desain sistem yang telah disusun diwujudkan ke dalam bentuk kode program menggunakan bahasa pemrograman yang dipilih. Proses implementasi memerlukan ketelitian agar kode yang ditulis sesuai dengan spesifikasi yang telah ditentukan. Tahap ini menghasilkan perangkat lunak awal (prototype) yang siap untuk diuji.

4. *Testing*

Setelah tahap implementasi selesai, sistem yang telah dibangun diuji secara menyeluruh untuk memastikan semua fitur berfungsi dengan baik. Pengujian dilakukan mulai dari pengujian unit, pengujian integrasi antar modul, hingga pengujian sistem secara keseluruhan. Tujuan utama adalah untuk menemukan dan memperbaiki kesalahan atau bug agar sistem dapat berjalan stabil dan sesuai dengan kebutuhan.

5. *Maintenence*

Tahap terakhir adalah pemeliharaan, yaitu proses memperbaiki kesalahan yang mungkin masih ditemukan setelah perangkat lunak digunakan, serta melakukan pembaruan sesuai kebutuhan pengguna di masa mendatang. Pemeliharaan penting untuk memastikan perangkat lunak tetap relevan, stabil, dan dapat digunakan dalam jangka waktu panjang.

2.10 Penelitian terdahulu

Penelitian ini mengintegrasikan berbagai referensi terkait dengan metode dan objek penelitian untuk memberikan batasan yang jelas pada metode dan sistem yang akan dikembangkan lebih lanjut. Uraian penelitian sebelumnya yang menjadi dasar pengembangan penelitian ini dapat dilihat pada tabel 2.1.

Tabel 2. 1 Penelitian terdahulu

No	Peneliti	Judul	Hasil
1	Eva Sulistiana et al., (2024)	Implementasi aplikasi TOEFL Test Pro berbasis android bagi Mahasiswa	Hasilnya menunjukkan adanya peningkatan signifikan pada pemahaman grammar. Jumlah mahasiswa dengan nilai kategori "Good" meningkat dari 19,14% menjadi 42,55%, sementara yang berada di kategori "Poor" turun dari 51,06% menjadi 10,63%.

No	Peneliti	Judul	Hasil
2	Arsalna Furqan et al., (2023)	Perancangan aplikasi simulasi tes toefl berbasis android	Hasil pengujian menunjukkan bahwa aplikasi ini mendapatkan skor System Usability Scale (SUS) sebesar 80,2, yang termasuk kategori grade B, rating excellent, dan acceptable. Selain itu, validasi dari ahli media memperoleh skor 94%, sedangkan validasi dari ahli materi mencapai 97,7%, yang berarti aplikasi ini sangat layak digunakan.
3	Tengku Mhd. Zulfikar et al., (2024)	Rancang Bangun Aplikasi Quiz Simulasi TOEFL Memanfaatkan Algoritma Linear Congruential Generator (LCG) Berbasis Android	Hasil pengujian menunjukkan aplikasi berjalan baik di Android maupun laptop, mampu memperbarui data secara realtime, dan memiliki tampilan sederhana yang mudah dipahami.
4	Yuli Yana Astuti et al., (2024)	Rancang Bangun Aplikasi Simulasi Tes Toefl Berbasis Website (Studi Kasus: Program Studi Teknik Informatika Universitas Mataram)	aplikasi simulasi TOEFL berbasis website yang dikembangkan layak digunakan, memudahkan mahasiswa Universitas Mataram dalam mempersiapkan diri menghadapi tes TOEFL, serta dinilai memiliki kualitas baik oleh penggunanya

No	Peneliti	Judul	Hasil
5	Wandi Aprianto, (2025)	Rancang Bangun Aplikasi Tes Toefl Online Berbasis Web dengan Penilaian Otomatis pada Lembaga Bahasa Asc Indonesia	Hasil penelitian ini menunjukkan bahwa aplikasi tes TOEFL online berbasis web dengan sistem penilaian otomatis yang dibangun menggunakan Laravel, Vue.js, dan MySQL mampu memberikan pengalaman ujian yang lebih efisien, cepat, dan akurat. Sistem ini dapat menghitung skor secara instan sesuai standar TOEFL, menampilkan laporan hasil secara detail, serta memudahkan peserta dan admin dalam mengelola ujian.

2.11 Perbandingan Aplikasi

Integrasi *Artificial Intelligence* (AI) dalam *Mobile Assisted Language Learning* (MALL) kini telah mengubah standar pembelajaran menjadi lebih adaptif dan personal. Guna menemukan celah pengembangan (*research gap*) yang tepat di antara banyaknya aplikasi yang beredar, penelitian ini melakukan studi komparasi terhadap lima *platform* populer, yaitu Duolingo, Elsa Speak, TestGlider, TOEFL Go!, dan Talkpal AI. Analisis difokuskan pada metode AI yang diterapkan serta evaluasi kelebihan dan kekurangannya untuk memperjelas posisi penelitian. Rangkuman perbandingan tersebut disajikan dan dapat dilihat pada Tabel 2.2 berikut.

Tabel 2. 2 Perbandingan Aplikasi

No	Nama Aplikasi	Penonjolan Fitur AI (metode)	Kelebihan	Kekurangan
1	Duolingo	*Birdbrain AI* (Adaptive Learning) untuk personalisasi tingkat kesulitan soal.	• Tampilan gamifikasi yang menarik minat belajar. • Melatih 4 aspek bahasa dasar secara gratis.	• Materi kurang spesifik untuk format akademik/TOEFL. • Pola latihan cenderung repetitif.
2	Elsa Speak	*Speech Recognition* mendalam untuk analisis fonem dan intonasi.	• Melatih 4 aspek bahasa dasar secara gratis. • Efektif untuk melatih aksen *native*.	• Hanya fokus pada *Speaking/Listening*. • Tidak menyediakan latihan *Reading* atau struktur kalimat kompleks.
3	TestGlider	*AI Grader* untuk penilaian otomatis esai dan *speaking*.	• Mampu memberikan prediksi skor TOEFL instan. • Simulasi sangat mirip dengan tes asli.	• Biaya Berlangganan relatif mahal • Penilaian esai terkadang kaku pada *template*.
4	TOEFL Go	*SpeechRater* dan *e-rater* (mesin penilai resmi ETS).	• Akurasi penilaian menggunakan standar resmi • Menggunakan soal – soal autentik terdahulu	• Antar muka (UI) kurang responsif • Jumlah soal latihan gratis sangat terbatas

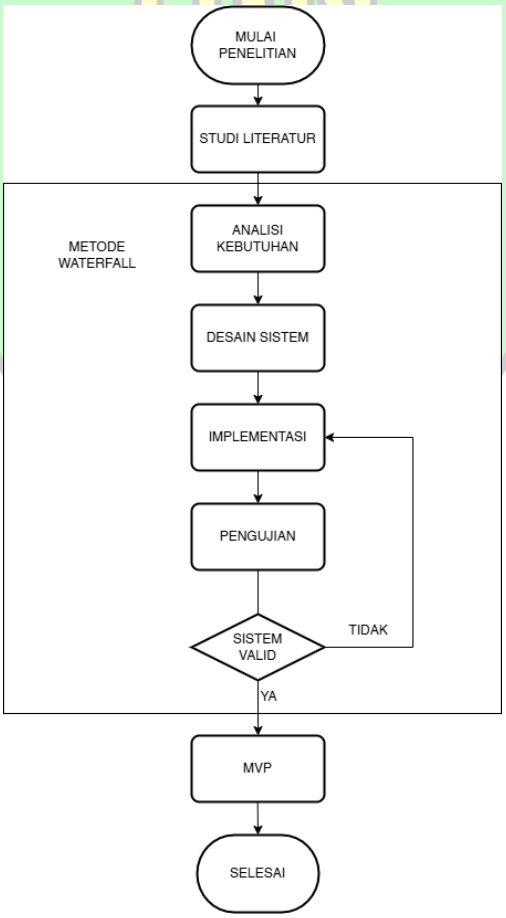
No	Nama Aplikasi	Penonjolan Fitur AI (metode)	Kelebihan	Kekurangan
5	TalkPad AI	Menghadirkan tutor AI yang bisa diajak berdebat, diskusi topik akademik, atau simulasi wawancara secara natural tanpa skrip kaku.	• Interaksi sangat natural dan fleksibel • Melatih spontanitas dalam berbicara	• Risiko halusinasi (fakta tidak akurat) • Kurang terstruktur dalam menjawab soal ujian

Berdasarkan Tabel 2.2 di atas, dapat dilihat bahwa meskipun aplikasi-aplikasi tersebut telah menerapkan teknologi AI yang canggih, sebagian besar masih berfokus pada kemampuan bahasa umum atau memiliki keterbatasan dalam aksesibilitas (berbayar mahal). Belum banyak ditemukan aplikasi yang secara spesifik menggabungkan simulasi TOEFL yang terjangkau dengan kemampuan analisis pembahasan soal berbasis *Large Language Model* (Gemini AI). Oleh karena itu, penelitian ini bertujuan untuk mengisi celah tersebut dengan mengembangkan aplikasi yang tidak hanya menyajikan skor akhir, namun juga terjangkau untuk mahasiswa karena bisa diakses dengan gratis dan bisa diakses ketika saat tidak ada jaringan internet atau dalam keaadan offline.

BAB III
METODE PENELITIAN

3.1 Tahapan Penelitian

Penelitian ini dilaksanakan melalui beberapa tahapan utama yang bertujuan untuk menghasilkan sistem pengelolaan survei dan laporan yang efektif serta sesuai dengan kebutuhan pengguna. Tahapan tersebut mencakup identifikasi masalah untuk merumuskan fokus penelitian, pengumpulan data sebagai landasan perancangan sistem, perancangan sistem dengan menggunakan metode Waterfall, implementasi aplikasi dengan bahasa pemrograman Kotlin, evaluasi sistem melalui pengujian serta uji coba pengguna, dan penyusunan laporan penelitian sebagai bentuk dokumentasi keseluruhan proses. Setiap tahapan akan dijelaskan secara lebih rinci pada subbab berikutnya. Tahapan penelitian ini dapat dilihat pada Gambar 3.1.



Gambar 3. 1 Diagram Alur Penelitian

3.2 Studi Literatur

Pada tahap ini, penulis mengumpulkan berbagai data atau informasi yang terkait dengan pelatihan TOEFL dan perancangan aplikasi yang relevan dengan penelitian ini. Sumber data mencakup jurnal, buku, penelitian terdahulu, artikel, dan literatur lainnya.

Penulis juga melakukan eksplorasi terhadap aplikasi-aplikasi latihan TOEFFL yang telah dikembangkan dalam penelitian sebelumnya dan aplikasi serupa yang tersedia di toko aplikasi saat penelitian ini dilakukan. Penulis meneliti apakah aplikasi yang dikembangkan telah sesuai dengan kaidah penyusunan materi pembelajaran TOEFL, serta menelaah keberadaan fitur-fitur pendukung seperti penerepan AI untuk bank soal, latihan *listening*, *reading*, dan *structure*, kuis interaktif, serta modul latihan speaking dan writing dan juga mode offline nya.

Melalui studi literatur ini, penulis memperoleh pemahaman yang lebih mendalam mengenai kebutuhan pengguna, kekuatan dan kelemahan dari aplikasi pelatihan TOEFL yang telah ada, serta peluang inovasi yang dapat diimplementasikan dalam pengembangan aplikasi latihan TOEFL ini.

Analisis terhadap kekurangan fungsional dan kurangnya integrasi fitur pada aplikasi yang telah tersedia menjadi dasar pertimbangan dalam merancang solusi yang lebih komprehensif dan responsif terhadap kebutuhan mahasiswa untuk latihan TOEFL secara mandiri. Dengan demikian, studi literatur tidak hanya berfungsi sebagai landasan teoretis, tetapi juga sebagai pijakan strategis dalam menentukan arah pengembangan aplikasi yang tepat guna.

3.3 Metode Pengembangan Sistem

Metode pengembangan aplikasi yang digunakan dalam penelitian ini adalah *Waterfall*, yang bersifat sekuensial atau berurutan. Setiap tahapan harus diselesaikan terlebih dahulu sebelum melanjutkan ke tahap berikutnya, sehingga proses pengembangan dapat terdokumentasi secara sistematis. Tahapan utama dalam metode *Waterfall* mencakup *requirements analysis, system design, implementation, testing, deployment,* dan *maintenance*. Model ini dipilih karena sesuai dengan kebutuhan penelitian, di mana pengembangan aplikasi latihan TOEFL berbasis mobile menggunakan Kotlin memerlukan alur yang terstruktur serta dokumentasi yang jelas dalam setiap tahap pengembangannya.

4.3. Analisis Kebutuhan

Penulis pertama mengidentifikasi kebutuhan atau persyaratan yang diperlukan dalam pengembangan aplikasi sesuai dengan langkah pengembangan dalam metode *Waterfall* termasuk kebutuhan fungsional dan non-fungsional.

1. Kebutuhan Fungsional

Kebutuhan fungsional menggambarkan fungsi dan fitur yang akan disediakan oleh aplikasi. Untuk aplikasi Pelatihan TOEFL berbasis *mobile* ini, sebagian besar kebutuhan fungsional diadopsi dari fitur aplikasi pelatihan TOEFL yang sudah ada di toko aplikasi, serta dari hasil penelitian sebelumnya. Namun, dalam penelitian ini dilakukan beberapa peningkatan, seperti peningkatan antarmuka pengguna dan penambahan fitur baru, detail kebutuhan fungsional dapat dilihat pada tabel 3.1.

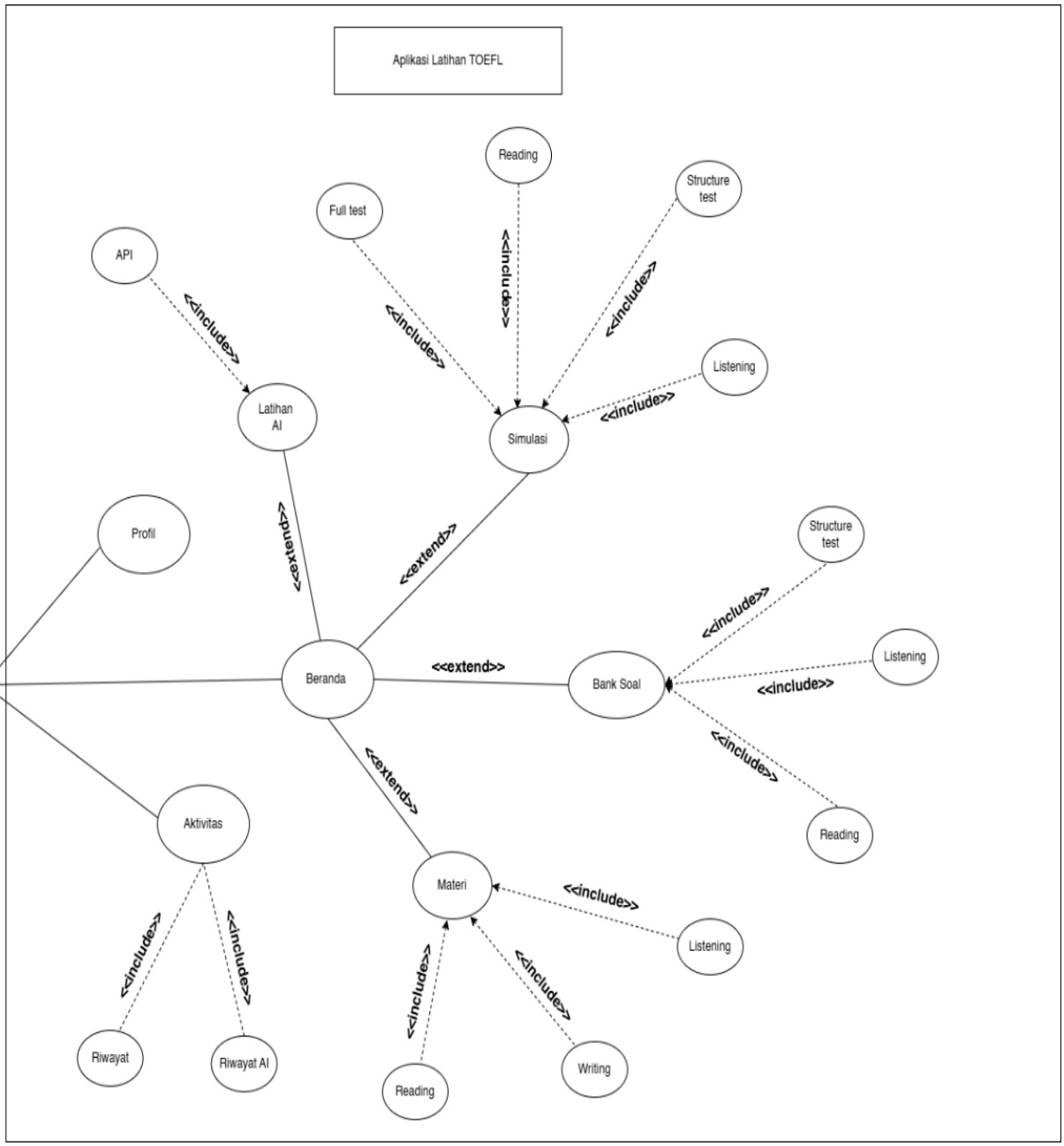
Tabel 3. 1 Fitur Aplikasi

No	Fitur	Deskripsi
1	*Main*	Fitur ini berfungsi sebagai tampilan awal yang menyediakan *shortcut* langsung menuju fitur utama lainnya pada aplikasi, seperti *simulasi*, Bank Soal, Materi, dan Latihan Pintar yang terintegrasi dengan AI
2	*Materi*	Fitur ini berisi materi materi pembelakajaran untuk latihan toefl beserta rumus nya dan juga contoh penggunaan kalimat yang baik dan benar.
3	*Simulasi*	Fitur ini berfungsi untuk memberikan pengalaman ujian yang menyerupai tes TOEFL sebenarnya. Pengguna dapat mengerjakan soal Listening, Structure, dan Reading dan soal campuran dalam durasi waktu tertentu
4	*Bank Soal*	Fitur berikut berisi kumpulan pertanyaan dari berbagai kategori seperti Listening, Structure, dan Reading. Pengguna dapat memilih dan mengerjakan soal untuk melatih kemampuan secara bertahap.

