Pengembangan Aplikasi Simulasi Sesi *Speaking* Tes *IELTS* Berbasis *Android*

1st Indratama Pangasian Manalu
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
ianindratama@student.telkomuniversity.ac.id

2nd Casi Setianingsih
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
setiacasie@telkomuniversity.ac.id

3rd Astri Novianty
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
astrinov@telkomuniversity.ac.id

***Abstrak*—*International English Language Testing System* atau biasa dikenal dengan *IELTS* adalah salah satu tes kemampuan bahasa Inggris. Dalam tes *IELTS* terdiri dari 4 sesi yaitu *Reading*, *Writing*, *Listening*, dan *Speaking*. Menurut statistik pada tahun 2022, sesi *Speaking* menempati posisi kedua dengan nilai terendah. Hal ini dikarenakan persiapan yang dilakukan untuk sesi *Speaking* ini cukup kompleks dan memakan waktu yang cukup lama.**

Untuk mengatasi permasalahan tersebut, dikembangkan sebuah aplikasi Android IELTS Speaking Simulation yang dapat menjadi media pelatihan tes IELTS sesi *speaking* yang dapat memberikan penilaian secara langsung dengan menggunakan teknologi Artificial Intelligence. Dalam memberikan penilaian simulasi tes IELTS sesi *speaking*, aplikasi IELTS Speaking Simulation menggunakan matriks penilaian tes IELTS sesi speaking yaitu *Fluency, Lexical, Grammar,* dan *Pronunciation.*

Aplikasi IELTS Speaking Simulation telah diuji terhadap 33 responden yang merupakan calon pengguna aplikasi yang terdiri dari pelajar dan ahli bahasa Inggris. Dari hasil pengujian didapat hasil pengujian alfa aplikasi Android 100% dan hasil pengujian beta aplikasi mendapatkan hasil yang memuaskan dimana 96% dari 33 responden setuju bahwa *User Interface / User Experience* dari aplikasi telah sesuai dengan apa yang diharapkan dibutuhkan pada aplikasi simulasi sesi *speaking* tes IELTS.

Kata kunci*— *IELTS*, *Speaking*, *Fluency*, *Lexical*, *Grammar*, *Pronunciation*, *Android

I. PENDAHULUAN

International English Language Testing System (*IELTS*) adalah sebuah tes kemampuan bahasa Inggris yang menjadi salah satu prasyarat wajib untuk seseorang yang ingin bekerja, belajar, atau bermigrasi ke negara yang menggunakan bahasa Inggris sebagai bahasa utama *[1]*. Tes ini terdiri dari 4 sesi yaitu *Reading*, *Writing*, *Listening*, dan *Speaking [2]*. Menurut statistik resmi yang dikeluarkan oleh pihak penyelenggara tes di tahun 2022, sesi *Speaking* menempati posisi kedua dengan nilai terendah *[3]*. Hal ini dikarenakan memerlukan proses pelatihan yang melelahkan dan kompleks karena perlunya untuk merekam percakapan diri sendiri ataupun mencari teman bicara untuk mendapatkan *feedback* dari pengucapan yang telah diucap. Terdapat beberapa matriks evaluasi pada sesi *Speaking* tes *IELTS* yaitu *Fluency, Lexical, Grammar,* dan *Pronunciation* dan di setiap matriks evaluasi memiliki kriteria masing masing di setiap *band [4].*

Untuk mempersiapkan sebelum mengikuti tes *IELTS*, terdapat simulasi yang diadakan oleh pihak resmi penyelenggara tes *IELTS*. Simulasi ini mencakup format tes dengan berbagai contoh pertanyaan dan jawaban tes [5]. Biaya untuk melakukan simulasi di website resmi *IELTS* berkisar sekitar Rp700.000 [6] dan hasil penilaian berupa *feedback* report akan diberikan lima hari setelah pengambilan simulasi tes [7]. Dari kedua hal tersebut, banyak peserta yang mengurungkan niatnya untuk mengambil simulasi tes dikarenakan biaya yang cukup mahal dan proses menunggu hasil penilaian cukup lama.

Dari permasalahan yang didapat, maka akan dikembangkan sebuah sistem yang dapat melakukan penilaian simulasi sesi *speaking* tes *IELTS* secara langsung. Untuk mengembangkan sistem ini akan digunakan 3 teknologi yaitu Android, Machine Learning (ML) & Artificial Intelligence (AI), dan Cloud. Agar sistem ini dapat digunakan oleh pengguna dengan mudah, sistem akan mengintegrasikan ketiga teknologi tersebut dimana Android berperan sebagai *platform* untuk pengguna aplikasi melakukan simulasi, ML & AI untuk mengevaluasi setiap jawaban simulasi pengguna dan sistem Cloud untuk mengakses model ML dan menerima berkas audio jawaban pengguna untuk dinilai oleh model ML yang sudah di *deploy* di Cloud.

Pada jurnal ini akan dibahas bagaimana pengembangan sistem dari sudut pandang pembuatan aplikasi Android yang nantinya dapat digunakan oleh pengguna untuk melakukan simulasi sesi *speaking* tes IELTS.

II. KAJIAN TEORI

A. Pengembangan Aplikasi Android

Pada Android Studio, penulisan XML berfungsi sebagai salah satu cara untuk melakukan penempatan tata letak yang nantinya akan digunakan sebagai *User Interface* untuk pengguna aplikasi Android.

Kotlin adalah bahasa pemrograman yang populer digunakan untuk membangun aplikasi Android. Di 2019, Google memprioritaskan Kotlin sebagai bahasa pemrograman utama untuk Android dibanding bahasa pemrograman C++ atau Java [8].

B. Android Room Database

Android Room Database adalah sebuah library yang disediakan oleh Android untuk mempermudah pengelolaan operasi data dalam sebuah *database* lokal di aplikasi

Android. *Library* Room menyediakan *layer* abstraksi SQLite yang memungkinkan akses database yang mudah sambil memanfaatkan kemampuan penuh dari SQLite.

C. Android MediaRecorder

MediaRecorder adalah komponen yang disediakan oleh Android agar aplikasi dapat merekam suara / vidio [9]. Dengan API ini, pengembang melalui aplikasi dapat merekam inputan suara / vidio pengguna. Komponen ini akan langsung merekam kemudian menyimpan inputan suara / vidio tersebut ke sebuah berkas. Berkas ini nantinya dapat ditelusuri secara manual direktorinya di dalam gawai yang telah memasang aplikasi tersebut ataupun diputar didalam aplikasi menggunakan komponen MediaPlayer.

D. Courotines

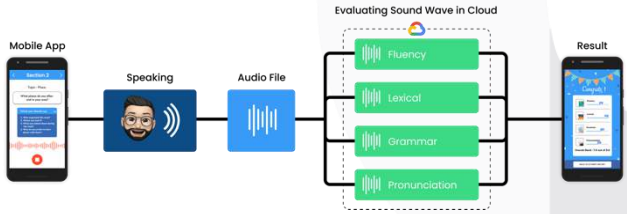
Sebuah Courotines adalah sebuah pola desain *concurrency* yang dapat digunakan pada Android untuk menyederhakan blok kode yang dieksekusi secara asinkron. Pada Android, Courotines dapat membantu pengembang untuk mengatur pekerjaan – pekerjaan yang bersifat *long-running* yang mungkin dapat menghentikan Thread utama dan mengakibatkan aplikasi menjadi tidak responsif untuk pengguna [10].

