BAB 3

METODOLOGI PENELITIAN

3.1 Metodologi Penelitian

Metodologi penelitian yang digunakan pada penelitian ini adalah sebagai berikut.

a. Telaah literatur

Pada tahap literatur, dilakukan pengumpulan informasi yang berkaitan dengan aplikasi *chatbot* Jacob, *text mining*, IndoBERT-lite, RoBERTa, SQuAD, dan TyDI QA. Pengumpulan informasi dilakukan dengan cara membaca berbagai sumber yang terkait dengan judul penelitian melalui jurnal, *conference proceeding*, *thesis*, dan buku.

b. Analisis dan Perancangan

Pada tahap analisis dan perancangan, langkah pertama diawali dengan mengnalisis cara penggunaan, skema kerja dan bahasa pemrograman yang dipakai pada aplikasi *chabot* Jacob. Selanjutnya melakukan perancangan sistem *text mining* secara daring berupa *web service* pada *chatbot* Jacob. *Web service* dirancang dengan tujuan agar *chatbot* Jacob mendapatkan beberapa informasi Berbahasa Indonesia terkait dengan pertanyaan yang diberikan *user*. Lalu, dilakukan analisis untuk menerapkan proses *pre-training* dan *fine-tuning* pada *pre-trained* model RoBERTa. Setelah mendapatkan *pre-trained model*, kemudian menganalisis cara menerapkan *pre-trained model* IndoBERT-lite dan RoBERTa pada aplikasi *chatbot* Jacob sehingga *chatbot* dapat memberikan jawaban yang sesuai dengan pertanyaan *user*.

c. Implementasi

Pada tahap implementasi, dilakukan implementasi pada *chatbot* Jacob dengan menerapkan rancangan yang dibuat pada tahap analisis dan perancangan. Proses implementasi terdiri dari 2 yaitu proses *training* dan proses *predict answer*. Pada proses *training*, diawali dengan melakukan proses *pre-training* untuk mendapatkan *pre-trained model* RoBERTa dalam Bahasa Indonesia. Kemudian dilakukan proses *fine-tuning* dengan *dataset* SQuAD dan TyDI Qa untuk mendapatkan *fine-tuned model* RoBERTa yang digunakan untuk melakukan *question answering*. Implementasi pada proses *pre-training* dan *fine-tuning* dilakukan dengan menggunakan bantuan Google Colaboratory. Pada proses *predict answer*, *web service* akan melakukan proses pencarian jawaban yang terkait dengan pertanyaan *user* setelah mendapatkan hasil dari Google Custom Search API dengan menimplementasikan *fine-tuned* model IndoBERT-lite dan RoBERTa. *Web service* dibangun dengan menggunakan bahasa pemrograman Python. *Fine-tuned model* IndoBERT-lite dan RoBERTa akan menghasilkan output berupa jawaban dari pertanyaan *user*. Jawaban tersebut kemudian akan disampaikan kepada *user*.

d. Pengujian

Pada tahap pengujian, dilakukan pengujian terhadap proses *pre-training* dan *fine-tuning model* dengan mengukur nilai akurasi yang dihasilkan. Selain itu, kemampuan dalam melakukan *text mining* Berbahasa Indonesia dan mendapatkan jawaban yang signifikan terhadap pertanyaan *user* yang telah selesai diemplementasikan diuji. Pengujian dilakukan dengan menggunakan pendekatan *whitebox testing* untuk melihat seberapa cocok jawaban yang dihasilkan dari *fine-tuned model* IndoBERT-lite dan RoBERTa pada aplikasi *chatbot* Jacob.

e. Evaluasi

Pada tahap ini, evaluasi dilakukan untuk menentukan apakah kecerdasan baru yang ditambahkan pada aplikasi *chatbot* Jacob telah memenuhi tujuan dari penelitian ini. Evaluasi dilakukan dengan mengukur seberapa tinggi nilai akurasi, dan *F-Score* dari *fine-tuned model* IndoBERT-lite dan RoBERTa yang digunakan.

f. Penulisan Naskah Penelitian

Pada tahap ini, hasil akhir dari pengejaan penelitian dituliskan dalam bentuk bentuk laporan sebagai bukti atas dilakukannya penelitian ini.