No	Fitur	Deskripsi
5	*Latihan AI*	Fitur latihan berbasis AI ini berfungsi untuk memberikan latihan yang disesuaikan secara otomatis berdasarkan kemampuan pengguna.
6	Aktivitas	Fitur ini akan memperlihatkan riwayat latihaan Anda, dan juga Anda bisa mereview latihan yang sudah Anda kerjakan. Dan juga ada fitur riwayat AI yaitu riwayat latihan yang diberikan oleh AI
7	*Profil*	Fitur ini berfungsi sebagai pusat identitas pengguna yang menampilkan informasi pribadi seperti nama, foto, alamat email, serta status akun. Melalui fitur ini, pengguna dapat melihat dan mengubah data diri, mengatur preferensi aplikasi, serta mengakses riwayat aktivitas atau progres penggunaan.

2. Use Case Diagram

Berdasarkan kebutuhan fungsional yang telah dijelaskan sebelumnya, hubungan antara pengguna dan sistem dapat digambarkan dalam *Use Case* Diagram berikut. Diagram ini menggambarkan interaksi antara aktor (pengguna) dengan fungsi utama yang disediakan oleh sistem.



Gambar 3. 2 *Use Case Diagram*

3. Kebutuhan Non-Fungsional

Kebutuhan non-fungsional menjelaskan batasan terkait fungsi atau fitur yang disediakan oleh aplikasi. Batasan-batasan ini mencakup perangkat lunak dan perangkat keras yang digunakan dalam pengembangan aplikasi. Dalam penelitian ini, perangkat lunak yang digunakan untuk mengembangkan aplikasi latihan TOEFL ini dapat dilihat pada Tabel 3.2.

Tabel 3. 2 Perangkat Lunak

No	Nama	Versi
1	Android Studio	2024.1.1
2	Kotlin	2.0.21
3	Postman	11.17.0
5	Github	-
6	Figma	124.4.7
7	n8n	

Perangkat keras yang digunakan dalam pengembangan aplikasi latihan TOEFL pada penelitian ini dapat dilihat pada Tabel 3.3.

Tabel 3. 3 Perangkat Keras

No	Nama	Versi
1	Laptop	Asus TUF F15
2	*Operation System*	Windows 11
3	CPU	Intel gen 11 core i5-11400H
5	GPU	RTX 3050
6	RAM	24 GB

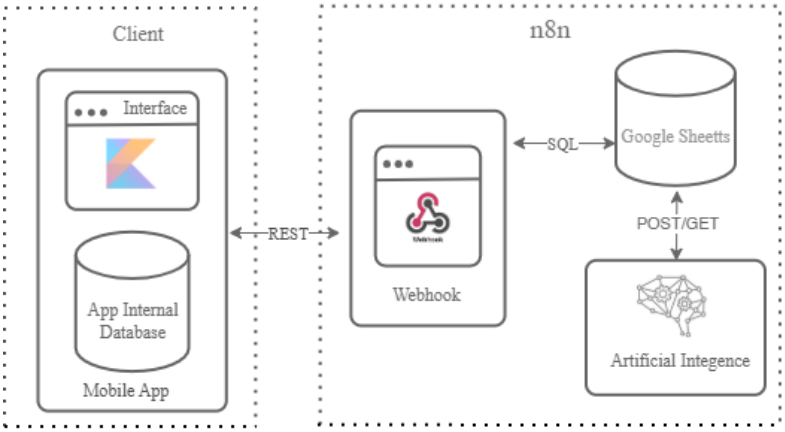
4.4. Desain Sistem

Pada tahap ini, penulis merancang desain antarmuka pengguna aplikasi serta memodelkan sistem yang akan diimplementasikan selama proses iterasi. Desain antarmuka dan pemodelan sistem dibuat dengan tujuan untuk memenuhi analis kebutuhan tanpa mencoba memprediksi kebutuhan masa depan. Pendekatan ini memastikan fokus pengembangan tetap pada kebutuhan yang telah ditentukan, sesuai dengan prinsip iteratif yang digunakan.

1. Arsitektur Aplikasi

Aplikasi yang dirancang pada penelitian ini menerapkan gaya arsitektur *Representational State Transfer* (REST). Salah satu prinsip utama REST adalah penerapan pola desain *client-server*, yang berarti tugas pada *client* dan *server* dipisahkan. Pada sisi aplikasi, Latihan **TOEFL** menggunakan Kotlin sebagai bahasa pemrograman untuk menangani antarmuka pengguna pada sisi *client*. Data diproses melalui API REST yang berperan sebagai penghubung antara aplikasi *mobile* dan *server*. Pada sisi *server*, API berinteraksi dengan *database* melalui perintah SQL untuk penyimpanan dan pengolahan data. Selain itu, terdapat integrasi antara *server* dengan model *Machine Learning* yang berinteraksi dengan *database* untuk melakukan operasi POST/GET sesuai kebutuhan.

Aplikasi yang dirancang pada penelitian ini mengintegrasikan n8n sebagai mesin orkestrasi alur kerja (workflow automation) yang melengkapi arsitektur client–server berbasis REST. Sejalan dengan prinsip pemisahan tanggung jawab, antarmuka pengguna tetap ditangani di sisi client menggunakan Kotlin, sedangkan n8n berperan di sisi server sebagai middleware yang menerima permintaan melalui Webhook/HTTP Trigger, memproses logika bisnis terotomasi, lalu berkomunikasi dengan layanan lain menggunakan konektor atau permintaan REST standar. Data dari aplikasi mobile dikirim dalam format JSON ke endpoint n8n; alur kerja n8n kemudian memanggil API internal/eksternal, melakukan transformasi data (misalnya normalisasi atau validasi), dan meneruskan hasilnya ke layanan penyimpanan maupun komponen lain yang membutuhkan.



Gambar 3. 3 Arsitektur Aplikasi

2. *Mid-fidelity Wireframe*

Mid-fidelity wireframe adalah sketsa atau kerangka gambar yang menampilkan struktur desain aplikasi secara lebih rinci, termasuk letak dan interaksi antar fitur utama. Pada *wireframe* jenis ini, elemen pada aplikasi sudah lebih jelas dibandingkan *low-fidelity wireframe*, tetapi belum melibatkan elemen visual penuh seperti warna dan gambar yang kompleks dan final.

a. *Main Screen*

Main Screen adalah halaman utama yang ditampilkan ketika pengguna pertama kali membuka aplikasi dan akan berfungsi sebagai pusat navigasi utama aplikasi. Pengguna dapat memilih fitur utama yang ingin mereka akses menggunakan *bottom navigation* dan *shortcut* yang terdapat pada halaman ini.

Gambar 3. 4 *Main Screen – Wireframe*

b. Simulasi *Screen*

Halaman ini digunakan sebagai tempat bagi pengguna untuk mempersiapkan diri menghadapi ujian TOEFL melalui berbagai mode simulasi, seperti Listening, Structure, Reading, maupun Full Test. Yang terintegras langsung dengan berpa lama waktu simulasi nya.

Gambar 3. 5 Dialog Status *Screen* – *Wireframe*

c. *Reading* Simulasi *Screen*

Halaman ini digunakan oleh pengguna untuk mengerjakan bagian Reading Test dalam simulasi TOEFL. Pada halaman ini ditampilkan teks bacaan (passage) beserta pertanyaan dan pilihan jawaban yang harus dipilih oleh pengguna. Terdapat juga pengatur waktu untuk memantau durasi pengerjaan serta tombol navigasi untuk berpindah ke soal sebelumnya..

Gambar 3. 6 *Reading* Simulasi *Screen – Wireframe*

d. *Listening* simulasi *Screen*

Halaman ini digunakan oleh pengguna untuk mengerjakan bagian Listening Test dalam simulasi TOEFL. Pada halaman ini tersedia pemutar audio yang menampilkan percakapan atau monolog berbahasa Inggris, disertai pertanyaan dan pilihan jawaban yang harus dipilih berdasarkan isi rekaman. Tersedia juga indikator waktu pengerjaan serta tombol navigasi untuk berpindah antar soal.

Gambar 3. 7 *Listening* simulasi *Screen– Wireframe*

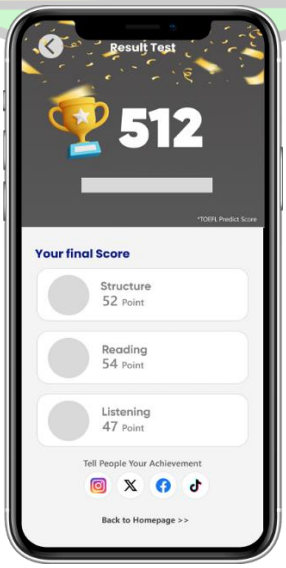
e. Struktur *Simulasi Screen*

Halaman ini digunakan oleh pengguna untuk mengerjakan bagian Structure Test dalam simulasi TOEFL. Pada halaman ini disajikan soal yang berfokus pada tata bahasa (grammar) dan struktur kalimat dalam bahasa Inggris. Pengguna diminta memilih jawaban yang paling tepat untuk melengkapi atau memperbaiki kalimat yang disediakan. Tersedia pula timer untuk memantau waktu serta tombol navigasi untuk berpindah antar soal.

Gambar 3. 8 *Struktur simulasi Screen – Wireframe*

f. Hasil Siulasi *Screen*

Halaman ini digunakan untuk menampilkan hasil akhir dari simulasi TOEFL yang telah dikerjakan oleh pengguna. Pada halaman ini ditampilkan skor prediksi TOEFL secara keseluruhan beserta rincian nilai dari setiap bagian tes, yaitu Structure, Reading, dan Listening. Selain itu, pengguna juga dapat membagikan pencapaiannya ke media sosial serta kembali ke halaman utama aplikasi.



Gambar 3. 9 Hasil Simulasi *Screen – Wireframe*

g. Bank Soal *Screen*

Halaman ini digunakan oleh pengguna untuk mengakses kumpulan soal latihan yang tersimpan dalam Bank Soal. Pada halaman ini, pengguna dapat memilih jenis tes seperti Structure serta menentukan bab atau chapter yang ingin dikerjakan. Fitur soal acak juga tersedia untuk memberikan variasi dalam latihan, sehingga pengguna dapat menguji pemahaman mereka secara lebih menyeluruh.

Gambar 3. 10 Bank Soal *Screen – Wireframe*

h. Daftar lengkap bank soal *Screen*

Halaman ini digunakan untuk menampilkan daftar lengkap kumpulan soal latihan yang tersedia dalam Bank Soal. Pengguna dapat mencari soal berdasarkan kata kunci melalui kolom pencarian dan memilih soal yang ingin dikerjakan dengan menekan tombol “Lihat”. Tampilan ini memudahkan pengguna dalam menemukan dan mengakses materi latihan secara cepat.

Gambar 3. 11 Daftar lengkap bank soal *Screen – Wireframe*

i. Bank soal *Reading Screen*

Halaman ini digunakan oleh pengguna untuk mengerjakan bagian Reading Test sekaligus mempelajari pembahasannya. Selain menampilkan teks bacaan, pertanyaan, dan pilihan jawaban, halaman ini juga menyediakan fitur "Pembahasan" yang berfungsi untuk menampilkan penjelasan atas jawaban yang benar. Fitur ini membantu pengguna memahami kesalahan dan meningkatkan kemampuan membaca serta menganalisis teks bahasa Inggris secara efektif.

Gambar 3. 12 *Bank* Soal *reading Screen – Wireframe*

j. *Bank* Soal *Listening Screen*

Halaman ini digunakan oleh pengguna untuk mengerjakan bagian Listening Test sekaligus memahami pembahasannya. Pada halaman ini terdapat pemutar audio yang berisi percakapan atau monolog dalam bahasa Inggris, disertai pertanyaan dan pilihan jawaban. Fitur "Pembahasan" di bagian bawah membantu pengguna mempelajari alasan di balik jawaban yang benar, sehingga meningkatkan pemahaman terhadap konteks percakapan dan kemampuan mendengarkan secara efektif.

Gambar 3. 13 *Bank* Soal *Listening Screen – Wireframe*

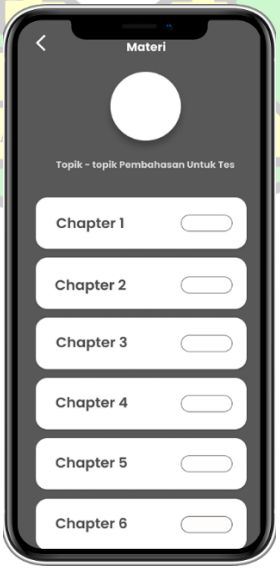
k. *Bank Soal Structure Screen*

Halaman ini digunakan untuk mengerjakan bagian Structure Test yang berfokus pada tata bahasa. Pengguna memilih jawaban yang tepat untuk melengkapi kalimat dan dapat melihat pembahasan untuk memahami jawabannya. Fitur ini membantu meningkatkan kemampuan grammar secara efektif.

Gambar 3. 14 *Bank soal structure – Wireframe*

l. Materi *Screen*

Halaman ini digunakan untuk menampilkan daftar materi pembelajaran TOEFL yang terbagi ke dalam beberapa chapter. Pengguna dapat memilih setiap chapter untuk mempelajari topik tertentu yang berkaitan dengan bagian tes. Fitur ini membantu pengguna memahami konsep dasar sebelum melakukan latihan soal atau simulasi.



Gambar 3. 15 Materi *Screen – Wireframe*

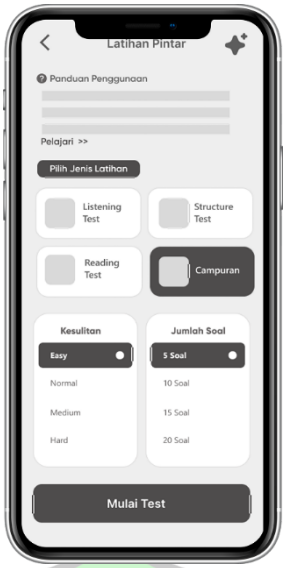
m. Materi *Screen*

Halaman ini menampilkan daftar materi aksara yang tersedia dalam aplikasi seperti cara baca dan aturan penulisan aksara sesuai dengan jenis aksara yang dipilih pada Belajar *Screen*.

Gambar 3. 16 Materi *Screen – Wireframe*

n. Latihan Pintar *Screen*

Halaman ini digunakan untuk mengakses fitur Latihan Pintar yang memungkinkan pengguna berlatih soal TOEFL secara adaptif. Pengguna dapat memilih jenis latihan seperti Listening, Structure, Reading, atau Campuran, serta menentukan tingkat kesulitan dan jumlah soal. Fitur ini membantu pengguna berlatih sesuai kemampuan dan fokus pada peningkatan hasil belajar secara bertahap.



Gambar 3. 17 latihan pintar *screen - wireframe*

o. Aktivitas Screen

Halaman ini digunakan untuk menampilkan riwayat hasil simulasi TOEFL yang telah dikerjakan oleh pengguna. Melalui halaman ini, pengguna dapat melihat kembali skor dan detail ujian sebelumnya. Fitur ini membantu pengguna memantau perkembangan kemampuan mereka dari waktu ke waktu dan mengevaluasi hasil belajar secara mandiri.