E. Firebase

Firebase adalah suatu layanan dari Google untuk memberikan kemudahan bahkan mempermudah para pengembang aplikasi dalam mengembangkan aplikasinya [11]. Firebase memiliki beberapa fitur seperti Analytics, Cloud Messaging and Notifications, Authentication, Cloud Firestore, Realtime Database, dan Hosting.

III. METODE

A. Desain Sistem



GAMBAR 1.
Desain Sistem

Aplikasi Android simulasi sesi *speaking* tes IELTS dikembangkan menggunakan 3 teknologi yaitu Android, AI, dan Cloud. Pada aplikasi Android, simulasi akan dikembangkan sesuai dengan alur tes IELTS sesi *speaking* yang terdiri dari 3 *Sections*. Penilaian simulasi akan diproses dari setiap pertanyaan yang dijawab pengguna. Setelah aplikasi menampilkan dan membacakan pertanyaan ke pengguna aplikasi, maka untuk setiap jawaban pengguna, aplikasi akan membuat sebuah berkas audio yang berisi jawaban pengguna dan lalu kemudian mengimkannya ke Cloud sebagai *request* ke REST API proyek ini untuk dievaluasi oleh model – model ML yang terdapat di Cloud.

Setelah hasil evaluasi selesai, maka Cloud akan mengirimkan hasil evaluasi ke aplikasi Android dalam bentuk *response data*. Penting untuk diingat, *response data* yang dikirimkan kembali adalah 4 nilai metriks penilaian sesi *speaking* tes IELTS (*Fluency, Lexical, Grammar,* dan *Pronunciation)* dan sebuah data teks transkrip dari jawaban audio pengguna.

Selanjutnya, aplikasi akan menyimpan *response data* tersebut ke Firebase Realtime Database (FRTDB). Data transkrip akan disimpan sebagai data simulasi beserta dengan id pertanyaan sedangkan 4 data nilai hasil evaluasi akan diakumulasi dengan 4 data nilai hasil evaluasi lainnya lalu disimpan sebagai data nilai. Terakhir, ketika pengguna melihat nilai simulasi, maka aplikasi akan mendapatkan data nilai tersebut dari FRTDB dan kemudian menghitung nilai keseluruhannya dengan cara membaginya dengan jumlah pertanyaan yang dijawab oleh pengguna lalu nilai rerata tersebut dibulatkan sesuai dengan aturan pembulatan penilaian IELTS. Terakhir, setelah nilai keseluruhan diproses, maka aplikasi akan menampilkannya ke pengguna seperti yang dapat dilihat pada Gambar 1.

B. Analisis Kebutuhan Sistem

Berdasarkan penjabaran desain sistem, maka aplikasi Android simulasi sesi speaking tes IELTS diperlukan untuk mengimplementasikan beberapa bagian berikut:

1. *Database Topik dan Pertanyaan*

Database topik dan pertanyaan perlu untuk diimplementasikan dengan tujuan untuk mendapatkan pertanyaan – pertanyaan yang sesuai dan relevan dengan tes IELTS bagian *speaking* dan kemudian menampilkannya kepada pengguna aplikasi sebagai bagian dari simulasi sesi speaking tes IELTS.

2. Sistem Pengaturan Waktu Menjawab Pertanyaan Simulasi

Sistem pengaturan waktu menjawab pertanyaan simulasi perlu untuk diimplementasikan karena akan pakai saat pengguna melakukan simulasi. Saat simulasi berlangsung, di setiap section pada simulasi, pengguna akan mendapatkan pertanyaan yang ditampilkan dan dibacakan oleh aplikasi. Setelah pertanyaan selesai dibacakan, sistem akan memulai merekam jawaban pertanyaan dari pengguna berupa audio dan akan memulai sistem waktu *section* tersebut. Hal ini dirancang agar menyerupai tes resmi IELTS sesi speaking. Ketika pengguna sudah mencapai batas maksimum dari *section* tersebut, maka sistem akan memberitahu komponen perekaman untuk memberhentikan rekaman suara pengguna. Sesuai denga tes resmi IELTS sesi *speaking*, setiap bagian memiliki pengaturan waktunya masing-masing untuk menjawab pertanyaan, dimana *Section* 1 dan 3 memiliki batas waktu maksimal menjawab adalah 5 menit, sedangkan untuk *Section* 2 adalah 2 menit.

3. Sistem Asesmen Keseluruhan dari Simulasi

Sistem Asesmen keseluruhan dari Simulasi perlu diimplementasikan untuk menghitung nilai akhir dari simulasi yang dinilai berdasarkan hasil penilaian ke-4 matriks yang sudah ditentukan. Penting untuk diingat, nilai akhir adalah nilai akumulasi yang didapatkan dari setiap data response yang diterima secara asinkron oleh Android lalu diteruskan ke Firebase Realtime Database ketika Android mengirim *request* ke API dengan menyertakan berkas audio jawaban pengguna. Penting untuk diingat, aplikasi akan melakukan perhitungan untuk membagi nilai setiap matriks

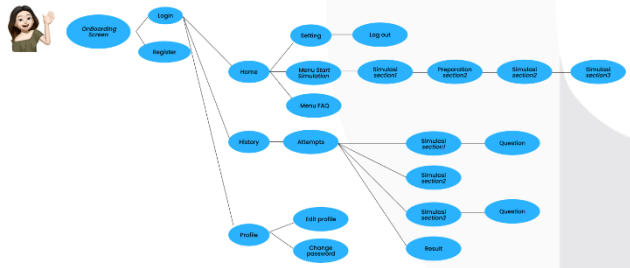
penilaian dengan jumlah pertanyaan yang ada pada simulasi dan selanjutnya melakukan pembulatan nilai.

4. Sistem Manajemen Pengguna

Dalam pembuatan aplikasi ini memerlukan sebuah sistem manajemen pengguna sebagai salah satu cara untuk membantu pengguna dalam mengakses aplikasi berdasarkan akun yang mereka miliki. Dengan bantuan Firebase [12], sistem manajemen pengguna akan dibagi menjadi dua bagian, yaitu bagian penyimpanan data akun pengguna dan bagian penyimpanan data simulasi yang pengguna telah laksanakan. Pada aplikasi kali ini, produk Authentication digunakan sebagai backend servis untuk melakukan proses autentikasi akun pada aplikasi. Produk Realtime Database digunakan untuk menyimpan data akun pengguna beserta dengan data simulasi yang pengguna laksanakan. Produk Storage digunakan untuk menyimpan berkas gambar dari foto profil akun pengguna. Terakhir, produk Dynamic Links digunakan untuk verifikasi pendaftaran akun pengguna.