3.2 Analisis Kebutuhan

Hasil analisis kebutuhan terhadap aplikasi *chatbot* Jacob terdiri dari hasil analisis *chatbot* Jacob yang telah dibangun oleh peneliti sebelumnya, kemudian terdapat juga hasil analisis terhadap penerapan *web service* dan *chatbot* Jacob Bahasa Indonesia. Berdasarkan hasil analisis pada aplikasi *chatbot* Jacob yang telah dibangun oleh peneiliti sebelumnya, diketahui bahwa:

a. Bahasa yang digunakan sebagai *input* dan *output* pada Aplikasi *chatbot* Jacob adalah Bahasa Inggris.

b. Apliksasi *chatbot* Jacob menggunakan satu *app* dengan menggunakan Bahasa Inggris pada platform Wit.ai untuk mendapatkan nilai dari *intent* dan *entities* dari suatu kalimat.

c. Pencarian informasi secara daring dilakukan dengan menggunakan bantuan Google Custom Search Engine API yang memiliki pengaturan Bahasa Inggris sehingga dokumen yang didapat untuk proses *text mining* berbasis Bahasa Inggris.

d. Prediksi jawaban dari informasi yang diperoleh secara daring dilakukan dengan menggunakan *pre-trained model* ALBERT dengan bantuan *library* Transformers pada bahasa pemrograman Python.

e. Aplikasi *chatbot* Jacob dibangun dengan menggunakan bahasa pemrograman PHP dengan *framework* Laravel dan juga menggunakan bahasa pemrograman Python pada *web service* dengan menggunakan *framework* Flask.

f. Tabel pada *database chatbot* Jacob hanya terdapat tabel untuk menyimpan respon yang akan diberikan dalam Bahasa Inngris.

Hasil analisis terhadap penerapan *web service* dan *chatbot* Jacob Berbahasa Indonesia adalah sebagai berikut:

a. Tambahan bahasa lain berupa Bahasa Indonesia dengan menambahkan opsi yang memungkinkan *user* dapat memilih bahasa yang ingin dipakai saat berinteraksi dengan *chatbot* Jacob.

b. Penambahan *app* baru pada *platform* Wit.ai dengan menggunakan Bahasa Indonesia sehingga *chatbot* Jacob dapat menerima *intent* dan *entitites* dari suatu kalimat Berbahasa Indonesia.

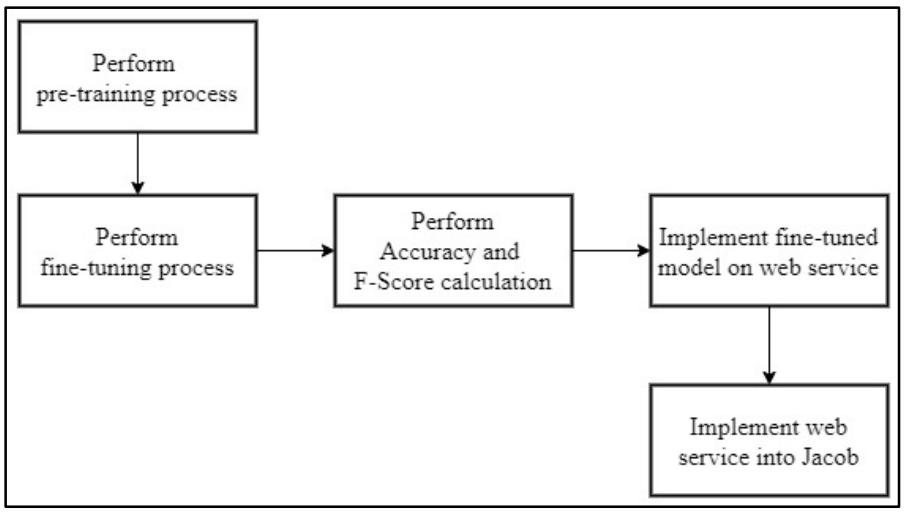
c. Penambahan Google Custom Search Engine API dengan menggunakan pengaturan Bahasa Indonesia sehingga dokumen yang dihasilkan untuk proses *text mining* berupa dokumen Bahasa Indonesia.

d. Penggunaan *fine-tuned* model IndoBERT-lite dan RoBERTa dengan bantuan *library* Transformers dan Simpletransformers pada bahasa pemrograman Python untuk melakukan prediksi jawaban dari hasil dokumen yang diperoleh secara daring.

e. Penambahan tabel baru pada *database chatbot* Jacob untuk menyimpan respon yang akan diberikan dalam Bahasa Indonesia.