Gambar 3. 18 Aktivitas *Screen - Wireframe*

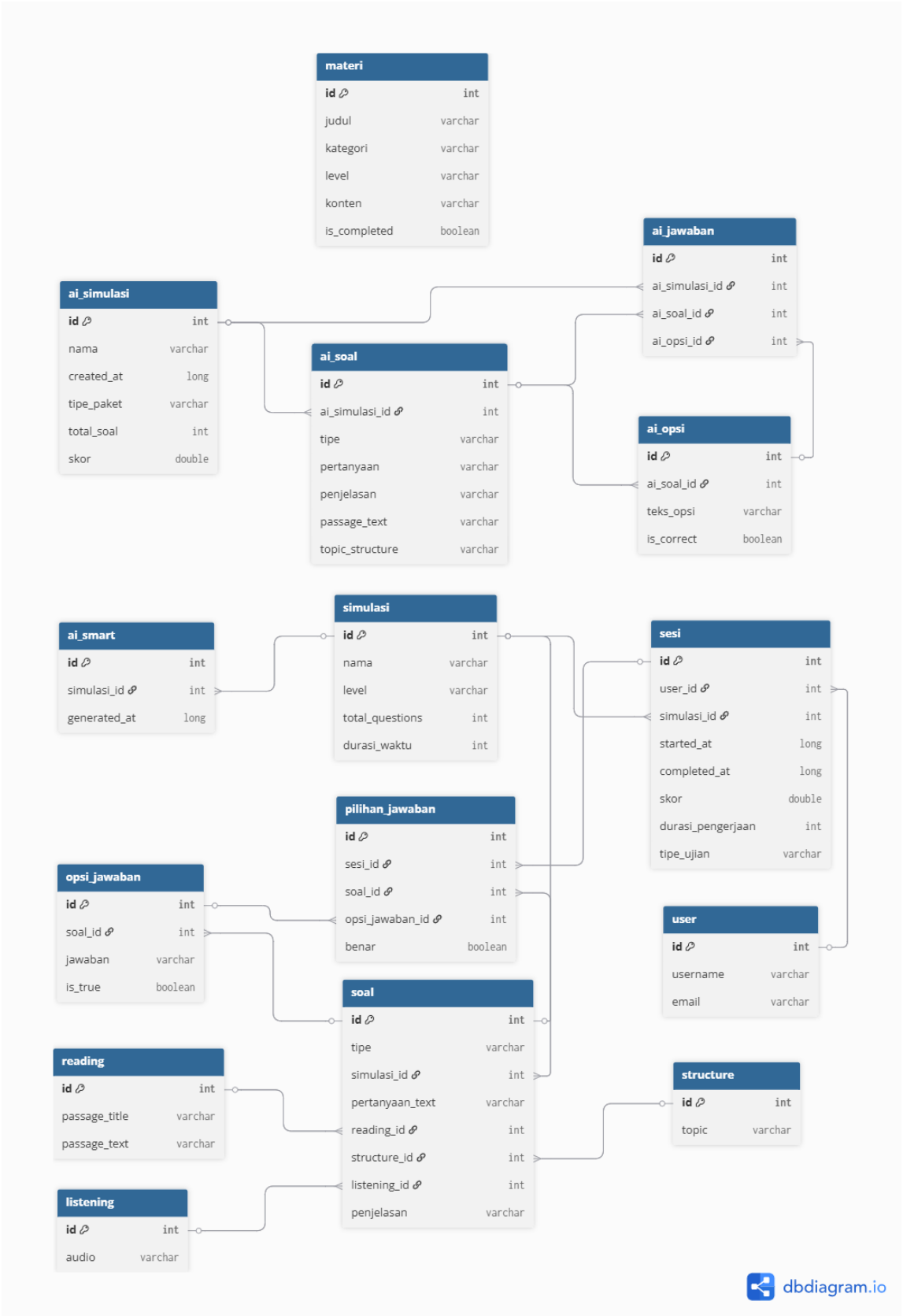
p. *Profile* Screen

Halaman ini digunakan untuk menampilkan dan mengelola informasi profil pengguna. Pada halaman ini, pengguna dapat melihat data pribadi seperti nama, email, dan nomor telepon, serta melakukan pembaruan melalui fitur “Edit Profil”. Selain itu, tersedia tombol “Logout” untuk keluar dari akun dengan aman.

Gambar 3. 19 *Profile Screen – Wireframe*

3. *Database Schema*

Database schema adalah rancangan atau cetak biru yang memudahkan pengembang dalam memvisualisasikan struktur data pada database relasional. Ini meliputi elemen-elemen utama seperti nama tabel, kolom atau field, tipe data untuk setiap kolom, serta hubungan antara tabel. Pada Gambar 3.20 dapat dilihat database schema untuk aplikasi Latihan TOEFL.



Gambar 3. 20 *Database Schema*

Tabel 3. 4 Tabel Ai_Simulasi

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**
judul	Text	Judul materi belajar
kategori	Text	Kategori materi
level	Text	Level kesulitan
konten	Text	Isi materi
is_completed	Boolean	Status penyelesaian

Tabel 3. 5 Tabel Ai_Soal

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**, Auto Increment
ai_simulasi_id	Integer	**FOREIGN KEY** (Ref: ai_simulasi.id)
tipe	Text	Jenis soal (reading, structure)
pertanyaan	Text	Teks pertanyaan
penjelasan	Text	Penjelasan jawaban
passage_text	Text	*Nullable* (Teks bacaan langsung di sini)
topic_structure	Text	*Nullable* (Topik grammar)

Tabel 3. 6 Tabel Ai_opsi

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**, Auto Increment
ai_soal_id	Integer	**FOREIGN KEY** (Ref: ai_soal.id)
teks_opsi	Text	Pilihan jawaban (A, B, C, D)
is_correct	Boolean	Menandakan jawaban benar

Tabel 3. 7 Tabel Ai_Jawaban

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**, Auto Increment
ai_simulasi_id	Integer	**FOREIGN KEY** (Ref: ai_simulasi.id)
ai_soal_id	Integer	**FOREIGN KEY** (Ref: ai_soal.id)
ai_opsi_id	Integer	**FOREIGN KEY** (Ref: ai_opsi.id)

Tabel 3. 8 Tabel Simulasi

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**, Manual Input
nama	Text	Nama Simulasi (misal: "Paket 1")
level	Text	*Nullable*
total_questions	Integer	Jumlah soal
durasi_waktu	Integer	Durasi (dalam menit/detik)

Tabel 3. 9 Tabel Ai_Smart

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**, Auto Increment
simulasi_id	Integer	**FOREIGN KEY** (Ref: simulasi.id)
generated_at	Long (BigInt)	Waktu generate

Tabel 3. 10 Tabel Soal

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**, Manual Input
tipe	Text	reading, structure, listening
simulasi_id	Integer	**FOREIGN KEY** (Ref: simulasi.id)
pertanyaan_text	Text	Teks Soal
reading_id	Integer	*Nullable*, **FK** (Ref: reading.id)
structure_id	Integer	*Nullable*, **FK** (Ref: structure.id)
listening_id	Integer	*Nullable*, **FK** (Ref: listening.id)
penjelasan	Text	*Nullable*

Tabel 3. 11 Tabel Opsi_Jawaban

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**, Manual Input
soal_id	Integer	**FOREIGN KEY** (Ref: soal.id)
jawaban	Text	Teks pilihan jawaban
is_true	Boolean	true jika benar, false jika salah

Tabel 3. 12 Tabel *Reading*

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**
passage_title	Text	Judul bacaan
passage_text	Text	Isi teks bacaan lengkap

Tabel 3. 13 Tabel *Structure*

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**
topic	Text	Topik grammar (misal: "Tenses")

Tabel 3. 14 Tabel Listening

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**
audio	Text	Path/Filename file audio

Tabel 3. 15 Tabel Sesi

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**, Auto Increment
user_id	Integer	**FOREIGN KEY** (Ref: user.id)
simulasi_id	Integer	**FOREIGN KEY** (Ref: simulasi.id)
started_at	Long (BigInt)	Waktu mulai
completed_at	Long (BigInt)	*Nullable* (Waktu selesai)
skor	Double	*Nullable*
durasi_pengerjaan	Integer	Durasi real user mengerjakan
tipe_ujian	Text	Tipe ujian

Tabel 3. 16 Tabel Pilihan_Jawaban

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**, Auto Increment
sesi_id	Integer	**FOREIGN KEY** (Ref: sesi.id)

soal_id	Integer	**FOREIGN KEY** (Ref: soal.id)
opsi_jawaban_id	Integer	*Nullable*, **FK** (Ref: opsi_jawaban.id)
benar	Boolean	*Nullable*

Tabel 3. 17 Tabel User

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**
username	Text	Username pengguna
email	Text	Alamat email

Tabel 3. 18 Tabel Materi

Nama Kolom	Tipe Data	Keterangan
id	Integer	**PRIMARY KEY**
judul	Text	Judul materi belajar
kategori	Text	Kategori materi
level	Text	Level kesulitan
konten	Text	Isi materi
is_completed	Boolean	Status penyelesaian

4. *Activity* Diagram

Activity Diagram digunakan untuk memvisualisasikan alur proses serta urutan aktivitas yang terjadi ketika pengguna berinteraksi dengan aplikasi. Diagram ini merepresentasikan langkah-langkah operasional dari setiap fitur yang telah dirancang di dalam aplikasi latihan toefl ini.

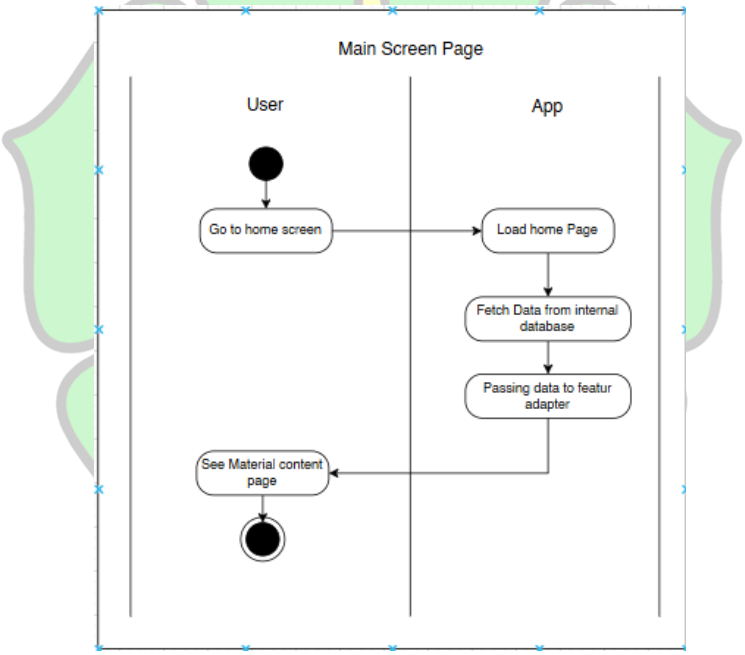
a. *Main Screen*

Gambar 3.21 menunjukkan alur aktivitas yang terjadi ketika pengguna membuka halaman utama (Main Screen) pada aplikasi latihan TOEFL. Proses dimulai ketika pengguna menekan tombol atau menu untuk menuju halaman beranda. Pada tahap awal ini, sistem akan melakukan proses load atau pemuatan halaman utama yang berisi berbagai fitur utama aplikasi.

Selanjutnya, aplikasi akan melakukan pengambilan data dari internal

database untuk memastikan seluruh informasi yang diperlukan, seperti daftar materi, kategori latihan, serta fitur lainnya, dapat ditampilkan secara lengkap. Setelah data berhasil diambil, sistem akan meneruskan data tersebut ke adapter agar dapat diolah dan ditampilkan pada antarmuka pengguna dengan format yang sesuai.

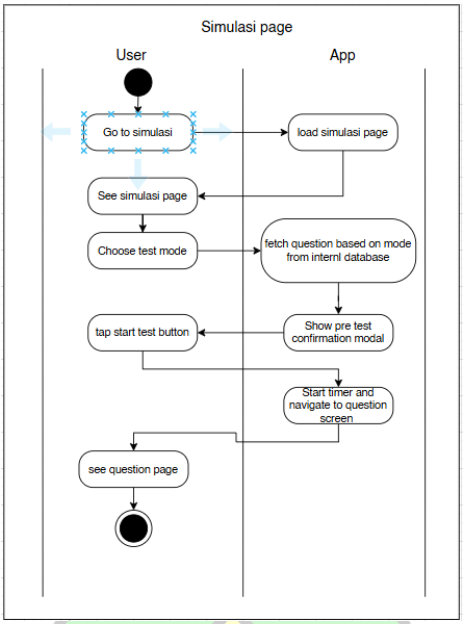
Ketika proses pemuatan selesai, pengguna dapat melihat tampilan utama yang berisi kumpulan fitur dan konten yang tersedia. Dari halaman ini, pengguna dapat melanjutkan ke halaman lain, misalnya halaman materi (Material Content Page), untuk melihat isi pembelajaran yang telah disediakan. Dengan demikian, diagram ini menggambarkan alur interaksi yang efisien antara pengguna dan aplikasi dalam menampilkan data dari sumber internal menuju antarmuka utama aplikasi.



Gambar 3. 21 *Activity* Diagram *Main Screen*

b. *Simulasi Screen*

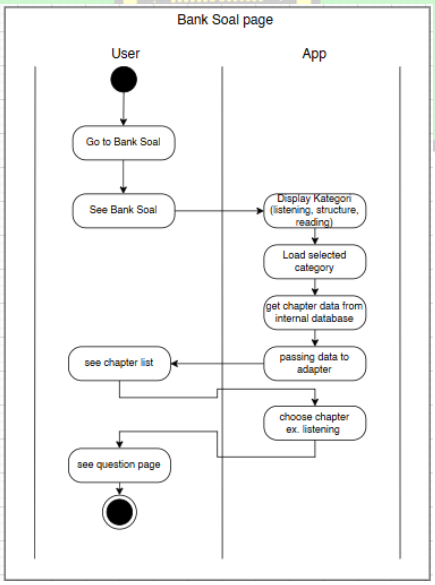
Urutan proses aktivitas ketika pengguna akan membuka Simulasi *Screen* dan melihat contoh-contoh simulasi yang tersedia pada aplikasi melalui Simulasi *Screen* yang tersedia dalam aplikasi dapat dilihat pada gambar 3.22.



Gambar 3. 22 *Activity* Diagram *Simulasi Screen*

c. Bank Soal *Screen*

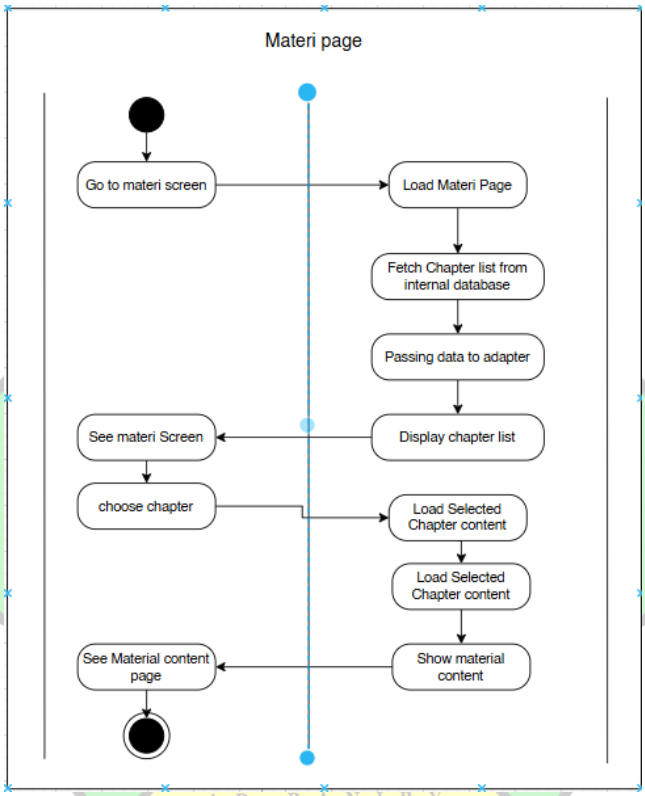
Urutan aktivitas yang terjadi ketika pengguna mengakses Bank Soal *Screen*, melalui *Main Screen* dengan cara memilih salah satu daftar menu yang tersedia pada halaman *main screen* yang dapat dilihat pada gambar 3.21.