5. Pengembangan *User Interface / User Experience* Aplikasi

Implementasi *User Interface* (UI) / *User Experience* (UX) pada aplikasi Android akan dibagi menjadi 4 bagian utama, yaitu bagian Login & Register akun (autentikasi), Home, History, dan Profile. Bagian Login & Register diperuntukkan untuk kegiatan pengguna dalam mendaftar akun dan kemudian masuk ke aplikasi menggunakan akun yang telah didaftarkan. Bagian Home diperuntukkan untuk kegiatan simulasi dan *logout* akun dari aplikasi. Bagian History digunakan untuk menampilkan list *attempt* simulasi yang telah dilakukan pengguna. Terakhir, bagian Profile digunakan untuk menampilkan data diri pengguna dan tersedia juga tombol untuk mengganti password dan sunting profil agar pengguna bisa memodifikasi data diri pengguna. Lebih lanjut, secara struktur, *use case diagram* dari aplikasi Android dapat dilihat di Gambar 2.



GAMBAR 2.
Use Case Diagram Aplikasi

6. Sistem Penghubung Antara Android dan API

Sistem penghubung antara Android dan API dibuat untuk digunakan ketika pengguna telah selesai menjawab pertanyaan simulasi dan kemudian menghentikan perekaman. Pada tahap ini, berkas audio dikirim ke API secara asinkron agar pengguna bisa melanjutkan ke pertanyaan berikutnya tanpa harus menunggu proses pengiriman berkas audio ke API dan penyimpanan data ke Firebase selesai.

C. Proses Pengembangan Sistem

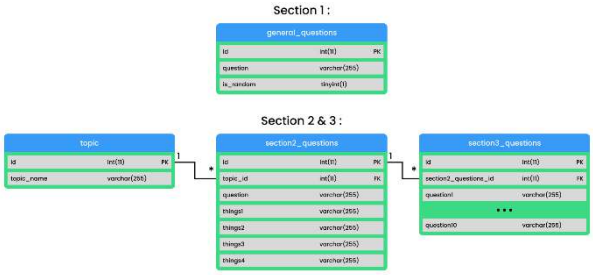
Seperti yang dijelaskan pada bagian B. Analisis Kebutuhan Sistem, terdapat 6 sub-sistem yang perlu dikembangkan untuk aplikasi Android. Berikut adalah proses pengembangan sistem untuk masing – masing sub-sistem:

1. *Database* Topik dan Pertanyaan

Pembuatan *database* topik dan pertanyaan dibuat menggunakan Android Room Database. *Database* disimpan secara lokal didalam perangkat selular pengguna. Di setiap sesi simulasi, pertanyaan yang diberikan ke pengguna akan diberikan secara acak.

Pembuatan *database* topik dan pertanyaan diawali dengan membuat struktur dari *database*, lalu membuat perintah untuk mengisi *database* secara otomatis saat pembuatan *database* (*prepopulate*) dengan *dataset* yang telah dibuat, dan kemudian terakhir, adalah membuat perintah untuk mengakses setiap data pada tabel saat pengguna melakukan simulasi.

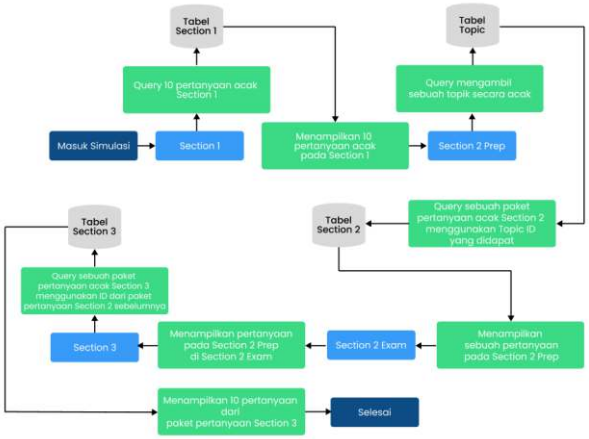
Struktur dari *database* dibuat menyesuaikan dengan tes *IELTS* sesi *speaking*, dimana dalam satu sesi speaking akan dibagi menjadi 3 bagian yaitu *section* 1 akan berisi beberapa pertanyaan umum, *section* 2 yang berisi sebuah topik, pertanyaan terkait topik tersebut, dan empat hal yang peserta harus sampaikan dalam monolog di *section* 2 tersebut, dan *section* 3 berisi beberapa pertanyaan lanjutan dari pertanyaan di section 2. Penting untuk diingat, dengan menggunakan *Android Room Database*, sebuah tabel yang dibentuk pada *database* adalah sebuah entitas (*instance*).



GAMBAR 3.
Relasi tabel pada Database

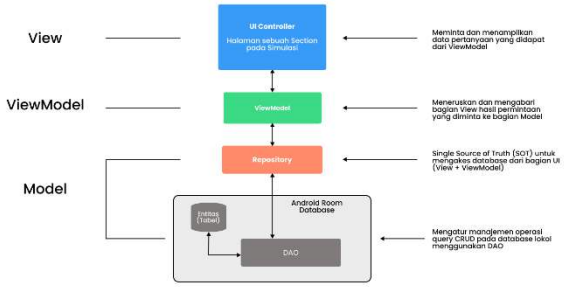
Setelah selesai menetapkan struktur dari *database*, langkah selanjutnya adalah membuat perintah untuk mengisi *database* secara otomatis saat pembuatan *database* (*prepopulate*). Dengan membuat *dataset* menjadi bentuk JSON, maka aplikasi dapat membaca setiap *object* JSON sebagai sebuah data masukan (*instance*) dan kemudian memasukkannya ke tabel dari *database*.

Langkah terakhir dari pengerjaan *database* adalah membuat perintah untuk mengakses setiap data pada tabel saat pengguna melakukan simulasi. Secara alur kerja, proses yang dilakukan untuk menampilkan pertanyaan berbeda di setiap *section*. Hal ini dilakukan agar sesuai dengan proses yang terjadi selama tes resmi *IELTS* sesi *speaking*.



GAMBAR 4.
Flowchart penggunaan database pertanyaan saat simulasi berlangsung

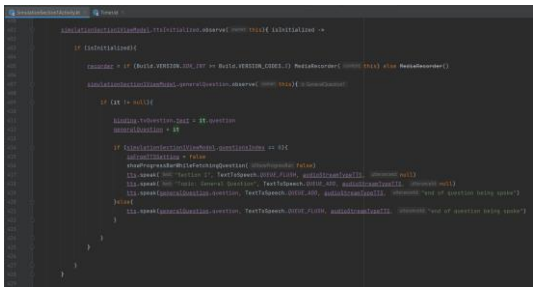
Berikut adalah struktur Model-View-ViewModel (MVVM) dari aplikasi terkait sub-sistem ini.