3.3 Perancangan Sistem

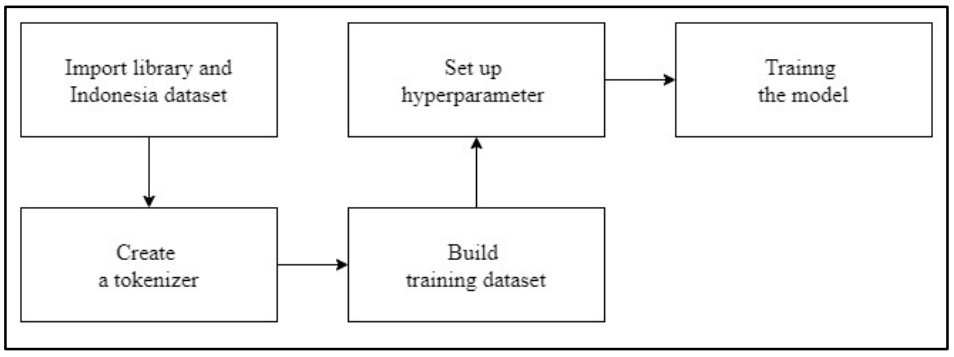
Dari hasil analisis yang telah dijabarkan pada penelitian ini, maka dilakukan perancangan sebuah *web service* dan aplikasi *chatbot* Jacob Berbahasa Indonesia. Perancangan sistem yang dilakukan, diawali dengan melakukan *pre-training model* dan dilanjutkan dengan melakukan *fine-tuning model*. Selanjutnya dilakukan perhitungan akurasi dan *F-Score* pada *fine-tuned model* yang didapatkan. Kemudian dilakukan perancangan pada *web service* dengan mengimplementasikan *fine-tuned* model tersebut. Pada tahap akhir, *web service* nantinya akan diimplementasikan pada aplikasi *chatbot* Jacob. Rancangan mengenai diagram *flowchart* utama, *flowchart web service* serta rancangan antarmuka pada halaman *web* untuk menampilkan aplikasi *chatbot* Jacob dalam Bahasa Indonesia dan menampilkan integrasi *web service* pada *chatbot* Jacob juga akan dilakukan. Alur kerja dari perancangan sistem dapat dilihat pada Gambar 3.1.



Gambar 3.1 Alur kerja perancangan sistem

3.3.1 Alur Proses Pre-training Model

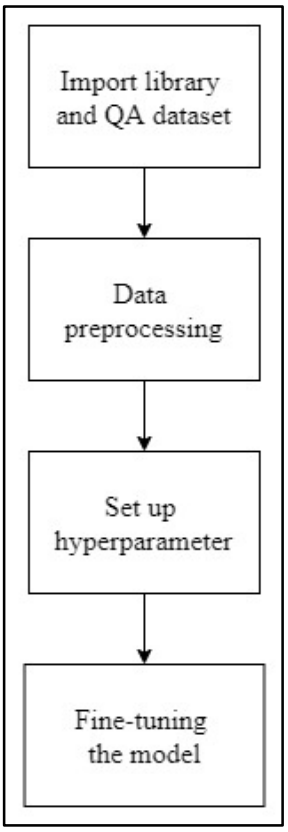
Proses *pre-training model* dilakukan dengan tujuan untuk mendapatkan *pre-trained model* RoBERTa dalam Bahasa Indonesia. Untuk melakukan proses *pre-training*, awalnya dilakukan *import dataset* terlebih dahulu. *Dataset* yang digunakan pada penelitian ini merupakan *dataset* dari Indo4B yang penjelasannya terdapat pada Bab 2. Selanjutnya dilakukan pembuatan *tokenizer* dari *dataset* tersebut. Pembuatan *tokenizer* ditujukan untuk mendapatkan daftar kosakata dan pasangan kata dalam Bahasa Indonesia yang nantinya akan dibutuhkan oleh sistem pada saat melakukan proses *pre-training*. Setelah mendapatkan *tokenizer*, kemudian dilakukan pembuatan *data training* yang akan digunakan pada *proses pre-training*. Sebelum melakukan *pre-training*, dilakukan pengaturan terlebih dahulu dengan memasukkan *hyperparameter* yang digunakan, *tokenizer* yang didapatkan, serta *data training* yang telah dibuat. Setelah melakukan konfigurasi, selanjutnya menjalankan proses *pre-training model*. Alur proses *pre-training model* dapat dilihat pada Gambar 3.2.