Gambar 3. 23 *Activity* Diagram bank soal *Screen*

d. *Materi Screen*

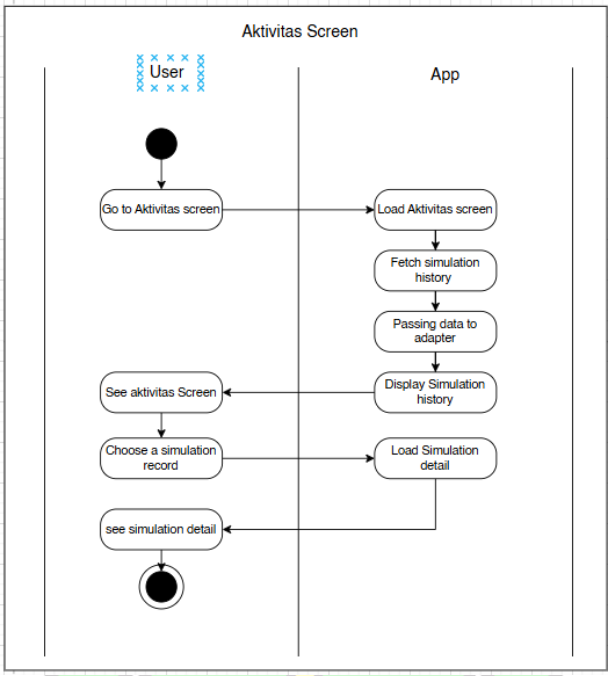
Pada Gambar 3.24 ditunjukkan urutan aktivitas yang terjadi ketika pengguna mengakses halaman materi melalui menu *Main screen*. Ketika pengguna memilih menu tersebut, sistem akan melakukan proses pengambilan data materi dari basis data internal. Data yang diperoleh kemudian diteruskan ke komponen adapter untuk diproses dan ditampilkan dalam bentuk daftar pada antarmuka halaman kuis.



Gambar 3. 24 *Activity* Diagram Materi *Screen*

e. *Activity Screen*

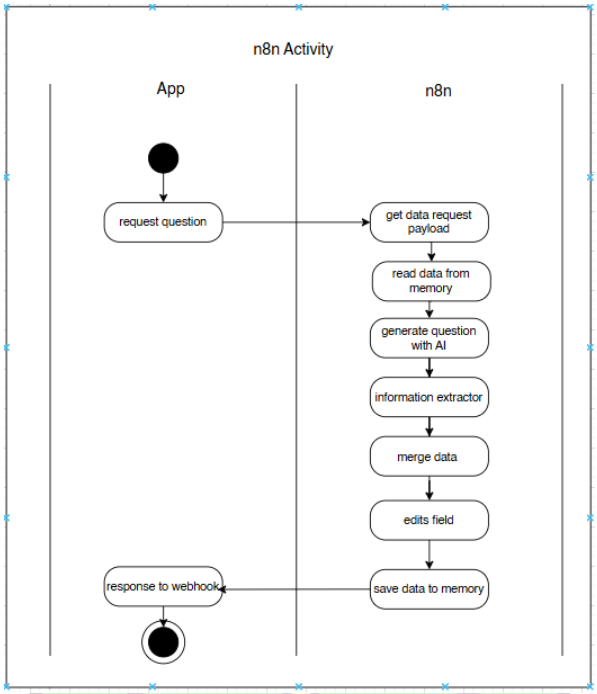
Rincian urutan aktivitas pada Aktivitas *Screen* dimulai ketika pengguna menekan *item* Aktivitas pada bottom navigation, dapat dilihat pada Gambar 3.25.



Gambar 3. 25 *Activity* Diagram Aktivitas *Screen*

f. n8n *question genarator*

Alur n8n pada diagram tersebut menggambarkan proses otomatisasi pada menu Latihan Pintar berbasis AI dalam aplikasi latihan TOEFL. Ketika pengguna di sisi aplikasi mengirimkan permintaan soal (*request question*), sistem n8n menerima permintaan tersebut melalui *get* data *request payload* dan membaca data pengguna yang dikirimkan. Selanjutnya, n8n menggunakan modul kecerdasan buatan untuk menghasilkan soal sesuai kebutuhan pengguna. Hasil keluaran AI kemudian dianalisis melalui information extractor untuk mengekstraksi informasi penting, lalu data digabung (merge data) dan dilakukan penyesuaian atau penyuntingan (*edits field*) agar sesuai format sistem. Setelah semua data siap, n8n Kn mengirimkan respons ke webhook aplikasi, sehingga pengguna dapat langsung menerima soal yang telah dihasilkan AI, adapun *activity diagram* n8n *question generator* dapat dilihat di Gambar 3.26.



Gambar 3. 26 *Activity* Diagram *n8n question genarator*

4.5. Implementasi

Implementasi dimulai pada lapisan *client* dengan pengembangan aplikasi Android berbasis Kotlin yang memanfaatkan komponen Jetpack (ViewModel, LiveData/StateFlow, Navigation, dan Room) untuk memastikan arsitektur yang terstruktur dan mudah diuji. Antarmuka dirancang mengikuti prinsip Material Design agar alur latihan Listening, Structure, dan Reading konsisten, sementara modul "Latihan Pintar" menyediakan input preferensi/riwayat belajar pengguna. Data lokal disimpan menggunakan Room untuk mode offline dan sinkronisasi diferensial ketika daring. Komunikasi dengan server dilakukan melalui Retrofit/OKHttp menggunakan format JSON, termasuk skema request untuk permintaan soal adaptif dan skema response untuk paket soal, pembahasan, serta tingkat kesulitan dan estimasi waktu.

Pada sisi server, implementasi berpusat pada API REST yang menangani autentikasi, manajemen user, bank soal, dan endpoint khusus "/ai/smart-practice". Endpoint ini terhubung ke n8n melalui webhook; n8n menerima payload permintaan, membaca profil/riwayat dari penyimpanan (misalnya Redis/PostgreSQL), mengeksekusi node AI untuk generasi atau seleksi soal,

mengekstrak atribut penting (topik, tingkat kesulitan, rasional pembahasan), melakukan penggabungan dengan data kurikulum internal, lalu menyunting bidang yang diperlukan agar sesuai skema respons. Hasilnya disimpan kembali ke memory store untuk jejak belajar dan dikirim balik ke aplikasi melalui webhook response. Pipeline ini memungkinkan adaptasi berbasis performa sebelumnya (akurasi, waktu pengerjaan, dan kelemahan topik) sesuai diagram aktivitas yang telah Anda susun.

Tahap akhir implementasi mencakup penataan repositori bank soal dan materi, penyusunan pembahasan terstruktur, serta instrumentasi telemetri untuk analitik belajar. Pada client, hasil dari n8n dipresentasikan dalam UI yang menekankan umpan balik langsung: skor sementara, pembahasan per butir, dan rekomendasi lanjutan. Mekanisme caching memastikan pengalaman mulus di koneksi terbatas, sementara kebijakan keamanan diterapkan melalui HTTPS/TLS, token berbasis JWT, validasi skema JSON di kedua sisi, serta pembatasan laju pada endpoint AI. Dengan demikian, implementasi Bab III menunjukkan keterpaduan antara aplikasi Kotlin yang robust, orkestrasi n8n yang modular, dan layanan AI yang terukur untuk mewujudkan fitur Latihan Pintar yang adaptif dan dapat dipertanggungjawabkan secara akademik.

4.6. Pengujian Sistem

Pengujian sistem adalah tahap pengujian fungsionalitas secara keseluruhan terhadap semua fitur yang telah terintegrasi. *System testing* akan dilakukan oleh pengembang dengan pendekatan yang digunakan adalah *black-box testing*, yang menempatkan penguji dalam sudut pandang pengguna akhir yang fokusnya adalah menguji aplikasi secara *end-to-end*.

Pengujian dilakukan dengan menjalankan aplikasi baik itu pada perangkat fisik maupun emulator, untuk memastikan aliran aplikasi berjalan sesuai harapan tanpa memperhatikan Detail kode program. Pengujian ini mencakup bagaimana aplikasi merespons *input* pengguna, apakah *output* yang dihasilkan sesuai dengan yang diharapkan, serta apakah aplikasi berfungsi dengan baik di lingkungan nyata.

3.4 Minimum Viable Product (MVP)

Minimum Viable Product (MVP) merupakan pendekatan yang berfokus pada pengembangan produk dengan menyediakan fitur-fitur paling esensial untuk pengguna awal. Dalam pengembangan aplikasi Latihan TOEFL pendekatan MVP diterapkan guna memastikan produk keluaran penelitian ini memiliki fungsi utama yang benar-benar mampu memenuhi kebutuhan inti pengguna.

Pendekatan MVP ini memungkinkan aplikasi Latihan TOEFL diuji langsung oleh pengguna untuk mendapatkan umpan balik yang konstruktif. Uji coba dilakukan untuk memperoleh umpan balik, guna mengevaluasi sejauh mana fitur utama aplikasi telah memenuhi kebutuhan pengguna, baik dari sisi kelayakan, kemudahan penggunaan, maupun kebermanfaatan. Dalam praktiknya, umpan balik terhadap MVP dapat diperoleh melalui berbagai metode, salah satunya adalah pemberian *rating* dari pengguna awal.

Pada penelitian ini, pemberian *rating* dilakukan menggunakan skala Likert. Skala Likert adalah metode penilaian yang digunakan untuk mengukur tingkat kesesuaian atau pendapat responden terhadap suatu pernyataan berdasarkan tingkatan tertentu. Skor diberikan dalam rentang nilai 1 hingga 5, dengan kriteria sebagaimana tercantum pada tabel berikut:

Tabel 3. 19 Skala Penilaian *Rating* Aplikasi

Skor	Kategori Penilaian	Keterangan Penilaian
1	Sangat Tidak Sesuai	Fitur tidak berfungsi, sulit digunakan, dan tidak memberi manfaat sama sekali
2	Tidak Sesuai	Fitur kurang berjalan dengan baik atau masih membingungkan
3	Cukup Sesuai	Fitur berjalan namun masih kurang nyaman atau belum optimal kegunaannya
4	Sesuai	Fitur berjalan baik, mudah digunakan, dan memberi manfaat yang jelas
5	Sangat Sesuai	Fitur sangat membantu, mudah diakses, dan sesuai dengan kebutuhan pengguna

BAB IV
PEMBAHASAN DAN HASIL

4.1 *Implementation*

Subbab ini memaparkan hasil realisasi sistem beserta rincian teknisnya. Secara garis besar, tahapan implementasi diawali dengan melakukan konversi desain menjadi program menggunakan bahasa pemograman kotlin di mana penulis menerjemahkan rancangan antarmuka dan spesifikasi fitur yang telah didefinisikan pada bab sebelumnya menjadi baris kode program menggunakan lingkungan pengembangan Android Studio.

Setelah proses konversi dilakukan maka penulis selanjutnya akan melakukan inisialisasi *database* internal aplikasi menggunakan Room, Room merupakan *library* yang digunakan untuk menyimpan dan mengelola basis data lokal pada aplikasi Android dengan menyederhanakan kompleksitas SQLite. Setelah itu penulis melakukan penulisan kode logika pemograman seperti *viewmodel* dan *repository* yang jika sudah selesai maka akan dilakukan proses integrasi antara UI dan juga logika pemograman yang telah dibuat.

Setelah logika aplikasi dan UI aplikasi telah berhasil di integrasikan dengan sempurna maka selanjutnya penulis akan melakukan pembuatan *workflow n8n* berdasarkan activity diagram yang telah di rancang pada bab sebelumnya, setelah pembuatan workflow n8n selesai maka akan dilakukan pengintegrasian, detail implementasi setiap bagian akan di jelaskan pada subbab berikut.

4.1.1. Implementasi *Database*

Berikut ini adalah hasil antarmuka aplikasi Latihan TOEFL yang merupakan hasil konversi dari desain *wireframe* sebelumnya. Alur aktivitas sistem disesuaikan dengan *Activity Diagram* yang telah dirancang, sementara logika pemrogramannya dibangun menggunakan bahasa Kotlin dengan pendekatan deklaratif menggunakan *Jetpack Compose*. Seluruh elemen antarmuka tersebut telah diintegrasikan dengan logika kode yang sesuai. Rincian implementasi desain ini akan diuraikan lebih lanjut pada poin berikut berdasarkan fitur dan halaman yang telah dibuat

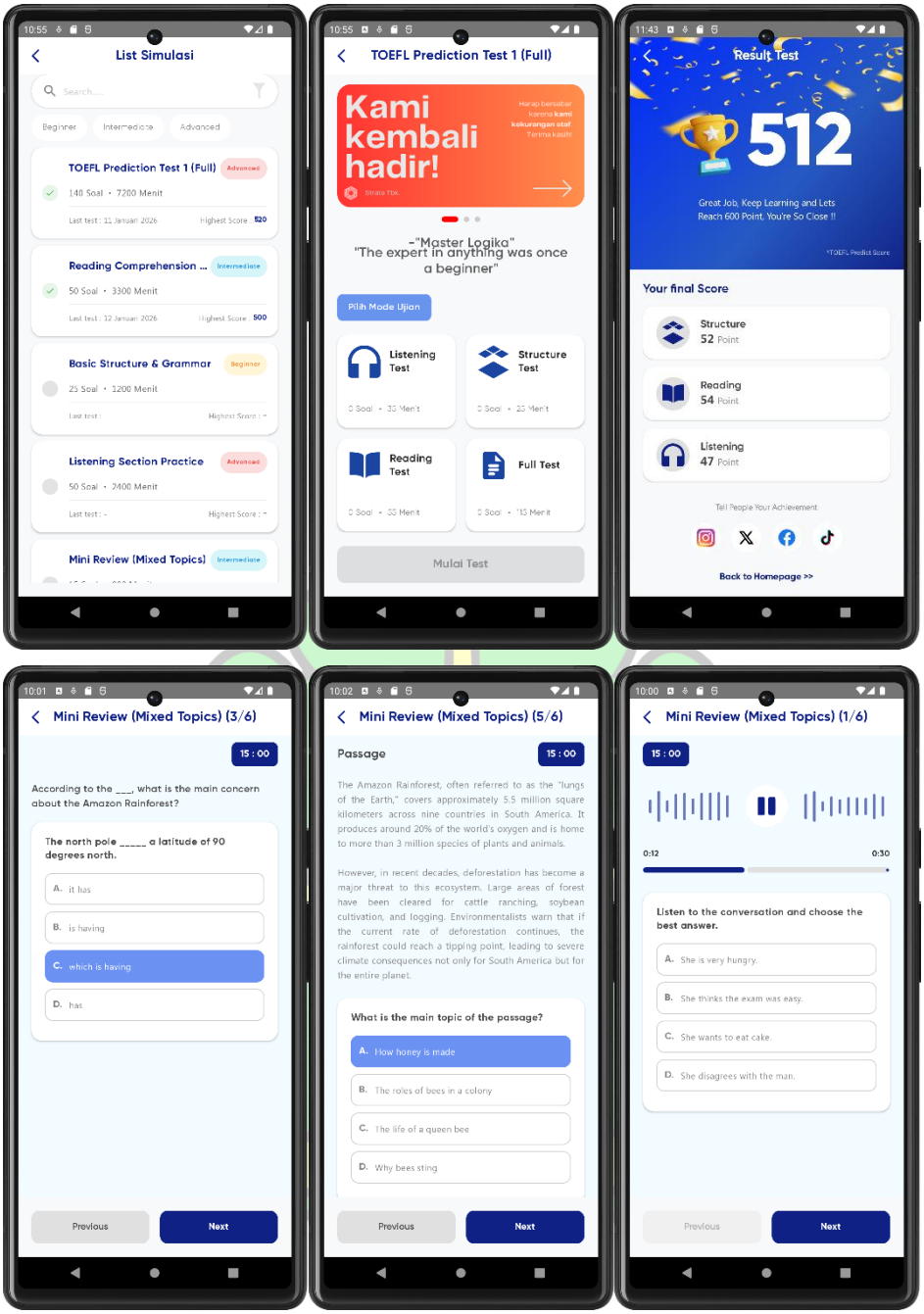
a. *Main Screen*

Tampilan awal atau *Main Screen* adalah halaman pertama yang akan dilihat pengguna ketika membuka aplikasi Latihan TOEFL. Antarmuka ini menyajikan beberapa *shortcut* berupa tombol navigasi menuju fitur-fitur utama, yakni Materi, Simulasi, Bank Soal, dan Latihan Pintar. Implementasi tampilan *Main Screen* dapat dilihat pada Gambar 4.1.