GAMBAR 5.
Struktur MVVM dari aplikasi terkait database topik dan pertanyaan

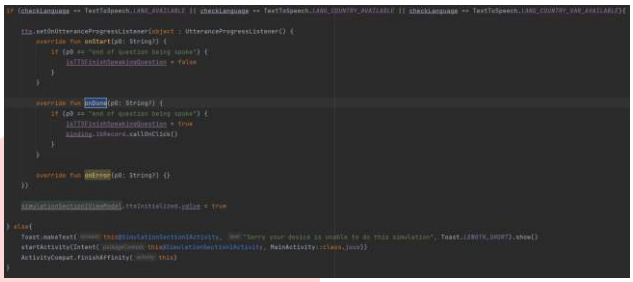
2. Sistem Pengaturan Waktu Menjawab Pertanyaan Simulasi

Alur dari implementasi sistem pengaturan waktu menjawab pertanyaan simulasi diawali ketika aplikasi membacakan pertanyaan. Penting untuk diingat, komponen *Timer* baru akan mulai mencatat durasi waktu jawab pengguna ketika pertanyaan selesai dibacakan oleh *TextToSpeech*. Agar aplikasi dapat memberi tahu komponen *Timer* untuk memulai pencatatan waktu ketika pertanyaan selesai dibacakan oleh *TextToSpeech*, maka aplikasi, melalui metode "*speak*" akan memberikan *tag* "*end of question being spoke*" pada kalimat terakhir yang diucapkan oleh *TextToSpeech*.



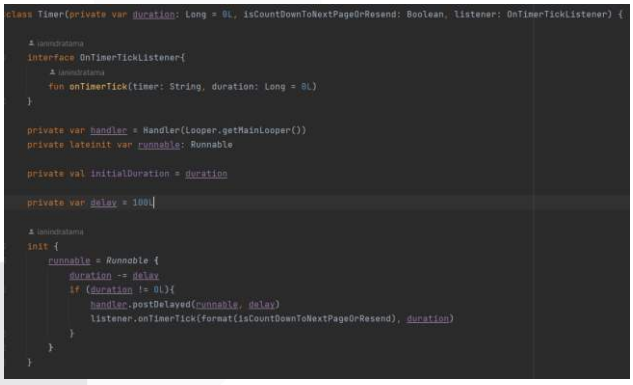
GAMBAR 6.
Inisialisasi dan pembacaan pertanyaan menggunakan komponen TextToSpeech

Dengan menggunakan *tag*, dapat melakukan *override* pada metode *onDone* yang dieksekusi oleh *komponenTextToSpeech* ketika pertanyaan selesai dibacakan. Penting untuk diingat, metode *onDone* akan dieksekusi oleh *komponenTextToSpeech* ketika pertanyaan selesai dibacakan. Oleh karena itu, dengan menggunakan *tag* yang telah kita buat sebelumnya, maka kita dapat melakukan *override* pada metode *onDone* lalu kemudian menambahkan percabangan logika yang dimana ketika kalimat dengan *tag* "*end of question being spoke*" selesai dibacakan maka aplikasi akan secara otomatis menekan tombol *record* untuk memulai perekaman suara.



GAMBAR 7.
Override metode onDone pada komponen TextToSpeech

Ketika tombol *record* ditekan, maka aplikasi akan mengeksekusi metode *startRecording* yang bertugas untuk memulai perekaman pada komponen *Media Recorder* dan memulai pencatatan durasi waktu jawab pengguna pada komponen *Timer*. Penting untuk diingat, komponen *Timer* yang dibuat pada aplikasi telah diatur untuk mengeksekusi metode yang komponen *Timer* miliki yaitu "*onTimerTick*" setiap 100 milisekon.



GAMBAR 8.
Struktur kelas dari komponen Timer

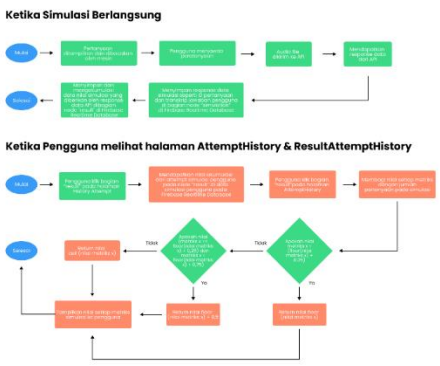
Hal ini dibuat agar komponen *Timer* dapat memberi tahu aplikasi bahwa ketika durasi menjawab pertanyaan sudah kurang dari 1 menit maka ketika tombol *stop record* ditekan, maka aplikasi tidak akan menampilkan pertanyaan selanjutnya dan ketika durasi sudah habis maka aplikasi akan secara otomatis mengeksekusi fungsi *stopRecording*.

GAMBAR 9.
Fungsi onTimerTick

Fungsi *stopRecording* akan dieksekusi aplikasi ketika durasi waktu jawab pengguna pada komponen *Timer* sudah habis (5 menit untuk *section* 1 & 3 dan 2 menit untuk *section* 2). Selain ketika durasi komponen *Timer* habis, fungsi *stopRecording* akan dieksekusi juga ketika pengguna menekan tombol *stop record*. Fungsi *stopRecording* berfungsi untuk menghentikan sementara (*pause*) atau menghentikan selamanya komponen *Timer*. Setelah itu, aplikasi akan memberhentikan rekaman komponen *MediaRecorder*.

3. Sistem Asesmen Keseluruhan dari Simulasi

Sistem ini bekerja saat aplikasi mendapatkan data *response* dari API, aplikasi akan langsung meneruskan data *response* tersebut untuk disimpan di *Firebase Realtime Database*. Data *response* API yang berupa 4 nilai matriks penilaian tes *IELTS* bagian *speaking* disimpan ke *node* "*results*" yang ada pada *node* "*simulasi*" pada *Firebase*. Penting untuk diingat, proses menyimpan data nilai pada *Firebase* ini dilakukan pada setiap berkas audio yang dikirimkan ke API, sehingga sistem menyimpan nilai pada node "*results*" bersifat akumulasi.



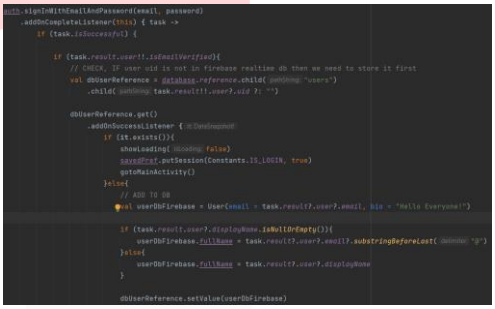
GAMBAR 10.
Alur kerja sub-sistem "Sistem Asesmen Keseluruhan dari Simulasi"

Setelah pengguna menyelesaikan simulasi, pengguna dapat melihat *attempt* simulasi pengguna pada menu *History*. Sebelum halaman tampil, aplikasi akan melakukan perhitungan untuk membagi nilai setiap matriks penilaian dengan jumlah pertanyaan yang ada pada simulasi. Setelah mendapatkan hasilnya, aplikasi akan lanjut untuk melakukan pembulatan nilai. Pada aturan pembulatan yang diterapkan oleh lembaga IELTS, dapat disimpulkan bahwa jika nilai berada direntang kurang dari .25 maka nilai akan dibulatkan kebawah, namun jika nilai berada direntang lebih besar atau sama dengan .25 namun kurang dari .75 maka nilai akan dibulatkan ke setengah *band* berikutnya. Selanjutnya, jika nilai lebih dari .75 maka nilai akan dibulatkan keatas (*band* selanjutnya).