Gambar 3.2. Alur proses *pre-training model*

3.3.2 Alur Proses Fine-tuning Model

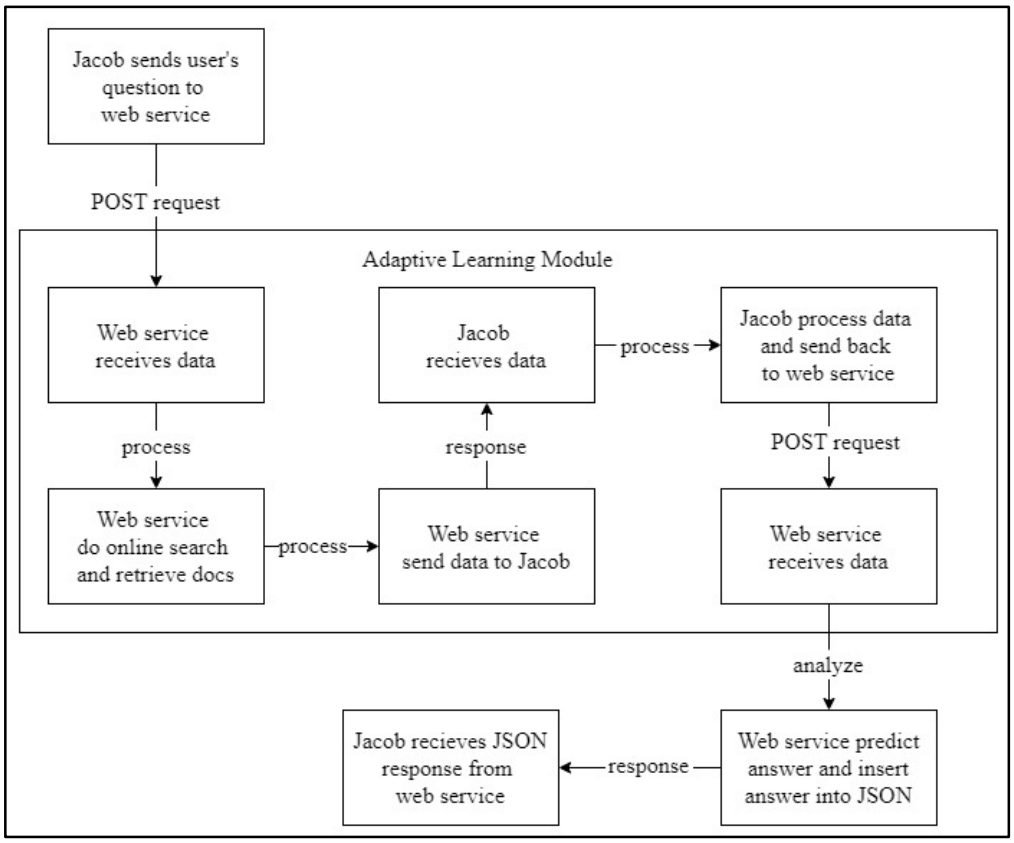
Pre-trained model yang didapatkan dari proses *pre-trainng* sebelumnya merupakan *pre-trained model* yang digunakan untuk melakukan proses *masking*, belum dapat digunakan untuk melakukan QA (*question answering*). Maka dari itu, dilakukan proses *fine-tuning* pada *pre-trained model* RoBERTa yang didapatkan dari proses *pre-training* dengan menggunakan dataset QA. Pada awalnya dilakukan *import dataset* QA. Selanjutnya dilakukan *data preprocessing* untuk mendapatkan *data training*, *testing* dan *validating*. Kemudian dilakukan konfigurasi pada *hyperparameter* yang digunakan pada proses *finetuning*. Setelah selesai melakukan konfigurasi, maka proses *fine-tuning model* dijalankan. Alur mengenai proses *fine-tuning model* dapat dilihat pada Gambar 3.3.



Gambar 3.3. Alur proses *fine-tuning model*

3.3.3 Alur Kerja Web Service

Alur kerja *web service* digunakan untuk menjelaskan cara kerja *web service* yang diawali dengan menerima *input* berupa pertanyaan *user* dari *chatbot* Jacob sampai dengan memberikan prediksi jawaban kepada *chatbot* Jacob. Gambaran mengenai alur kerja *web service* yang digunakan pada penelitian ini dapat dilihat pada Gambar 3.4.