Gambar 4. 1 Implementasi *Main Screen*

b. Simulasi *Screen*

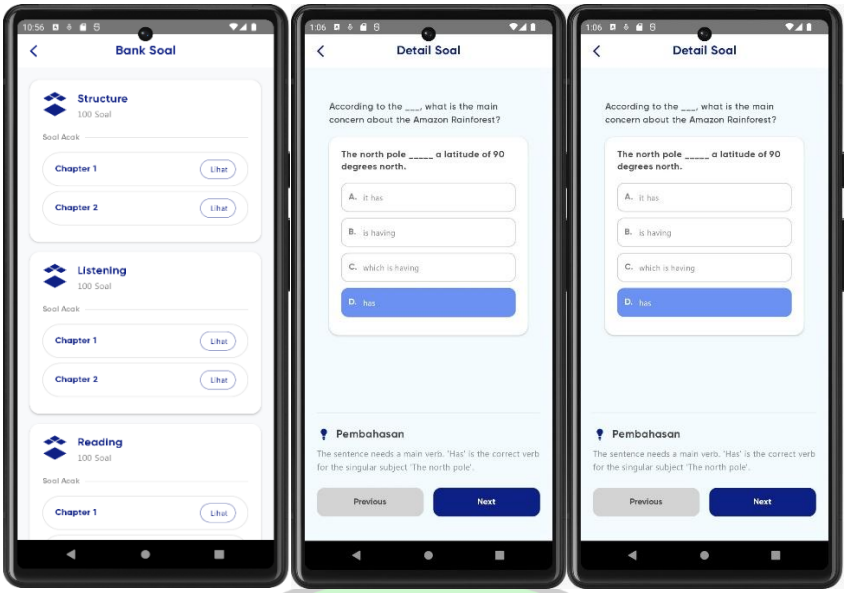
Tampilan ketika pengguna menekan *shortcut* Simulasi di halaman utama. Antarmuka ini menampilkan list simulasi yang tersedia dan detail dari simulasi tersebut beserta tampilan saat simulasi di lakukan. Implementasi Simulasi *Screen* dapat dilihat pada Gambar 4.2.



Gambar 4. 2 Implementasi Simulasi *Screen*

c. Bank Soal *Screen*

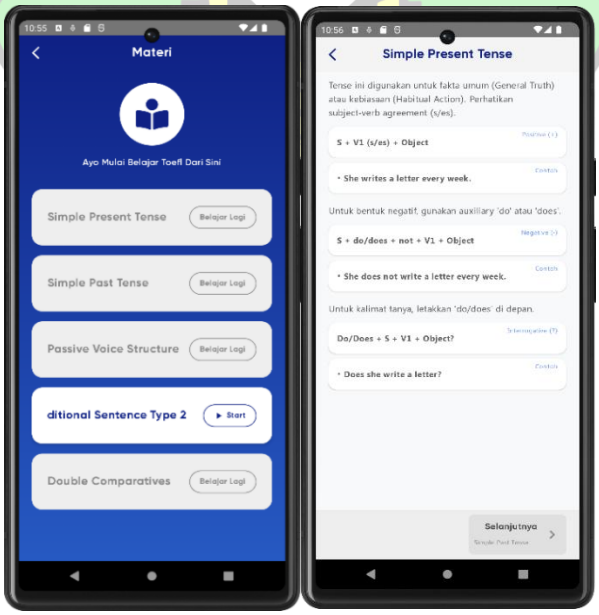
Antarmuka ini menampilkan *list* seluruh soal yang tersedia dan detail dari soal tersebut yang disertai pembahasan tentang soalnya. Implementasi Bank Soal *Screen* dapat dilihat pada Gambar 4.3.



Gambar 4. 3 Implementasi Bank Soal *Screen*

d. Materi *Screen*

Halaman ini akan menampilkan *list* materi yang tersedia untuk pengguna belajar serta detail dari materi yang dipilih untuk pengguna, Implementasi Materi *Screen* dapat dilihat pada Gambar 4.4.

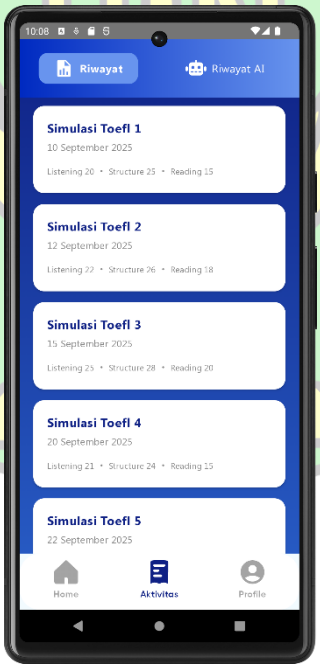


Gambar 4. 4 Implementasi Materi *Screen*

e. *Activity Screen*

Pada halaman ini, pengguna dapat melihat riwayat seluruh simulasi yang pernah dikerjakan sebelumnya. Halaman riwayat ini berfungsi sebagai media dokumentasi dan evaluasi, sehingga pengguna dapat meninjau kembali hasil simulasi yang telah dilakukan tanpa harus mengulangi proses dari awal. Implementasi *Activity Screen* pada halaman riwayat simulasi dapat dilihat pada Gambar 4.5.

Riwayat simulasi pada halaman ini dibagi ke dalam dua kategori utama, yaitu riwayat simulasi biasa dan riwayat simulasi yang dihasilkan menggunakan fitur latihan pintar. Pemisahan ini bertujuan untuk memberikan kejelasan kepada pengguna mengenai sumber dan metode simulasi yang digunakan, sekaligus memudahkan proses pencarian serta analisis hasil simulasi.



Gambar 4. 5 Implementasi *Activity Screen*

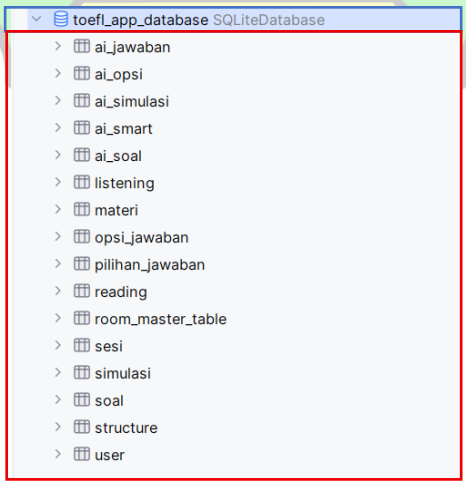
4.1.2. Implementasi *Database*

Basis data internal pada aplikasi Latihan TOEFL diimplementasikan menggunakan *library* Room, yang merupakan bagian dari Android *Jetpack*. Room berperan sebagai lapisan abstraksi di atas *SQLite* sehingga proses pengelolaan dan akses data lokal dapat dilakukan secara lebih terstruktur, aman, dan efisien oleh pengembang aplikasi.

Struktur basis data yang digunakan merupakan hasil implementasi dari rancangan *Databse Schema* pada BAB III dengan masing-masing entitas merepresentasikan tabel yang berfungsi untuk menyimpan data sesuai dengan kebutuhan fitur dalam aplikasi.

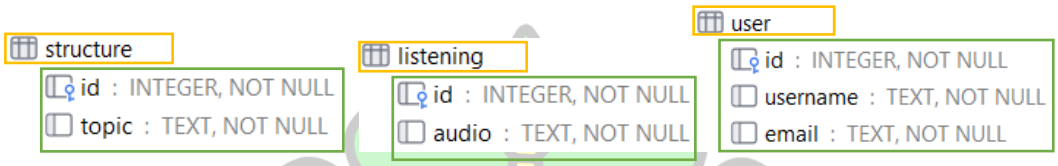
Untuk memastikan efisiensi dan konsistensi akses data, Room *database* diinisialisasi menggunakan pola *singleton*, sehingga satu *instance* basis data dapat digunakan secara bersama di seluruh bagian aplikasi tanpa menimbulkan inisialisasi berulang yang berpotensi memengaruhi kinerja sistem.

Visualisasi hasil implementasi struktur basis data internal aplikasi diperoleh melalui fitur *Database Inspector* pada Android Studio, sebagaimana ditunjukkan pada Gambar 4.6. Pada tampilan tersebut, kolom berwarna biru menunjukkan representasi *database* yang telah di buat, sedangkan kolom berwarna merah menandakan tabel atau entitas yang akan digunakan dalam proses penyimpanan data secara lokal.

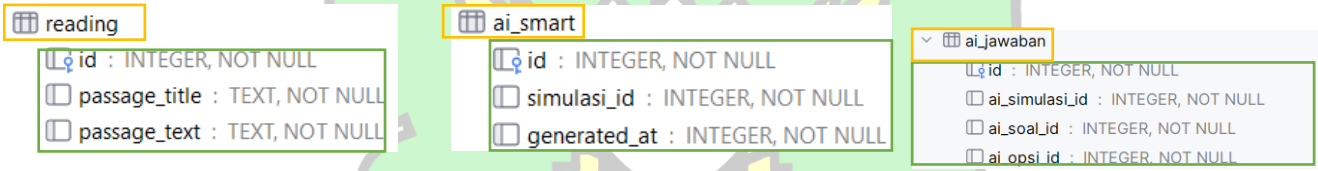


Gambar 4. 6 Implementasi *Activity Screen*

Detail *field* tiap tabel dapat dilihat pada gambar berikut dengan penanda berwarna kuning menunjukkan nama tabel yang digunakan dalam basis data, sedangkan penanda berwarna hijau merepresentasikan *field* atau atribut yang dimiliki oleh masing-masing tabel. Seluruh nama tabel dan *field* yang diimplementasikan telah disesuaikan dengan rancangan *Database Schema* pada BAB III, sehingga menunjukkan kesesuaian antara tahap perancangan dan tahap implementasi. Kesesuaian ini memastikan bahwa struktur data yang digunakan mampu mendukung seluruh kebutuhan fungsional aplikasi Latihan TOEFL.

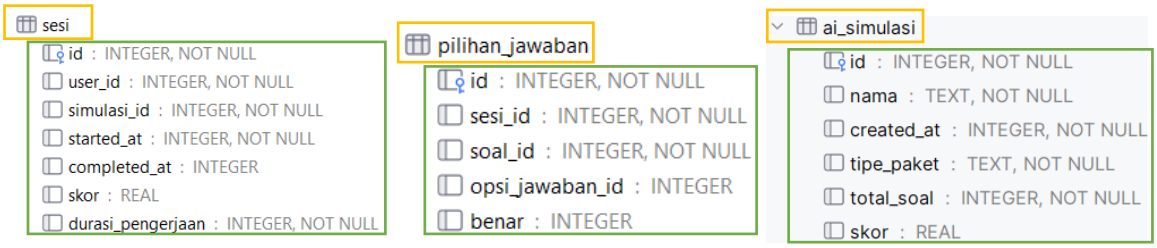


Gambar 4. 7 Implementasi tabel *structure*, *listening* dan *user*

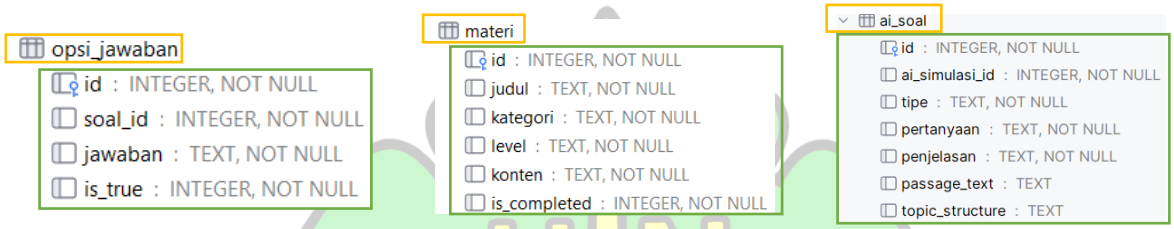


Gambar 4. 8 Implementasi tabel *reading*, *ai_smart, ai_jawaban*

Gambar 4. 9 Implementasi tabel soal, simulasi dan *ai_opsi*



Gambar 4. 10 Implementasi tabel sesi, pilihan jawaban dan ai_simulasi



Gambar 4. 11 Implementasi tabel opsi_jawaban, materi dan ai_soal

4.1.3. Implementasi *n8n soal generation*

Berdasarkan hasil perancangan *use case* dan *activity diagram* yang telah dijelaskan pada bab sebelumnya, pada tahap ini penulis melakukan implementasi terhadap workflow n8n yang berfungsi sebagai tools untuk membuat soal simulasi secara otomatis. Workflow n8n ini dirancang untuk menghasilkan simulasi secara dinamis sesuai dengan pilihan dan parameter yang dipilih oleh pengguna pada fitur Latihan Pintar di dalam aplikasi.

Setiap permintaan simulasi yang dilakukan oleh pengguna akan dikirimkan dari aplikasi klien ke n8n melalui mekanisme komunikasi webhook berbasis REST API menggunakan metode POST. Data yang dikirimkan mencakup parameter pilihan latihan, tingkat kesulitan, serta informasi pendukung lain yang diperlukan dalam proses simulasi.

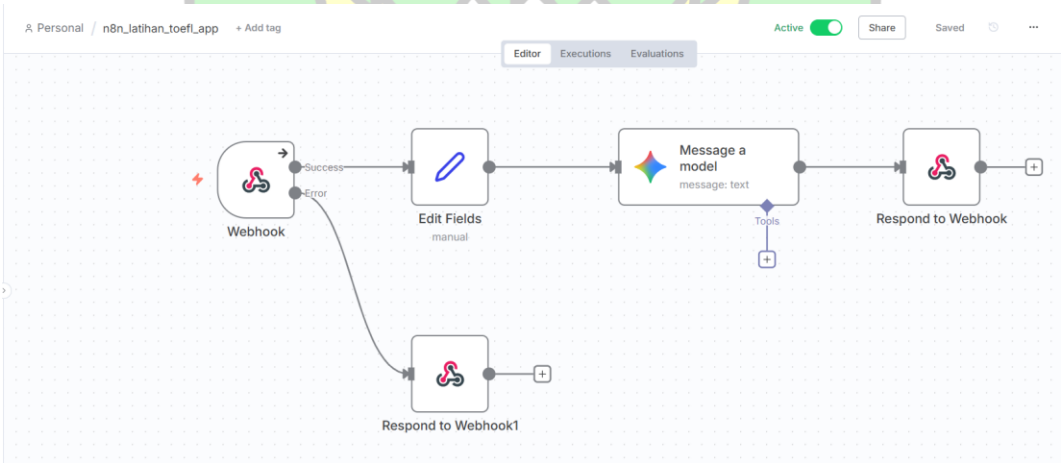
Selanjutnya, webhook pada n8n akan menerima permintaan tersebut dan memicu rangkaian node workflow yang telah dirancang, mulai dari proses validasi

data, untuk menghasilkan output simulasi yang sesuai. Hasil pemrosesan kemudian dikembalikan ke aplikasi klien dalam bentuk json untuk ditampilkan kepada pengguna

Detail lebih lanjut mengenai mekanisme integrasi antara aplikasi klien dan n8n, termasuk struktur data yang dikirimkan, alur eksekusi workflow, serta pengolahan respons dari server, akan dibahas secara lebih mendalam pada poin di dalam subbab ini.

1. *n8n Workflow Implementation*

Pada tahap ini, *workflow* n8n diimplementasikan sebagai *tools* untuk melakukan pembuatan soal simulasi secara otomatis dengan memanfaatkan node Google Gemini. *Workflow* ini menjadi penghubung antara aplikasi Latihan TOEFL dan layanan AI sehingga sistem dapat menghasilkan soal secara dinamis berdasarkan parameter yang dipilih pengguna pada fitur Latihan Pintar. Visualisasi *workflow* dapat dilihat pada Gambar 4.12.



Gambar 4. 12 *node n8n workflow*

Node yang digunakan dalam *workflow* beserta fungsinya adalah sebagai berikut:

a. *Webhook* (*Trigger*)

Node ini berfungsi sebagai titik awal *workflow* yang menerima permintaan

dari aplikasi menggunakan metode *POST*. Parameter yang diterima berupa jenis latihan, tingkat kesulitan, dan jumlah soal

b. *Set* / *Edit Fields* (*Mapping Parameter*)

   Node ini digunakan untuk melakukan pemetaan nilai parameter dari *request* menjadi variabel yang lebih terstruktur. Tahap ini penting agar data yang diterima memiliki format konsisten dan siap digunakan untuk membangun *prompt*.

c. Google Gemini (*AI Generation Node*)

   Node Gemini merupakan komponen utama untuk melakukan proses generasi soal. Pada node ini, prompt yang telah disiapkan dikirimkan ke model Gemini untuk menghasilkan output soal. Hasil yang diperoleh berupa teks/JSON yang berisi pertanyaan, pilihan jawaban serta kunci jawaban.

d. *Respond to Webhook (Return JSON Response)*

   Node ini mengirimkan hasil akhir generasi soal ke aplikasi klien dalam bentuk JSON *response*. Data tersebut kemudian ditampilkan pada halaman simulasi, dan dapat disimpan sebagai riwayat jika fitur tersebut diaktifkan.

Dengan menggunakan *node* Gemini, implementasi integrasi AI menjadi lebih sederhana dan terintegrasi langsung di dalam n8n, tanpa memerlukan konfigurasi *endpoint* manual sebagaimana pendekatan *HTTP request*.