4. Sistem Manajemen Pengguna

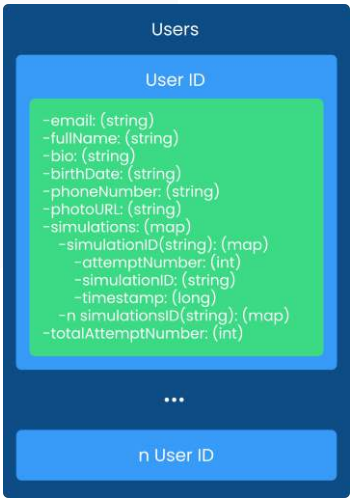
Sistem manajemen dimulai saat pengguna melakukan proses *register* akun di halaman *Register*. Ketika pengguna selesai memasukkan data akun, maka aplikasi akan membuat akun di *Firebase Authentication*. Penting untuk diingat, di momen ini pengguna akan mendapatkan surel verifikasi akun yang akan berisi *link Firebase Dynamic Link* yang akan digunakan nanti ketika pengguna berhasil terverifikasi agar bisa masuk kembali ke aplikasi.

Di halaman *Login*, ketika pengguna sudah memasukkan email dan *password*, maka aplikasi akan melakukan proses autentikasi *login* menggunakaan *Firebase Authentication*. Setelah itu, jika ini adalah pertama kalinya pengguna *login* menggunakan akun yang baru dibuat, maka aplikasi akan menyimpan data pengguna di *Firebase Realtime Database*.



GAMBAR 11.
Script kode proses *login* aplikasi pada halaman *Login*

Selanjutnya, sistem manajemen dipakai saat pengguna mengunjungi halaman *profile*. Secara struktur lengkap, berikut adalah bagaimana struktur akun pengguna tersimpan pada *node* "*users*" pada *Firebase Realtime Database*.



GAMBAR 12.
Struktur node "users" pada sistem manajemen pengguna Firebase Realtime Database

Terakhir, sistem manajemen digunakan saat aplikasi mendapatkan *response* data dari *request* yang dikirimkan ke API. Pada tahap ini, aplikasi akan terlebih dahulu menyimpan data *id* pertanyaan dan transkrip jawaban pada *node*

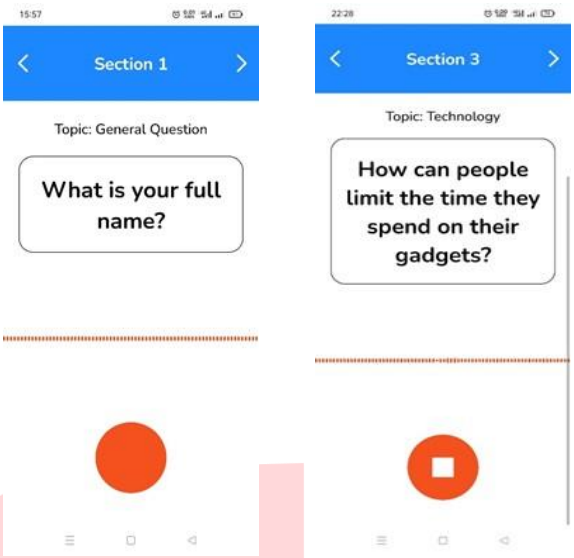
"*simulationID*" yang berada di dalam node "*simulations*" pada *Firebase Realtime Database*. Selanjutnya, setelah berhasil menyimpan data *id* pertanyaan dan transkrip jawaban, maka langkah selanjutnya adalah untuk menyimpan 4 nilai matriks yang ada di *response* data ke *node* "*results*" yang berada di dalam *node* "*simulations*" pada *Firebase Realtime Database*. Sistem akan menyimpan nilai tersebut menggunakan metode "*runTransaction*" yang dimana akan mengakumulasikan nilai sebelumnya dengan nilai sekarang dan memastikan tidak adanya konflik ketika proses akumulasi nilai.

Secara struktur lengkap, berikut adalah bagaimana struktur sebuah data simulasi tersimpan pada *node* "*simulationID*" pada *Firebase Realtime Database*. Penting untuk diingat, *Firebase Realtime Database* adalah *database* dengan struktur *NoSQL*, sehingga setiap data yang ada pada struktur tidaklah harus semua diisi untuk mengisi *record* pada tabel.

GAMBAR 13.
Struktur node "simulationID" pada sistem manajemen pengguna Firebase Realtime Database

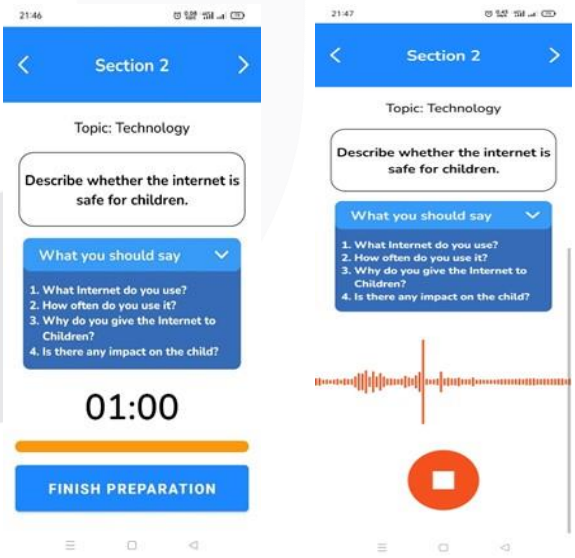
5. Pengembangan *User Interface / User Experience* Aplikasi

Pada kegiatan simulasi di aplikasi, terdapat empat tampilan halaman. Dua halaman pertama untuk halaman simulasi *Section* 1 & 3. Kedua halaman ini menampilkan komponen yang sama, hanya berbeda pada isi pertanyaannya.



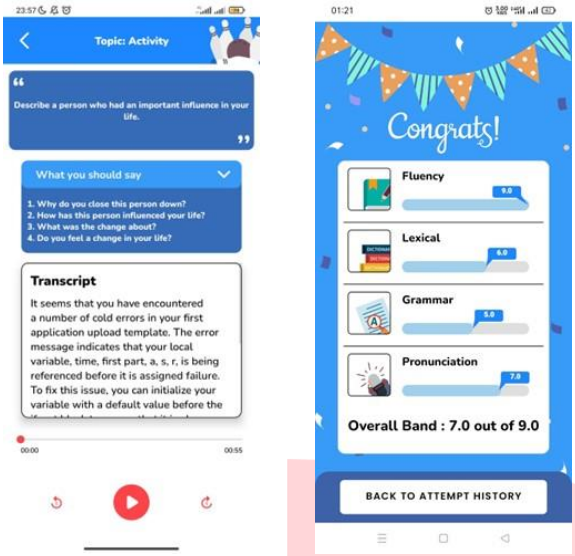
GAMBAR 14.
Implementasi halaman simulasi *Section* 1 & 3

Selanjutnya, dua halaman kedua diperuntukkan khusus untuk *Section* 2. Terdapat dua halaman untuk *Section* 2 yaitu halaman Section 2 Preparation dan Section 2. Halaman Section 2 Preparation digunakan agar pengguna dapat mempersiapkan jawaban monolog untuk menjawab pertanyaan *Section* 2. Kemudian untuk halaman Section 2 sendiri, tampilannya kurang lebih sama, hanya berbeda di bagian bawah yang menampilkan komponen *custom* bernama "*AudioWaveFormView*" yang digunakan untuk memvisualisasikan suara dari pengguna ketika menjawab pertanyaan dan komponen *ImageButton* yang dapat digunakan pengguna ketika ingin menghentikan perekaman jika sudah selesai menjawab pertanyaan.



GAMBAR 15.
Implementasi halaman simulasi *Section* 2 Preparation dan *Section* 2

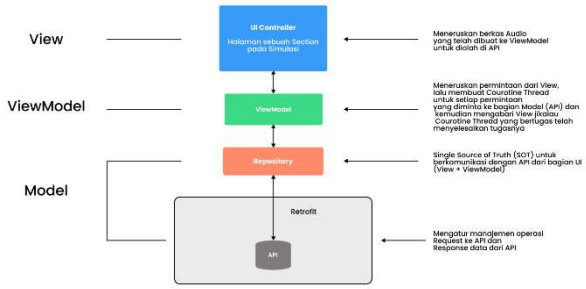
Selanjutnya, ketika pengguna selesai melakukan simulasi, maka pengguna dapat melihat *history* dari *attempt* yang telah dia ambil. Di bagian History, pengguna dapat melihat nilai simulasi beserta dengan data – data pertanyaan yang telah Ia jawab pada simulasi.



GAMBAR 16.
Implementasi halaman History Attempt Section 2 dan Result

6. Sistem Penghubung Antara Android dan API
   Proses pengembangan

Secara struktur, seperti yang dijelaskan pada sub-sistem "Database Topik dan Pertanyaan", aplikasi menggunakan pendekatan arsitektur MVVM (*Model View ViewModel*), sehingga secara umum pada kasus pengiriman *request* ke API, aplikasi memisahkan antara bagian tampilan (*View*), pembuatan *Coroutines Thread* (*ViewModel*), dan proses *request* API (*Model*). Sebagai gambaran, berikut adalah hubungan antar setiap komponen MVVM saat proses *request* API di setiap *section*.



GAMBAR 17.
Struktur MVVM dari aplikasi terkait sistem penghubung antara Android dan API

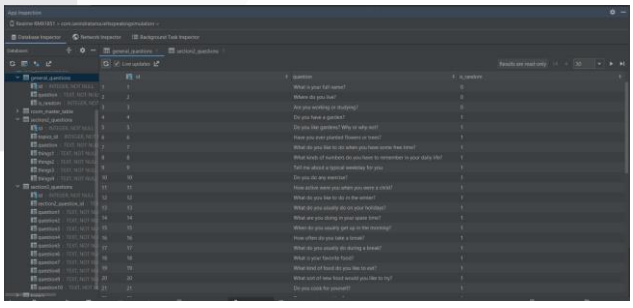
Alur dari implementasi sistem penghubung antara *Android* dan API diawali ketika pengguna telah selesai menjawab pertanyaan simulasi dan kemudian menghentikan *recording*. Pada tahap ini, berkas audio akan diteruskan ke bagian *ViewModel* dari *section* pada simulasi. Penting untuk diingat, proses pengiriman berkas audio ke API bersifat asinkron agar pengguna bisa melanjutkan ke pertanyaan berikutnya tanpa harus menunggu proses pengiriman berkas audio ke API dan penyimpanan data simulasi dan nilai ke *Firebase* selesai. Untuk mengimplementasikan proses asinkron tersebut, maka aplikasi akan membuat sebuah *thread* terpisah dari UI *thread* yang bernama *Courotines Thread*. Penting untuk diingat bahwa setiap pengiriman berkas audio berada di *Courotines Thread* yang berbeda-beda untuk tercipta proses yang *seamless*, sehingga masing-masing proses *request* ke API bisa berjalan sendiri tanpa menunggu yang lainnya. Proses pengiriman berkas audio ke API dimulai ketika berkas audio diterima oleh *ViewModel* dan kemudian diteruskan ke bagian *Repository*. Setelah *Repository* menerima berkas audio, maka *Repository* akan membuat *request* ke API dan menyertakan berkas *recording* pengguna melalui *Retrofit*. Melalui *Retrofit*, *Repository* akan menerima *response* data dari API untuk kemudian diolah kembali didalam *Courotine Thread*. Jika *response code* dari API *request* yang dikirimkan bukanlah 200 (*OK / Success*), maka *Courotine Thread* akan langsung memberi tahu *ViewModel* bahwa proses *request* gagal dan kemudian *ViewModel* akan langsung meneruskan informasi ini ke *View* / halaman simulasi *section* yang kemudian akan memberi tahu pengguna bahwa proses *upload* berkas audio gagal dan kemudian mengarahkan pengguna ke *home screen*. Di sisi lain, jikalau *response code* dari API *request* yang dikirim adalah 200 (*OK / Success*), maka *Courotine Thread* akan menyimpan data simulasi beserta nilainya di *Firebase* dan kemudian memberitahu *ViewModel* bahwa proses *request* API dan penyimpanan data di *Firebase* telah berhasil pada *Courotine Thread* tersebut. Terakhir, *ViewModel* akan memberi tahu *View* / halaman simulasi *section* bahwa proses dari *Courotine Thread* tersebut telah berhasil.

IV. HASIL DAN PEMBAHASAN

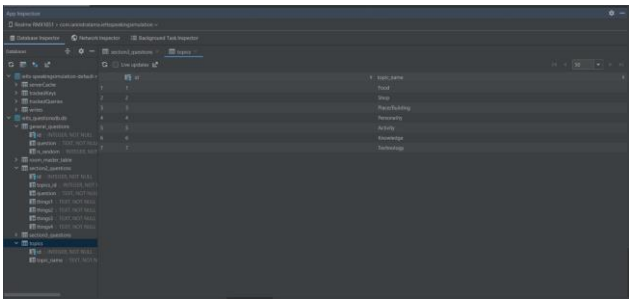
Berikut adalah hasil pengujian dan pembahasan pengujian untuk 6 sub-sistem yang diimplementasikan pada aplikasi Android:

1. *Database* Topik dan Pertanyaan

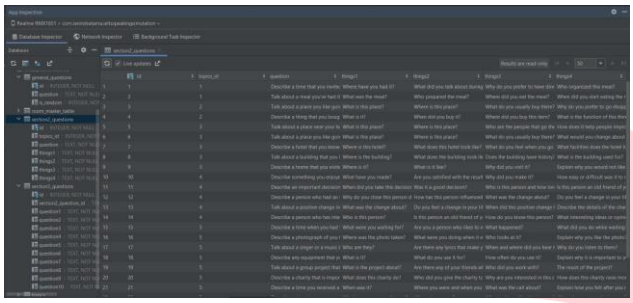
Pengujian pembuatan *database* topik dan pertanyaan ini, akan dilakukan dengan menampilkan *database* topik dan pertanyaan pada *Android Room Database*. Dari *dataset* yang sudah dibuat selanjutnya *dataset* topik dan pertanyaan akan dimasukkan ke *database Android* untuk dapat ditampilkan di aplikasi. Seperti yang ditampilkan pada Gambar 3. Relasi tabel pada Database, berikut adalah implementasi dari relasi tabel tersebut di aplikasi dengan menggunakan Android Room Database.