2. *Integrasi API Client Side*

Integrasi antara aplikasi dan n8n dilakukan melalui komunikasi *REST API* menggunakan metode POST dari aplikasi Android. Pada sisi client, implementasi koneksi jaringan dibangun menggunakan *library Retrofit* karena mendukung pemanggilan *API* secara terstruktur, mudah dipelihara, serta kompatibel dengan proses parsing data *JSON*. Secara umum, implementasi integrasi *API* pada sisi client terdiri dari dua komponen utama, yaitu *API Service* dan *API Config*

a. *API Service*

Komponen ini berfungsi sebagai *interface* yang mendefinisikan seluruh *endpoint* yang dapat diakses oleh aplikasi. API *Service* menentukan struktur permintaan HTTP seperti metode yang akan digunakan contohnya seperti GET dan POST, format parameter, jenis *payload*, serta tipe data yang diharapkan sebagai respons. Sedangkan rincian implementasi integrasi *endpoint* API *Service* yang digunakan dengan aplikasi dapat dilihat pada tabel 4.1.

Tabel 4. 1 Integrasi API *Service* TOEFL

No	*Endpoint*	API *Service*	Hasil

1	https://{*webhook*}/ *generate-soal*	`@POST( value = "generate-soal")`
		`suspend fun sendResultToServer(`
		`  @Body request: GenerateSoalRequest`
		`): GenerateSoalResponse`
			Terintegrasi

b. *API Config*

API *Config* berfungsi sebagai konfigurasi utama dalam membangun objek Retrofit yang akan digunakan untuk semua permintaan HTTP. Komponen ini bertanggung jawab untuk menentukan *base* URL dari *n8n* yang kita gunakan, mengatur konverter JSON, serta mengelola aspek penting lain seperti *timeout*, *interceptor*, dan *logging* bila diperlukan. Detail struktur API *Config* dapat dilihat pada Gambar 4.13.

ApiConfig.kt × ApiService.kt

```kotlin
package com.pandu.latihantoefl.networking

import ...

2 Usages
class ApiConfig {
    1 Usage
    companion object {
        1 Usage
        fun getPanduGeneralApiService(): ApiService {
            val loggingInterceptor = HttpLoggingInterceptor()
                .setLevel(HttpLoggingInterceptor.Level.BODY)
            val client = OkHttpClient.Builder()
                .connectTimeout( timeout = 2,  unit = TimeUnit.SECONDS)
                .readTimeout( timeout = 120,  unit = TimeUnit.SECONDS)
                .writeTimeout( timeout = 120,  unit = TimeUnit.SECONDS)
                .addInterceptor(loggingInterceptor)
                .build()
            val retrofit = Retrofit.Builder()
                .baseUrl( baseUrl = "https://n8n-jtmoastu.ap-southeast-1.clawcloudrun.com/webhook/")
                .addConverterFactory( factory = GsonConverterFactory.create())
                .client(client)
                .build()
            return retrofit.create(ApiService::class.java)
        }
    }
}
```

Gambar 4. 13 *API Config* Latihan TOEFL

4.2 Pengujian Sistem

Setelah seluruh proses implementasi kode program pada aplikasi Latihan TOEFL selesai dilakukan, tahapan berikutnya adalah *system testing*. Pengujian sistem ini dirancang berdasarkan dua skenario utama, yaitu: (1) skenario pengguna baru (*first-time user*), skenario ini menguji perilaku aplikasi ketika pengguna pertama kali membuka aplikasi sehingga belum memiliki data lokal maupun riwayat latihan, serta memastikan pengguna dapat mengakses dan menavigasi fitur utama dari *Main Screen* yang menyediakan *shortcut* menuju Materi, Simulasi, Bank Soal, dan Latihan Pintar; (2) skenario pengguna lama (*returning user*), skenario ini menguji alur aplikasi ketika pengguna telah menggunakan aplikasi sebelumnya sehingga data lokal dan riwayat latihan telah tersimpan, termasuk kemampuan aplikasi menampilkan riwayat simulasi serta pemisahan riwayat antara simulasi biasa dan simulasi dari fitur latihan pintar/AI pada halaman *Activity*.

Skenario pengujian mencakup seluruh test case yang tertera pada Tabel 4.2. *System testing* dilakukan oleh pengembang dengan menguji aplikasi dari sudut pandang pengguna secara menyeluruh (*end-to-end*) menggunakan metode *black-box*. Pengujian *black-box* berfokus pada verifikasi fungsionalitas sistem berdasarkan masukan dan keluaran yang tampak pada antarmuka, tanpa memerlukan pemahaman terhadap struktur internal maupun detail implementasi kode program. Pengujian dilakukan dengan menjalankan aplikasi pada perangkat fisik maupun emulator terhadap sistem yang telah terintegrasi secara menyeluruh.

Tabel 4. 2 *System testing*

Jenis Skenario	Feature	Pre-Condition	Test Case	Test Steps	Test Data	Expected Result	Result
First-time user	Main	Aplikasi terpasang, belum pernah digunakan, belum ada data/riwayat lokal	Aplikasi terbuka normal (cold start)	1) Buka aplikasi; 2) Tunggu sampai Main Screen tampil	-	Main Screen tampil tanpa crash dan shortcut menu terlihat	Pass
First-time user	Main	Main Screen tampil	Navigasi ke Materi melalui shortcut	1) Tap “Materi”	-	Berpindah ke halaman Materi	Pass
First-time user	Main	Main Screen tampil	Navigasi ke Simulasi melalui shortcut	1) Tap “Simulasi”	-	Berpindah ke halaman Simulasi	Pass
First-time user	Main	Main Screen tampil	Navigasi ke Bank Soal melalui shortcut	1) Tap “Bank Soal”	-	Berpindah ke halaman Bank Soal	Pass
First-time user	Main	Main Screen tampil	Navigasi ke Latihan Pintar/AI melalui shortcut	1) Tap “Latihan Pintar/AI”	-	Berpindah ke halaman Latihan Pintar/AI	Pass

Tabel 4. 2 *System testing*

Jenis Skenario	Feature	Pre-Condition	Test Case	Test Steps	Test Data	Expected Result	Result
First-time user	Materi	Pengguna belum memiliki riwayat, data materi tersedia	Materi tampil dan dapat dibuka	1) Masuk Materi; 2) Pilih salah satu materi 3) Buka detail materi	Materi A	Daftar materi tampil dan detail materi berhasil dibuka	Pass
First-time user	Simulasi	Pengguna belum memiliki riwayat simulasi	Memulai simulasi dengan konfigurasi valid	1) Masuk Simulasi 2) Pilih parameter 3) Tap “Mulai”	Section: Reading, Level: Medium, Jumlah: 10	Simulasi dimulai dan soal tampil sesuai parameter	Pass
First-time user	Simulasi	Simulasi sedang berjalan	Menjawab soal dan berpindah nomor	1) Pilih jawaban 2) Tap Next 3) Ulangi beberapa kali	Jawaban acak	Jawaban tersimpan dan navigasi antar soal berjalan normal	Pass
First-time user	Simulasi	Simulasi sedang berjalan	Validasi jika jawaban belum dipilih (jika diwajibkan)	1) Tap Next tanpa memilih jawaban	-	Muncul peringatan/ pengguna tidak dapat lanjut (sesuai aturan aplikasi)	Pass

Tabel 4. 2 *System testing*

Jenis Skenario	Feature	Pre-Condition	Test Case	Test Steps	Test Data	Expected Result	Result
First-time user	Simulasi	Soal terakhir selesai dikerjakan	Menyelesaikan simulasi dan melihat hasil	1) Tap "Selesai/Submit"	-	Halaman hasil/rekap nilai tampil	Pass
First-time user	Bank Soal	Pengguna baru, data bank soal tersedia	Daftar bank soal tampil	1) Masuk Bank Soal	-	Daftar kategori/paket soal tampil normal	Pass
First-time user	Bank Soal	Halaman Bank Soal terbuka	Membuka soal berdasarkan kategori	1) Pilih kategori 2) Pilih paket/soal	Kategori: Listening	Soal pada kategori terbuka dan dapat dikerjakan	Pass
First-time user	Latihan Pintar/AI	Pengguna baru, koneksi tersedia	Generate latihan AI dengan parameter valid	1) Masuk Latihan AI 2) Pilih parameter 3) Tap "Generate /Mulai"	Tipe: Grammar, Level: Easy, Jumlah: 5	Soal AI berhasil dibuat dan ditampilkan sesuai parameter	Pass

Tabel 4. 2 *System testing*

Jenis Skenario	Feature	Pre-Condition	Test Case	Test Steps	Test Data	Expected Result	Result
First-time user	Latihan Pintar/AI	Parameter latihan belum lengkap	Validasi input parameter	1) Kosongkan salah satu parameter 2) Tap "Generate/Mulai"	Jumlah soal kosong	Muncul pesan validasi bahwa parameter wajib diisi	Pass
Returning user	Main	Aplikasi pernah digunakan, data lokal/riwayat sudah tersimpan	Aplikasi terbuka dan menampilkan kondisi terakhir	1) Buka aplikasi 2) Amati tampilan awal	-	Aplikasi terbuka normal dan tetap stabil dengan data lokal tersedia	Pass
Returning user	Activity	Terdapat riwayat simulasi	Riwayat simulasi tampil pada Activity	1) Masuk Activity 2) Lihat daftar riwayat simulasi	-	Riwayat simulasi tampil sesuai data yang tersimpan	Pass
Returning user	Activity	Terdapat riwayat latihan AI	Riwayat latihan AI tampil pada Activity	1) Masuk Activity 2) Lihat daftar riwayat AI	-	Riwayat latihan AI tampil dan terpisah dari riwayat simulasi (jika ada pemisahan)	Pass
Returning user	Activity	Riwayat tersedia	Membuka detail riwayat	1) Tap salah satu item riwayat	Item riwayat 1	Detail riwayat tampil (skor/ringkasan/jawaban sesuai aplikasi)	Pass

Tabel 4. 2 *System testing*

Jenis Skenario	Feature	Pre-Condition	Test Case	Test Steps	Test Data	Expected Result	Result
Returning user	Profil	Data profil/konfigurasi tersimpan (jika ada)	Profil menampilkan data yang tersimpan	1) Masuk Profil	-	Data profil/konfigurasi tampil sesuai data terakhir	
Returning user	Global	Data lokal tersedia	Konsistensi data setelah penggunaan berulang	1) Jalankan simulasi 2) Jalankan latihan AI 3) Cek Activity	-	Riwayat bertambah sesuai aktivitas, tidak duplikat/tertukar	

4.3 *Minimum Viable Product* (MVP)

Sebagaimana telah dijelaskan pada Bab III, pendekatan MVP diterapkan untuk memvalidasi fungsi esensial aplikasi sejak tahap awal. Guna memperoleh umpan balik yang mendalam namun efisien, uji coba difokuskan pada kelompok responden terbatas berjumlah lima sampai sepuluh orang. Jumlah ini dinilai memadai secara metodologis untuk mencapai saturasi data dan mengungkap sebagian besar kendala *usability* utama tanpa mengalami redundansi informasi.

Para responden diminta memberikan penilaian terhadap tiga aspek krusial merujuk pada (Asroni et al., 2024). yaitu: (1) kelayakan produk, (2) kemudahan penggunaan, dan (3) manfaat fitur. Penilaian dilakukan menggunakan skala Likert 1 sampai 5 dengan ketentuan sebagaimana tercantum pada Tabel 4.3. Hasil penilaian dari masing-masing responden disajikan pada tabel berikut.

Tabel 4. 3 Hasil Penilaian Responden

No	Nama Responden	Kelayakan Produk	Kemudahan Penggunaan	Manfaat Fitur	Rata-Rata
1	Responden A	5	4	4	4.33
2	Responden B	4	5	5	4.67
3	Responden C	4	4	4	4.00
4	Responden D	5	5	5	5.00
5	Responden E	4	4	5	4.33
Rata-rata		**4.4**	**4.4**	**4.6**	**4.46**

Berdasarkan data tersebut, dapat disimpulkan bahwa aplikasi Latihan TOEFL telah memenuhi kriteria sebagai *Minimum Viable Product* (MVP). Hal ini ditunjukkan oleh nilai rata-rata pada seluruh aspek penilaian yang berada di atas skor 4, sehingga dapat diartikan bahwa fitur utama aplikasi berjalan dengan baik, memberikan manfaat yang dirasakan pengguna, serta mudah digunakan dan diakses oleh pengguna awal.

4.4 Tabel Perbandingan Hasil Penelitian

Dalam tahap pengembangan aplikasi ini, penulis melakukan tinjauan terhadap berbagai literatur dan penelitian sejenis yang telah dilakukan sebelumnya. Kajian ini bertujuan untuk memetakan perkembangan teknologi dalam media pembelajaran TOEFL serta mengidentifikasi metode, kelebihan, dan kekurangan dari sistem yang sudah ada. Berdasarkan tinjauan tersebut, penulis merangkum perbandingan antara penelitian terdahulu dengan penelitian yang diajukan (*proposed research*) untuk memperjelas posisi dan kebaruan (*novelty*) dari penelitian ini. Rangkuman perbandingan tersebut disajikan dalam Tabel 4.4 berikut:

Tabel 4. 4 Perbandingan hasil Penelitian

No	Judul Penelitian	Metode	Kelebihan (Hasil Penelitian)	Kekurangan	Perbedaan Dengan Penelitian Ini
1	Eva Sulistiana et al., (2024)	*Android (Mobile Learning)*	Efektif meningkatkan pemahaman grammar (kategori "Poor" turun drastis).	Fokus hanya pada peningkatan skor grammar, fitur soal cenderung statis.	**Integrasi AI (n8n):** Aplikasi Anda tidak hanya statis, tapi men-generate soal baru secara otomatis menggunakan AI workflow.
2	Arsalna Furqan et al., (2023)	*Android (Simulasi)*.	Usability tinggi (Skor SUS 80,2 - Excellent) & Validasi materi tinggi.	Fitur analisis hasil masih sederhana, belum ada rekomendasi remedial cerdas.	**Analisis Remedial:** Memberikan rekomendasi remedial spesifik berdasarkan pola kesalahan pengguna.
3	Tengku Mhd. Zulfikar et al.,(2024)	*Android + Algoritma LCG*	Menggunakan LCG untuk pengacakan soal (randomization) agar tidak monoton.	Pengacakan soal hanya mengacak urutan dari database yang sudah ada, bukan membuat soal baru.	**Generative AI:** Soal bukan sekadar diacak, tetapi *dibuat baru* (generated) oleh AI sehingga variasi tidak terbatas.

No	Judul Penelitian	Metode	Kelebihan (Hasil Penelitian)	Kekurangan	Perbedaan Dengan Penelitian Ini
4	Yuli Yana Astuti et al., (2024)	*Website (Web-based)*	Memudahkan akses mahasiswa via browser tanpa instalasi.	Harus selalu *online* (tergantung internet) dan kurang fleksibel dibanding mobile apps.	**Hybrid (Online & Offline):** Aplikasi Anda mendukung mode offline untuk materi/bank soal yang sudah diunduh.
5	Wandi Aprianto, (2025)	Web (Laravel, Vue.js)	Penilaian otomatis (auto-grading) yang cepat dan akurat sesuai standar TOEFL.	Tidak spesifik menyebutkan fitur adaptif atau personalisasi soal per pengguna.	**Latihan Pintar (Adaptive):** Menyediakan fitur "Latihan Pintar" yang menyesuaikan soal dengan parameter pengguna.