GAMBAR 18.
Database tabel "general_questions" pada Android Room Database



GAMBAR 19.
Database tabel "topics" pada Android Room Database



GAMBAR 20.
Database tabel "section2_questions" pada Android Room Database

GAMBAR 21.
Database tabel "section3_questions" pada Android Room Database

Dengan demikian, maka dapat dikatakan bahwa *database* topik dan pertanyaan sudah dapat digunakan dengan baik di aplikasi.

2. Sistem Pengaturan Waktu Menjawab Pertanyaan Simulasi

Sistem pengaturan waktu menjawab pertanyaan simulasi melakukan pengujian alfa dengan metode *Black Box*. Skenario pengujian fokus pada aksi yang seharusnya terjadi di aplikasi dari sisi waktu pengguna menjawab pertanyaan ketika mengikuti simulasi dari sudut pandang pengembang aplikasi.

$$Uji\ Alfa = \frac{Total\ berhasil}{Total\ pengujian}\ x\ 100\% \qquad (1)$$

$$Uji\ Alfa = \frac{Total\ Berhasil}{10\ x\ 9\ Aksi}\ x\ 100\% = \frac{90}{90}\ x\ 100\%$$

$$= 100\%$$

Hasil nilai uji alfa terhadap setiap aksi yang diuji pada sistem pengaturan waktu menjawab pertanyaan simulasi adalah 100%. Maka dapat dikatakan aplikasi berjalan sesuai tujuan dan dapat diterima.

3. Sistem Asesmen Keseluruhan dari Simulasi

*S*istem asesmen keseluruhan melakukan pengujian alfa dengan metode Black Box. Skenario pengujian alfa akan fokus pada nilai yang harusnya terakumulasi di *Firebase Realtime Database* ketika pengguna melakukan simulasi dan pembulatan nilai akhir simulasi pengguna dari sudut pandang pengembang aplikasi.

$$Uji\ Alfa = \frac{Total\ Berhasil}{10\ x\ 2\ Aksi}\ x\ 100\%$$

$$= \frac{20}{20}\ x\ 100\% = 100\%$$

Hasil nilai uji alfa terhadap setiap aksi yang diuji pada sistem asesmen keseluruhan dari simulasi tes *IELTS* bagian *speaking* adalah 100%. Maka dapat dikatakan aplikasi berjalan sesuai tujuan dan dapat diterima.

4. Sistem Manajemen Pengguna

Pengujian alfa pada sistem manajemen pengguna menggunakan metode *Black Box*. Skenario pengujian akan fokus pada kesesuaian data yang diinputkan di aplikasi dengan yang ada di *Firebase* dari sudut pandang pengembang aplikasi.

$$Uji\ Alfa = \frac{Total\ Berhasil}{10\ x\ 8\ Aksi}\ x\ 100\%$$

$$= \frac{80}{80}\ x\ 100\% = 100\%$$

Hasil nilai uji alfa terhadap setiap aksi yang diuji pada sistem manajemen pengguna adalah 100%. Maka dapat dikatakan aplikasi berjalan sesuai tujuan dan dapat diterima.

5. Pengembangan *User Interface / User Experience* Aplikasi

Pengembangan UI / UX aplikasi simulasi *IELTS* sesi *speaking* melakukan pengujian alfa dengan metode *Black Box* untuk menguji apakah semua komponen sudah berjalan sesuai dengan yang diharapkan. Selanjutnya, sub-sistem ini juga melakukan pengujian performa aplikasi dengan mengamati secara langsung performa aplikasi dengan menggunakan alat *Android Studio Profiler* dan *Cx File Explorer* untuk mengetahui penggunaan maksimal *RAM* ketika menggunakan aplikasi dan besaran penyimpanan yang dipakai oleh aplikasi.

Pengujian alfa dilakukan sebanyak 10 kali untuk masing masing aksi, dari pengujian menggunakan persamaan (1) maka diperoleh perhitungan akurasi uji alfa sebagai berikut:

$$Uji\ Alfa = \frac{Total\ Berhasil}{10\ x\ 100\ Aksi}\ x\ 100\%$$

$$= \frac{1000}{1000}\ x\ 100\% = 100\%$$

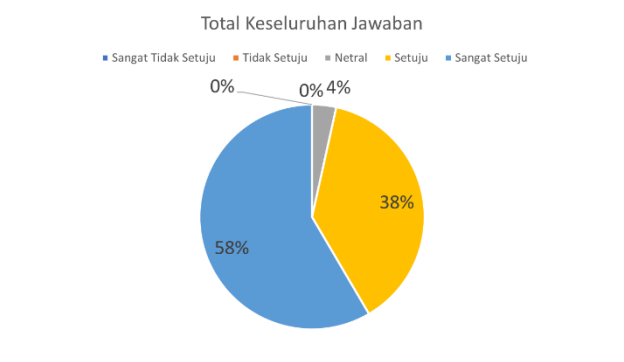
Hasil nilai uji alfa terhadap setiap aksi yang diuji pada implementasi UI/UX aplikasi simulasi *IELTS* sesi *speaking* pada *Android* adalah 100%. Maka dapat dikatakan aplikasi berjalan sesuai tujuan dan dapat diterima.

Selanjutnya, dari pengujian performa aplikasi yang telah dilaksanakan, didapatkan bahwa penggunaan RAM ketika pengguna menggunakan aplikasi stabil di kisaran 90 – 220 MB, besaran ukuran aplikasi ketika dipasang di gawai pengguna adalah 26 MB, dan besaran total ukuran berkas audio yang dibuat pengguna ketika melakukan sebuah simulasi adalah 1.12 MB. Berdasarkan hasil pengujian ini, kami tetapkan bahwa berikut adalah *minimum system requirements* dari aplikasi:

TABEL I.
Minimum System Requirements

Minimum Requirement
Android OS 8.0 (Oreo)
Android *Device Minimum* RAM: 1.5 GB (Rekomendasi: 2 GB atau lebih)
App Download Size: 9 MB
App Install Size: 26 MB
Device Minimum Available App Disk Space Recommendation: 48.4 MB
App Permissions: "Photos/Media/Files", "Camera", "Storage", "Microphone", "Internet"

Terakhir, pada pengujian beta yang dilakukan kepada 33 responden secara daring yang diberikan beberapa pertanyaan kuesioner, didapatkan bahwa dari keseluruhan pertanyaan kuesioner terkait sub-sistem ini, 96% responden memilih setuju dan sangat setuju terhadap UI / UX dari aplikasi.



GAMBAR 22.
Total keseluruhan jawaban pengujian beta pengembangan UI / UX aplikasi

6. Sistem Penghubung Antara Android dan API

Pada sub-sistem ini, dilakukan pengujian alfa menggunakan metode *Black Box*. Skenario pengujian alfa akan fokus pada proses dan aksi dari setiap *instance* Courotines yang dibuat untuk mengerjakan *background task* yang diberikan kepada masing - masing *instance* dari sudut pandang pengembang aplikasi.

$$Uji\ Alfa = \frac{Total\ Berhasil}{10\ x\ 3\ Aksi}\ x\ 100\%$$

$$= \frac{30}{30}\ x\ 100\% = 100\%$$

Hasil nilai uji alfa terhadap setiap aksi yang diuji pada sistem penghubung antara *Android* dan API adalah 100%. Maka dapat dikatakan aplikasi berjalan sesuai tujuan dan dapat diterima.

V. KESIMPULAN

Berdasarkan hasil penelitian, implementasi, pengujian, serta analisa yang telah dilakukan dalam pengembangan aplikasi simulasi sesi *speaking* tes *IELTS* berbasis *Android*, dapat disimpulkan bahwa *database* topik dan pertanyaan yang digunakan pada aplikasi mampu untuk menampilkan pertanyaan untuk setiap *Section* yang ada pada tes IELTS sesi *speaking*. Kemudian, untuk sistem pengaturan waktu beroperasi yang ditetapkan pada aplikasi telah berjalan dengan baik dalam membatasi waktu jawab pengguna di setiap *Section* yang terdapat pada simulasi. Hal ini didukung dengan hasil nilai uji alfa untuk setiap aksi pada skenario uji alfa adalah 100%. Untuk sistem asesmen keseluruhan dari simulasi tes IELTS bagian *Speaking* pada aplikasi juga telah beroperasi dengan baik dalam mengatur agar tidak terjadinya *collision* saat proses akumulasi nilai yang berjalan secara asinkron ketika aplikasi mengirimkan berkas Audio ke API. Selain itu, proses pembulatan nilai untuk setiap metriks penilaian sesi *Speaking* tes IELTS juga telah memenuhi ketentuan pembulatan nilai dari lembaga IELTS. Hal ini didukung dengan hasil nilai uji alfa untuk setiap aksi pada skenario uji Alfa adalah 100%. Selanjutnya, untuk sistem manajemen pengguna pada aplikasi telah beroperasi dengan baik dalam mengatur data simulasi dan data akun pengguna aplikasi. Hal ini didukung dengan hasil nilai uji alfa untuk setiap aksi pada skenario uji Alfa adalah 100%. Lalu aplikasi simulasi IELTS sesi *speaking* juga telah beroperasi dengan semestinya, baik dari sisi tampilan yang ditampilkan ke pengguna maupun dari sisi hal – hal yang terjadi dibelakang layar pada aplikasi. Melalui pengujian Alfa, diperoleh hasil nilai uji Alfa yang sebesar 100% untuk setiap aksi yang diuji pada skenario pengujian Alfa. Selain itu, dari hasil pengamatan pengujian performa aplikasi, diperoleh bahwa performa aplikasi untuk keseluruhan fitur pada aplikasi stabil menggunakan RAM dari perangkat Android pengguna di *range* 90 – 220 MB. Pembuktian aplikasi simulasi IELTS sesi *Speaking* berjalan dengan baik juga didukung dengan 96% dari total 33 responden pada pengujian beta yang memilih setuju dan sangat setuju terhadap UI / UX dari aplikasi. Terakhir, sistem penghubung antara Android dan API pun telah beroperasi dengan semestinya. Hal ini didukung oleh hasil nilai uji Alfa yang diraih pada setiap aksi yang diuji pada skenario pengujian Alfa sistem penghubung antara Android dan API adalah 100%.

Secara keseluruhan, setiap sistem yang diterapkan untuk menunjang aplikasi simulasi sesi *speaking* tes *IELTS* berbasis *Android* telah berjalan sesuai dengan yang direncanakan. Aplikasi telah dapat mensimulasikan tes *IELTS* sesi *speaking* dengan menerapkan semua aturan yang bisa diterapkan dari sisi pengembangan aplikasi (*software*).

REFERENSI

[1] Universitas Cambridge; British Council; IDP Education Australia, "What is IELTS?," [Online]. Available: https://www.ielts.org/about-ielts/what-is-ielts. [Diakses 17 Oktober 2022].

[2] A. Hashemi dan S. Daneshfar, "A Review of the IELTS Test: Focus on Validity, Reliability, and Washback," *Indonesian Journal of English Language Teaching and Applied Linguistics,* vol. 3, no. 1, pp. 42-43, 2018.

[3] IELTS, "Test taker performance 2022," [Online]. Available: https://www.ielts.org/for-researchers/test-statistics/test-taker-performance. [Diakses 1 Agustus 2023].

[4] IELTS, "IELTS scoring in detail," [Online]. Available: https://www.ielts.org/for-organisations/ielts-scoring-in-detail. [Diakses 1 Agustus 2023].

[5] Universitas Cambridge; British Council; IDP Education Australia, "IELTS practice test," [Online]. Available: https://www.ielts.org/usa/ielts-practice-test. [Diakses 12 November 2022].

[6] Universitas Cambridge; British Council; IDP Education Australia, "IELTS Academic – Practice Test 1 (Timed)," IELTS Progress Check, [Online]. Available: https://www.ieltsprogresscheck.com/product/ielts-academic-practice-test-1-timed/. [Diakses 12 November 2022].

[7] Universitas Cambridge; British Council; IDP Education Australia, "What is IELTS Progress Check?," IELTS Progress Check, [Online]. Available: https://www.ieltsprogresscheck.com/#:~:text=What%20is%20IELTS%20Progress%20Check%3F. [Diakses 12 November 2022].

[8] M. Ariffudin, "Mengenal Kotlin, Bahasa Pemrograman untuk Aplikasi Android," Niaga Hoster, 20 November 2021. [Online]. Available: https://www.niagahoster.co.id/blog/kotlin-adalah/. [Diakses 1 Desember 2022].

[9] Android Developers, "MediaRecorder," Google, [Online]. Available: https://developer.android.com/reference/android/media/MediaRecorder. [Diakses 1 Agustus 2023].

[10] Android Developers, "Kotlin coroutines on Android," Google, [Online]. Available: https://developer.android.com/kotlin/coroutines. [Diakses 1 Agustus 2023].

[11] Dicoding Intern, "Apa itu Firebase? Pengertian, Jenis-Jenis, dan Fungsi Kegunaannya," Dicoding, 25 November 2020. [Online]. Available: https://www.dicoding.com/blog/apa-itu-firebase-pengertian-jenis-jenis-dan-fungsi-kegunaannya/. [Diakses 1 Desember 2022].

[12] Google Developers, "Dokumentasi Firebase," [Online]. Available: https://firebase.google.com/docs?hl=id. [Diakses 1 Februari 2023].