4.5 Tabel Perbandingan Hasil Aplikasi Serupa

Selain meninjau penelitian akademis, penulis juga melakukan studi komparasi terhadap beberapa aplikasi pembelajaran bahasa Inggris dan simulasi TOEFL populer yang telah tersedia di masyarakat, seperti Duolingo, TOEFL Go!, dan Elsa Speak. Analisis ini bertujuan untuk memetakan fitur yang sudah ada serta mengidentifikasi keterbatasan yang sering dikeluhkan pengguna, seperti kewajiban koneksi internet (*online only*) atau materi yang tidak spesifik pada format TOEFL. Berikut adalah tabel perbandingan fitur antara aplikasi yang dirancang dengan aplikasi sejenis, perbandingan nya bisa di lihat pada tabel 4.5 berikut :

Tabel 4. 5 Perbandingan Hasil Aplikasi Serupa

No	Aspek komparasi	Aplikasi Kompetitor	Aplikasi Rancangan Saya
1	Fokus Materi	Umumnya *General English (Duolingo)* atau *Speaking (Elsa Speak)*. Materi TOEFL seringkali berbayar*/premium.*	Spesifik untuk TOEFL (Listening, Structure, Reading) dengan format akademik.
2	Teknologi AI	Menggunakan AI untuk *Speech* Recognition atau adaptasi level kesulitan dari bank soal yang ada.	Menggunakan Workflow n8n & Generative AI untuk memproduksi (generate) soal latihan baru secara otomatis.
3	Konektivitas	Mayoritas Wajib Online (Full Online). Tidak bisa digunakan jika internet putus.	Hybrid (Online & Offline). Materi dan bank soal dapat diunduh dan dikerjakan tanpa koneksi internet.
4	Fitur Analisis	Menampilkan skor akhir dan jawaban benar/salah.	Menampilkan skor + Pembahasan Rinci (Kenapa jawaban itu benar/salah) + Rekomendasi perbaikan.
5	Personalisasi	Pola latihan cenderung repetitif (mengulang soal yang sama).	Fitur "Latihan Pintar" memungkinkan pengguna mengatur parameter (topik, level, jumlah soal) sendiri.
6	Biaya	*Freemium* (Gratis terbatas, banyak fitur terkunci jika tidak langganan).	Gratis (Open Access) untuk tujuan edukasi/penelitian.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan uraian pada bab sebelumnya, maka dapat diambil kesimpulan sebagai berikut:

1. Penelitian ini berhasil merancang dan membangun aplikasi Latihan TOEFL berbasis Android menggunakan Kotlin. Aplikasi dapat digunakan dalam kondisi online maupun offline, serta mampu menampilkan dan mengelola data latihan secara dinamis, sehingga pengguna tetap dapat berlatih TOEFL sesuai konten yang tersedia pada aplikasi.
2. Integrasi kecerdasan buatan (AI) pada aplikasi berhasil diterapkan untuk mendukung proses latihan TOEFL, sehingga sistem dapat membantu pengguna memperoleh latihan yang lebih responsif terhadap parameter yang dipilih (misalnya jenis latihan/tingkat kesulitan/jumlah soal) dan meningkatkan efektivitas latihan mandiri.

   Secara keseluruhan, implementasi yang dibangun telah memenuhi tujuan penelitian dalam menyediakan media latihan TOEFL yang praktis dan dapat digunakan sebagai sarana pendukung pembelajaran.

5.2 Saran

Setelah melakukan penelitian ini, terdapat beberapa saran yang dapat dilakukan pada penelitian selanjutnya. Adapun saran tersebut adalah sebagai berikut:

1. Perlu adanya pengembangan mekanisme verifikasi identitas dan integrasi penyimpanan berbasis *cloud*, sehingga pengguna dapat mengakses progres latihan dan data personal mereka dari berbagai perangkat secara terintegrasi.

DAFTAR PUSTAKA

Aditia Ramadhani, Maulana Dwi Yantoro, Muhammad Farhan Akmal, Muhammad Mahfud, & Fauzi. (2025). CHATBOT OTOMATIS DENGAN N8N DAN AI UNTUK ANALISIS DATA DAN PELAPORAN HASIL. *Jurnal Riset Teknik Komputer*, *2*(2), 18–23. https://doi.org/10.69714/x1p94182

Alpian, A., & Muharom, S. (n.d.). *Positif : Jurnal Sistem dan Teknologi Informasi RANCANG BANGUN SISTEM INFORMASI PREVENTIVE MAINTENANCE BERBASIS MOBILE*.

Anggraini, T. E., & Maiyana, E. (2025). PENGGUNAAN ANDROID STUDIO DALAM PENGEMBANGAN APLIKASI PEMESANAN TIKET BIOSKOP. *Jurnal Innovation and Future Technology (IFTECH) P-ISSN*, *7*, 2656–1719.

Aplikasi, P., Android, B., Pembelajaran, U., Simulasi, D., Kemampuan, T., Hanna Balansa, K., Paturusi, S. D. E., & Sengkey, D. F. (n.d.). Development of Android-Based Application For Learning and Simulation of English Language Proficiency Test. *Jurnal Teknik Informatika*, *19*(03), 219–228.

Aprianto, W. (n.d.). *Rancang Bangun Aplikasi Tes Toefl Online Berbasis Web dengan Penilaian Otomatis pada Lembaga Bahasa Asc Indonesia*. *8*, 2025. https://doi.org/10.47002/metik.v9i1.1015

Ardhy, F., & Syahrobi, H. (2021). Rancang Bangun Aplikasi Pembelajaran Aksara Lampung Berbasis Android. *Jurnal Informasi Dan Komputer*, *9*(2).

Arsalna Furqan, 170212165, FTK, PTI, 082282124655. (n.d.).

Asroni, O., Wayan, I., Pratama, P., Putu, I., Sudarsana, E., Peong, H. K., & Innuddin, M. (2024). PENERAPAN USABILITY TESTING DENGAN MENGGUNAKAN METODE RETROSPECTIVE THINK ALOUD UNTUK PENGUKURAN TINGKAT KEBERGUNAAN APLIKASI WISATA LABUAN BAJO. In *Jurnal Mahasiswa Teknik Informatika* (Vol. 8, Number 2).

Dewi, D. S., Wilany, E., Ismadi, S., Saputro, M., & Kepulauan, U. R. (2023). PELATIHAN TOEFL iBT BAGI GURU BAHASA INGGRIS SMK. In *Jurnal Awam* (Vol. 3).

Diantoni, C., Komarudin, O., Rizal Informatika, A., Karawang, S., Ronggo Waluyo, J. H., Timur, T., Karawang, J., & Barat, I. (2024). ARSITEKTUR MVVM DAN FRAMEWORK JETPACK COMPOSE PADA PENGEMBANGAN APLIKASI ANDROID (STUDI KASUS: APLIKASI SUKACOLAB). In *Jurnal Mahasiswa Teknik Informatika* (Vol. 8, Number 3).

Ditha, J. D., Susanto, S., Ardianto, H., Saputri, V. A. M., Putra, D. W., & Kusuma, C. (2024). The Development of 'CATRA' Android-Based Learning Module with Gamification Approach for Students: A Research and Development on Learning Media. *International Journal of Science and Applied Science: Conference Series*, *8*(2), 114. https://doi.org/10.20961/ijsascs.v8i2.95119

Ditha, R. L., Faulina, S. T., & Wisnumurti. (2023). Rancang Bangun Aplikasi Layanan Pengaduan Pada Dinas Pendidikan Kabupaten Oku Berbasis Android Menggunakan Android Studio. *Jurnal Informatika Dan Komputer (JIK)*, *14*(2), 25–35.

Fauzi, R., Nasution, H. N., Hastini, F., Zainy, A., & Lumban Tobing, Y. R. (2022). PEGGUNAAN MEDIA ADOBE FLASH TERHADAP HASIL BELAJAR

SISWA SMKN 1 TANTOM ANGKOLA. *JURNAL EDUCATION AND DEVELOPMENT*, *11*(1), 437–442. https://doi.org/10.37081/ed.v11i1.2687

Indah Puji Astuti, Ega Feri Romawati, & Ida Widaningrum. (2020). Rancang Bangun Aplikasi Mobile Pengenalan Huruf Jawa (Aksara Jawa) Berbasis Android. *Jurnal CoSciTech (Computer Science and Information Technology)*, *1*(2), 93–100. https://doi.org/10.37859/coscitech.v1i2.2185

Mahardika, F., Zulfan, A., & Suseno, A. T. (2023). Implementasi Metode Waterfall pada Sistem Informasi Kepegawaian Berbasis Web. *Blend Sains Jurnal Teknik*, *2*(2), 135–143. https://doi.org/10.56211/blendsains.v2i2.300

Mhd Zulfikar, T., Tanti, L., Utama, P., & Yos Sudarso KM, J. K. (2024). Rancang Bangun Aplikasi Quiz Simulasi TOEFL Memanfaatkan Algoritma Linear Congruential Generator (LCG) Berbasis Android Design and build a TOEFL simulation quiz application utilizing the Android-based Linear Congruential Generator (LCG) algorithm. *Januari*, (2), 18. http://kti.potensi-utama.ac.id/index.php/JID

Mutammimah, F. S., Aeni, A. N., & Nugraha, R. G. (2024). Pengembangan Aplikasi KAGANGA Berbasis Android untuk Mengenalkan Aksara Sunda di Sekolah Dasar. *Jurnal Karya Ilmiah Guru*, *9*(2). https://doi.org/10.51169/ideguru.v9i2.858

Navintino, F. F., Farhan, M., Rilvani, E., & Artikel, R. (2025). *INFO ARTIKEL ABSTRAK*. *4*(1), 22–28. https://doi.org/10.55123

Pemanfaatan Aplikasi Mobile Jaminan Kesehatan Nasional Untuk Meningkatkan Pelayanan BPJS Kesehatan di Klinik Pratama Bertha Kota Medan Putri Utami, A., Asnawi, M., & Firah, A. (n.d.). Universitas Dharmawangsa 30. *Jurnal Bisnis Corporate*, *8*(2), 2579–6445.

Pratama, S. B., Suharto, M. E. F., & Saputro, W. E. (2023). Aplikasi Covid19 Monitoring berbasis Android menggunakan Android Studio dengan Bahasa Pemrograman Kotlin. *Sains Data Jurnal Studi Matematika Dan Teknologi*, *1*(1), 9–20. https://doi.org/10.52620/sainsdata.v1i1.5

Pujianti, W., Nasir, M., & Mawarni, S. (2019). Perancangan Aplikasi Pembelajaran Arab Melayu Berbasis Augmented Reality Untuk Siswa Tingkat Sekolah Dasar. *Seminar Nasional Industri Dan Teknologi (SNIT)*, 37–44.

Rahman, A., Maiyana, E., Sjech Djamil Djambek Bukittinggi Jl Gurun Aua, U. M., Putiah, K., Banuhampu, K., Bukittinggi, K., & Barat, S. (2025). MEMBANGUN APLIKASI CATATAN PERJALANAN SEDERHANA MENGGUNAKAN ANDROID STUDIO. In *Jurnal Innovation and Future Technology (IFTECH) P-ISSN* (Vol. 7, Number 1).

Riza, F., Jamal Al Din, S., Yusuf Al Afghani, D., Setiabudi, R., & Teknologi Budi Utomo, I. (n.d.). LLM-BASED SELF-RELATED LOCAL AI AGENT DESIGN THROUGH N8N ORCHESTRATION FOR CONVERSATIONAL MEMORY ON RAG PERANCANGAN AGEN AI LOKAL MANDIRI BERBASIS LLM MELALUI ORKESTRASI N8N UNTUK MEMORI PERCAKAPAN PADA RAG. *Journal of Information Technology and Computer Science (INTECOMS)*, *8*(3), 2025.

Sakina, R., & Khofifah, S. (2025). Pelatihan Strategi Pengerjaan TOEFL Test (Fokus pada Reading Comprehension). *Jurnal Abdidas*, *6*(1), 121–127. https://doi.org/10.31004/abdidas.v6i1.1117

Shabrina Ziha Fidela, Meisye Putri Azizah, & Septia Rizka Hidayah. (2023). Tren

Pengembangan Aplikasi Mobile: Sebuah Tinjauan Literatur. *Jurnal Teknik Mesin, Industri, Elektro Dan Informatika*, *2*(4), 30–48. https://doi.org/10.55606/jtmei.v2i4.2848

Siregar, U. D. (2023). Bahasa Inggris sebagai Bahasa Komunikasi Bisnis di Era Globalisasi. *JBSI: Jurnal Bahasa Dan Sastra Indonesia*, *3*(01), 129–135. https://doi.org/10.47709/jbsi.v3i01.2608

Sulistiana, E., Adianto, A., & Nadzifah Universitas Hafshawaty Zainul Hasan, W. (n.d.). *Implementasi aplikasi TOEFL Test Pro berbasis android bagi Mahasiswa Universitas Hafshawaty Zainul Hasan*. *8*(1).

Sutrisno, M., Davy Wiranata, A., & Irawan, D. (2023). *LITERATURE REVIEW : ANALISIS METODOLOGI PERANCANGAN APLIKASI TOEFL (TEST OF ENGLISH AS A FOREIGN LANGUAGE) DI INDONESIA* (Vol. 13, Number 2). https://jurnal.umj.ac.id/index.php/just-it/index

Tauhid, K., & Lubis, ; | Patiyasa. (2024). *Pentingnya Menguasai Bahasa Inggris dan Faktor Yang Mempengaruhi Kemampuan Berbahasa Inggris* (Vol. 3).

Widodo, Y. B., Sibuea, S., & Narji, M. (2024). Kecerdasan Buatan dalam Pendidikan: Meningkatkan Pembelajaran Personalisasi. *Jurnal Teknologi Informatika Dan Komputer*, *10*(2), 602–615. https://doi.org/10.37012/jtik.v10i2.2324

Wijaya, L. H., Zulkarnain, I. A., & Nurfitri, K. (2021). Pegon-Gliph Game Pengenalan Dan Pembelajaran Arab Pegon Berbasis Android. *KOMPUTEK*, *5*(1), 77. https://doi.org/10.24269/jkt.v5i1.685

Zulfania Arrahma, F., Mutiara, M., & Alfarisy, F. (2022). KESADARAN MAHASISWA BAHASA ASING AKAN PENTINGNYA BERBICARA BAHASA INGGRIS. *Jurnal Pendidikan Indonesia (Japendi)*, *3*(1).

LAMPIRAN

Lampiran 1. Hasil lembar wawancara

Judul Penelitian/Topik Wawancara: Kepuasan Pengguna Aplikasi Latihan TOEFL
Nama Responden: Hani Amir
Tanggal Wawancara:

No	Aspek Penilaian (Variable)	Indikator / Poin Pertanyaan	STS	TS	CS	S	SS
I	Kelayakan Produk *(Product Feasibility)*	1. Aplikasi berjalan stabil (tidak *crash/force close*) saat dijalankan di HP Android.					✓
		2. Ukuran aplikasi ringan dan tidak membuat perangkat menjadi lambat/panas.					✓
		3. Fitur *Offline* berfungsi dengan baik saat tidak ada koneksi internet.					✓
II	Kemudahan Penggunaan *(Ease of Use)*	4. Tampilan aplikasi (warna, teks, tata letak) enak dipandang dan jelas.				✓	
		5. Navigasi menu (perpindahan dari Beranda ke Latihan/Simulasi) mudah dimengerti.				✓	
		6. Instruksi penggunaan pada fitur "Latihan Pintar" mudah dipahami oleh pengguna baru.				✓	
III	Manfaat Fitur *(Feature Utility)*	7. Soal-soal latihan yang disediakan relevan dengan standar TOEFL.				✓	
		8. Fitur AI (Latihan Pintar) membantu memberikan soal yang sesuai dengan kemampuan saya.				✓	
		9. Adanya fitur "Pembahasan" setelah latihan sangat membantu proses belajar.				✓	

Keterangan Skala Likert:

- 1 = Sangat Tidak Sesuai / Sangat Buruk
- 2 = Tidak Sesuai / Buruk
- 3 = Cukup Sesuai / Cukup
- 4 = Sesuai / Baik
- 5 = Sangat Sesuai / Sangat Baik

Judul Penelitian/Topik Wawancara: Kepuasan Pengguna Aplikasi Latihan TOEFL
Nama Responden: Yudhi Saputra
Tanggal Wawancara:

No	Aspek Penilaian (Variable)	Indikator / Poin Pertanyaan	STS	TS	CS	S	SS
I	Kelayakan Produk *(Product Feasibility)*	1. Aplikasi berjalan stabil (tidak *crash/force close*) saat dijalankan di HP Android.				✓	
		2. Ukuran aplikasi ringan dan tidak membuat perangkat menjadi lambat/panas.				✓	
		3. Fitur *Offline* berfungsi dengan baik saat tidak ada koneksi internet.				✓	
II	Kemudahan Penggunaan *(Ease of Use)*	4. Tampilan aplikasi (warna, teks, tata letak) enak dipandang dan jelas.					✓
		5. Navigasi menu (perpindahan dari Beranda ke Latihan/Simulasi) mudah dimengerti.					✓
		6. Instruksi penggunaan pada fitur "Latihan Pintar" mudah dipahami oleh pengguna baru.					✓
III	Manfaat Fitur *(Feature Utility)*	7. Soal-soal latihan yang disediakan relevan dengan standar TOEFL.					✓
		8. Fitur AI (Latihan Pintar) membantu memberikan soal yang sesuai dengan kemampuan saya.					✓
		9. Adanya fitur "Pembahasan" setelah latihan sangat membantu proses belajar.					✓

Keterangan Skala Likert:

- 1 = Sangat Tidak Sesuai / Sangat Buruk
- 2 = Tidak Sesuai / Buruk
- 3 = Cukup Sesuai / Cukup
- 4 = Sesuai / Baik
- 5 = Sangat Sesuai / Sangat Baik

Judul Penelitian/Topik Wawancara: Kepuasan Pengguna Aplikasi Latihan TOEFL
Nama Responden: Putri Salsha Aprilia
Tanggal Wawancara:

No	Aspek Penilaian (Variable)	Indikator / Poin Pertanyaan	STS	TS	CS	S	SS
I	Kelayakan Produk *(Product Feasibility)*	1. Aplikasi berjalan stabil (tidak *crash/force close*) saat dijalankan di HP Android.				✓	
		2. Ukuran aplikasi ringan dan tidak membuat perangkat menjadi lambat/panas.				✓	
		3. Fitur *Offline* berfungsi dengan baik saat tidak ada koneksi internet.				✓	
II	Kemudahan Penggunaan *(Ease of Use)*	4. Tampilan aplikasi (warna, teks, tata letak) enak dipandang dan jelas.				✓	
		5. Navigasi menu (perpindahan dari Beranda ke Latihan/Simulasi) mudah dimengerti.				✓	
		6. Instruksi penggunaan pada fitur "Latihan Pintar" mudah dipahami oleh pengguna baru.				✓	
III	Manfaat Fitur *(Feature Utility)*	7. Soal-soal latihan yang disediakan relevan dengan standar TOEFL.				✓	
		8. Fitur AI (Latihan Pintar) membantu memberikan soal yang sesuai dengan kemampuan saya.				✓	
		9. Adanya fitur "Pembahasan" setelah latihan sangat membantu proses belajar.				✓	

Keterangan Skala Likert:

- 1 = Sangat Tidak Sesuai / Sangat Buruk
- 2 = Tidak Sesuai / Buruk
- 3 = Cukup Sesuai / Cukup
- 4 = Sesuai / Baik
- 5 = Sangat Sesuai / Sangat Baik

Judul Penelitian/Topik Wawancara: Kepuasan Pengguna Aplikasi Latihan TOEFL

Nama Responden: Dani Firgiawan

Tanggal Wawancara:

No	Aspek Penilaian (Variable)	Indikator / Poin Pertanyaan	STS	TS	CS	S	SS
I	**Kelayakan Produk** *(Product Feasibility)*	1. Aplikasi berjalan stabil (tidak *crash/force close*) saat dijalankan di HP Android.					✓
		2. Ukuran aplikasi ringan dan tidak membuat perangkat menjadi lambat/panas.					✓
		3. Fitur *Offline* berfungsi dengan baik saat tidak ada koneksi internet.					✓
II	**Kemudahan Penggunaan** *(Ease of Use)*	4. Tampilan aplikasi (warna, teks, tata letak) enak dipandang dan jelas.					✓
		5. Navigasi menu (perpindahan dari Beranda ke Latihan/Simulasi) mudah dimengerti.					✓
		6. Instruksi penggunaan pada fitur "Latihan Pintar" mudah dipahami oleh pengguna baru.					✓
III	**Manfaat Fitur** *(Feature Utility)*	7. Soal-soal latihan yang disediakan relevan dengan standar TOEFL.					✓
		8. Fitur AI (Latihan Pintar) membantu memberikan soal yang sesuai dengan kemampuan saya.					✓
		9. Adanya fitur "Pembahasan" setelah latihan sangat membantu proses belajar.					✓

Keterangan Skala Likert:

- 1 = Sangat Tidak Sesuai / Sangat Buruk
- 2 = Tidak Sesuai / Buruk
- 3 = Cukup Sesuai / Cukup
- 4 = Sesuai / Baik
- 5 = Sangat Sesuai / Sangat Baik

Judul Penelitian/Topik Wawancara: Kepuasan Pengguna Aplikasi Latihan TOEFL

Nama Responden: Anggi Mauana

Tanggal Wawancara:

No	Aspek Penilaian (Variable)	Indikator / Poin Pertanyaan	STS	TS	CS	S	SS
I	**Kelayakan Produk** *(Product Feasibility)*	1. Aplikasi berjalan stabil (tidak *crash/force close*) saat dijalankan di HP Android.				✓	
		2. Ukuran aplikasi ringan dan tidak membuat perangkat menjadi lambat/panas.				✓	
		3. Fitur *Offline* berfungsi dengan baik saat tidak ada koneksi internet.				✓	
II	**Kemudahan Penggunaan** *(Ease of Use)*	4. Tampilan aplikasi (warna, teks, tata letak) enak dipandang dan jelas.				✓	
		5. Navigasi menu (perpindahan dari Beranda ke Latihan/Simulasi) mudah dimengerti.				✓	
		6. Instruksi penggunaan pada fitur "Latihan Pintar" mudah dipahami oleh pengguna baru.				✓	
III	**Manfaat Fitur** *(Feature Utility)*	7. Soal-soal latihan yang disediakan relevan dengan standar TOEFL.					✓
		8. Fitur AI (Latihan Pintar) membantu memberikan soal yang sesuai dengan kemampuan saya.					✓
		9. Adanya fitur "Pembahasan" setelah latihan sangat membantu proses belajar.					✓

Keterangan Skala Likert:

- 1 = Sangat Tidak Sesuai / Sangat Buruk
- 2 = Tidak Sesuai / Buruk
- 3 = Cukup Sesuai / Cukup
- 4 = Sesuai / Baik
- 5 = Sangat Sesuai / Sangat Baik

Lampiran 2. *Room Database*

```
package com.pandu.latihantoefl.data.db

// 1. Tambahkan semua entity di sini
@Database(
    entities = [
        // --- Data Master ---
        Materi::class,
        User::class,

        // --- Simulasi Core ---
        Simulasi::class,
        AiSmart::class,

        // --- Bank Soal Components (Resources) ---
        Reading::class,
        Structure::class,
        Listening::class,

        // --- Struktur Soal ---
        Soal::class,
        OpsiJawaban::class,

        // --- Perekaman Hasil ---
        Sesi::class,
        PilihanJawaban::class,

        // ---- AI SECTION ------
        AiOpsi::class,
        AiSimulasi::class,
        AiSoal::class,
        AiJawaban::class
    ],
    version = 1,
    exportSchema = false)
abstract class AppDatabase : RoomDatabase() {

    abstract fun materiDao(): MateriDao
    abstract fun simulasiDao(): SimulasiDao
    abstract fun sesiDao(): SesiDao
    abstract fun userDao(): UserDao
    abstract fun soalDao(): SoalDao
    abstract fun readingDao(): ReadingDao
    abstract fun listeningDao(): ListeningDao
    abstract fun structureDao(): StructureDao
    abstract fun opsiJawabanDao(): OpsiJawabanDao
    abstract fun pilihanJawabanDao(): PilihanJawabanDao
    abstract fun aiDao(): AiDao

    // Callback untuk Injeksi Data
    private class AppDatabaseCallback(
        private val scope: CoroutineScope
    ) : Callback() {

        override fun onCreate(db: SupportSQLiteDatabase) {
            super.onCreate(db)
```

```
                // Proses Injeksi dimulai saat DB dibuat pertama kali
                INSTANCE?.let { database ->
                    scope.launch {
                        // A. Injeksi Materi
                        val materiDao = database.materiDao()
                        materiDao.insertAll(getDummyMateri())
                        Log.d("RoomDB", "Materi berhasil diinjeksi")

                        // B. Injeksi Simulasi
                        val simulasiDao = database.simulasiDao()

simulasiDao.insertAllSimulasi(getDummySimulasi())
                        Log.d("RoomDB", "Simulasi berhasil diinjeksi")

                        // B. Injeksi Simulasi(Todo ganti dengan data
asli hasil login kalau sudah siap)
                        val userDao = database.userDao()
                        userDao.insertUserData(getDummyUser())
                        Log.d("RoomDB", "User berhasil diinjeksi")

//                        // C. Injeksi Sesi (Todo dihapus saat sudah
ada data asli)
//                        val sesiDao = database.sesiDao()
//                        sesiDao.insertAllSesi(getDummySesi())
//                        Log.d("RoomDB", "Sesi berhasil diinjeksi")

                        // 1. Injeksi Resources dulu (Parent tables)
                        val readingDao = database.readingDao() //
Pastikan buat ReadingDao
                        readingDao.insertAllReading(getReading105())
                        Log.d("RoomDB", "Reading berhasil diinjeksi")

                        val listeningDao = database.listeningDao() //
Pastikan buat ListeningDao

listeningDao.insertAllListening(getListening105())
                        Log.d("RoomDB", "Listening berhasil
diinjeksi")

                        val structureDao = database.structureDao() //
Pastikan buat StructureDao
                        structureDao.insertAll(getStructure105())
                        Log.d("RoomDB", "Structure berhasil
diinjeksi")

                        // 2. Injeksi Soal (Child table)
                        val soalDao = database.soalDao()
                        soalDao.insertAllSoal(getSoalSimulasi105())
                        Log.d("RoomDB", "Soal berhasil diinjeksi")

                        // 3. Injeksi Opsi Jawaban
                        val opsiDao = database.opsiJawabanDao()
                        opsiDao.insertAll(getOpsiJawabanSimulasi105())
                        Log.d("RoomDB", "OpsiJawaban berhasil
diinjeksi")
                    }
                }
            }
        }
```

```
    companion object {
        @Volatile
        private var INSTANCE: AppDatabase? = null

        fun getDatabase(context: Context, scope: CoroutineScope):
AppDatabase {
            return INSTANCE ?: synchronized(this) {
                val instance = Room.databaseBuilder(
                    context.applicationContext,
                    AppDatabase::class.java,
                    "toefl_app_database"
                )
                    .addCallback(AppDatabaseCallback(scope)) //
Pasang Callback
                    .fallbackToDestructiveMigration()
                    .build()
                INSTANCE = instance
                instance
            }
        }
    }
}
```

Lampiran 3. ***Api service***

```
package com.pandu.latihantoefl.networking

import
com.pandu.latihantoefl.networking.request.GenerateSoalRequest
import
com.pandu.latihantoefl.networking.response.GenerateSoalResponse
import retrofit2.http.Body
import retrofit2.http.POST


interface ApiService {
    @POST("generate-soal")
    suspend fun sendResultToServer(
        @Body request: GenerateSoalRequest
    ): GenerateSoalResponse
}
```

Lampiran 4. ***Main Activity***

```
package com.pandu.latihantoefl

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
```

```
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.padding
import androidx.compose.material3.Scaffold
import androidx.compose.runtime.getValue
import androidx.compose.ui.Modifier
import androidx.navigation.NavGraph.Companion.findStartDestination
import androidx.navigation.compose.currentBackStackEntryAsState
import androidx.navigation.compose.rememberNavController
import com.pandu.latihantoefl.navigation.AppNavHost
import com.pandu.latihantoefl.navigation.AppScreen
import com.pandu.latihantoefl.navigation.getMainBottomItem
import com.pandu.latihantoefl.ui.components.AppBottomBar
import com.pandu.latihantoefl.ui.theme.LatihanTOEFLTheme
import dagger.hilt.android.AndroidEntryPoint

@AndroidEntryPoint
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            LatihanTOEFLTheme {
                val navController = rememberNavController()
                val navBackStackEntry by
navController.currentBackStackEntryAsState()
                val currentRoute =
navBackStackEntry?.destination?.route

                val bottomBarRoute = getMainBottomItem().map {
it.route }
                val showBottomBar = currentRoute in bottomBarRoute

                Scaffold(
                    modifier = Modifier.fillMaxSize(),
                    bottomBar = {
                        if (showBottomBar) {
                            AppBottomBar(
                                items = getMainBottomItem(),
                                currentRoute = currentRoute ?:
AppScreen.HOME.route, // Gunakan variable dynamic
                                onTabSelected = { newRoute ->
                                    // 4. Logika Navigasi Standar
Bottom Bar

navController.navigate(newRoute) {
                                        // Pop up ke start
destination agar tidak menumpuk stack

popUpTo(navController.graph.findStartDestination().id) {
                                            saveState = true
                                        }
                                        // Hindari duplikasi layar
jika diklik berkali-kali
                                        launchSingleTop = true
                                        // Restore state (scroll
position, dll)
                                        restoreState = true
                                    }
                                }
                            )
```

```
                        }
                    }
                ) { innerPadding ->
                    AppNavHost(
                        navController = navController,
                        modifier = Modifier.padding(innerPadding)
                    )
                }
            }
        }
    }
}
```

Lampiran 5. Manajemen Data AI (Repository)

```
package com.pandu.latihantoefl.data.repository

import androidx.room.withTransaction
import com.pandu.latihantoefl.data.db.AppDatabase
import com.pandu.latihantoefl.data.db.ai.AiDao
import com.pandu.latihantoefl.data.db.ai.AiOpsi
import com.pandu.latihantoefl.data.db.ai.AiSimulasi
import com.pandu.latihantoefl.data.db.ai.AiSoal
import com.pandu.latihantoefl.data.db.dao.SesiDao
import com.pandu.latihantoefl.data.helper.Resource
import com.pandu.latihantoefl.networking.ApiService
import
com.pandu.latihantoefl.networking.request.GenerateSoalRequest
import
com.pandu.latihantoefl.networking.response.GenerateSoalResponse
import kotlinx.coroutines.flow.Flow
import kotlinx.coroutines.flow.flow
import javax.inject.Inject

class LatihanPintarRepository @Inject constructor(
    private val db: AppDatabase,
    private val apiService: ApiService,
    private val aiDao: AiDao,
    private val sesiDao: SesiDao
) {

    suspend fun createAiSimulation(
        userId: Int,
        type: String,
        difficulty: String,
        amount: Int
    ): Flow<Resource<Int>> = flow {
        emit(Resource.Loading)

        try {
            // ----------------------------------------------------
------
            // 1. VALIDASI SESI OFFLINE (Cek Tabel 'Sesi')
            // ----------------------------------------------------
------
```

```
            val activeOffline =
sesiDao.getLatestActiveSession(userId)
            if (activeOffline != null) {
                // Sinyal ke ViewModel bahwa ada sesi Offline
nyangkut
                emit(Resource.Error("CONFLICT_OFFLINE"))
                return@flow
            }

            // ----------------------------------------------------
------
            // 2. VALIDASI SESI AI (Cek Tabel 'AiSimulasi')
            // ----------------------------------------------------
------
            val activeAi = aiDao.getActiveAiSession()
            if (activeAi != null) {
                // Sinyal ke ViewModel bahwa ada sesi AI nyangkut
                // Kita gunakan kode singkat agar mudah di-parse
di 'when' statement ViewModel
                emit(Resource.Error("CONFLICT_AI"))
                return@flow
            }

            // ----------------------------------------------------
------
            // 3. REQUEST KE SERVER (Jika Aman/Tidak Ada Konflik)
            // ----------------------------------------------------
------
            val request = GenerateSoalRequest(type, difficulty,
amount)
            val response = apiService.sendResultToServer(request)

            if (response.success) {
                // Simpan ke DB AI & Return ID Baru
                val newId = saveToAiDatabase(response, type,
difficulty, amount)
                emit(Resource.Success(newId))
            } else {
                emit(Resource.Error(response.message))
            }

        } catch (e: Exception) {
            e.printStackTrace()
            emit(Resource.Error(e.localizedMessage ?: "Gagal
terhubung ke server AI."))
        }
    }

    /**
     * Menyimpan Data Response AI ke Struktur Tabel Baru
(AiSimulasi, AiSoal, AiOpsi)
     */
    private suspend fun saveToAiDatabase(
        response: GenerateSoalResponse,
        tipeLatihan: String,
        difficulty: String,
        jumlahSoal: Int
    ): Int {
        return db.withTransaction {
```

```
                // A. Insert Header
                val header = AiSimulasi(
                    nama = "Latihan $tipeLatihan ($difficulty)",
                    tipePaket = tipeLatihan,
                    totalSoal = jumlahSoal,
                    skor = null // Null = Sesi Aktif
                )
                // Tangkap ID Auto-Increment
                val aiSimulasiId =
aiDao.insertAiSimulasi(header).toInt()

                // B. Loop Soal
                response.data.forEach { item ->
                    val soal = AiSoal(
                        aiSimulasiId = aiSimulasiId,
                        tipe = item.type,
                        pertanyaan = item.questionText,
                        penjelasan = item.explanation,
                        // Mapping field opsional sesuai tipe soal
                        passageText = if (item.type.equals("reading",
true)) item.passageText else null,
                        topicStructure = if
(item.type.equals("structure", true)) item.topic else null
                    )

                    val aiSoalId = aiDao.insertAiSoal(soal).toInt()

                    // C. Loop Opsi
                    val opsiList = item.options.map { opt ->
                        AiOpsi(
                            aiSoalId = aiSoalId,
                            teksOpsi = opt.text,
                            isCorrect = opt.isCorrect
                        )
                    }
                    aiDao.insertAiOpsi(opsiList)
                }

                // Return ID Header untuk Navigasi
                aiSimulasiId
            }
        }
    }
```