Gambar 3.4. Alur kerja *web service*

Web service yang digunakan pada penelitian ini memiliki konsep yang sama dengan *web service* pada modul Adaptive Learning yang telah dirancang pada penelitian sebelumnya. Perbedaannya terletak pada cara kerja *pre-trained model* yang digunakan untuk memberikan prediksi jawaban kepada *user*. Pada Gambar 3.3, gambaran umum mengenai cara kerja *web service* diawali pada aplikasi *chatbot*

Jacob mengirimkan *input* berupa pertanyaan dari *user* ke *web service* dengan POST *request*. *Web service* akan menerima pertanyaan dari *user* dan menggunakannya sebagai *keyword* untuk melakukan pencarian informasi secara daring. Hasil dari pencarian yang didapatkan berupa sejumlah dokumen yang mengandung informasi terkait dengan pertanyaan *user*. Dokumen tersebut kemudian akan dikirimkan ke *chatbot* Jacob. *Chatbot* Jacob menerima data dan mengolah data tersebut. Setelah selesai diolah, *chatbot* Jacob akan mengirimkan kembali data tersebut melalui POST *request*. *Web service* kemudian akan menerima data yang telah diolah oleh *chatbot* Jacob. Data yang didapat akan digunakan oleh *web service* melalui *pre-trained model* yang diterapkan untuk melakukan prediksi jawaban yang sesuai dengan pertanyaan *user*. Jawaban hasil prediksi *pre-trained model* akan disimpan dalam bentuk JSON dan dikirimkan pada *chatbot* Jacob. *Chatbot* Jacob akan menerima data tersebut dan nantinya akan digunakan untuk memberikan respon kepada *user*.

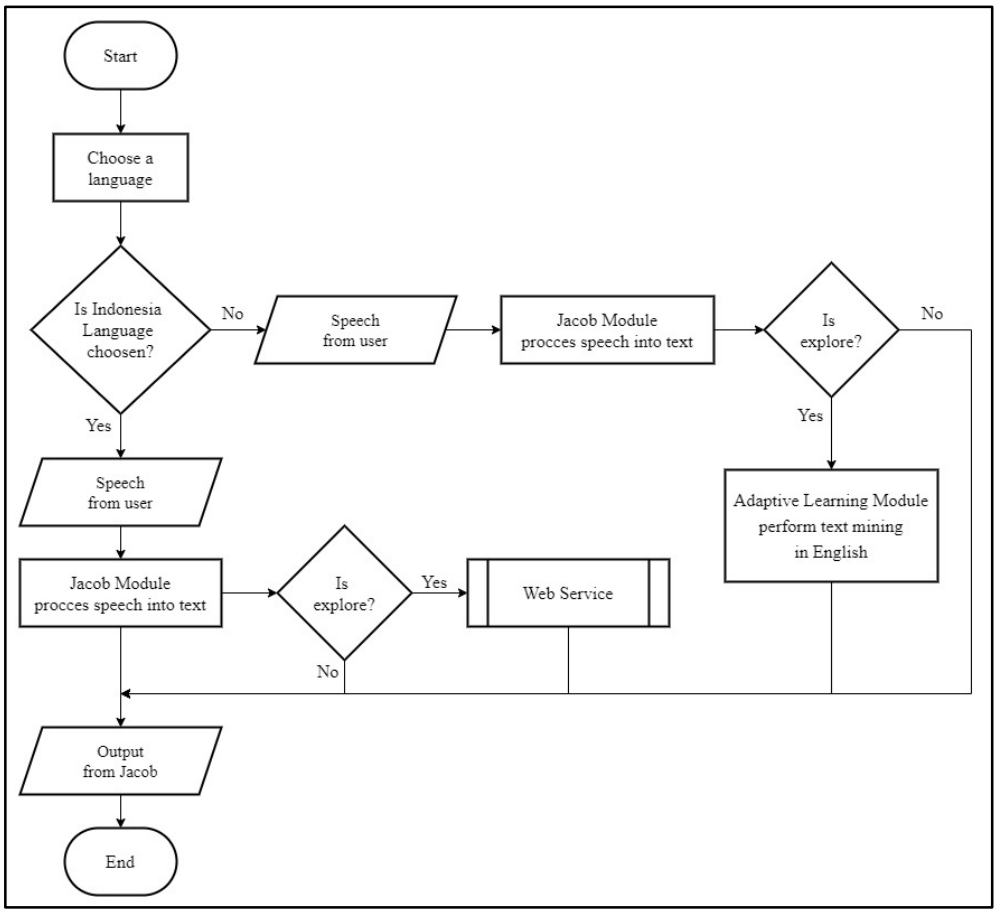
3.3.4 Flowchart

Pembuatan *flowchart* ditujukan untuk menggambarkan skema kerja penggunaan aplikasi *chatbot* Jacob Bahasa Indonesia dan skema kerja *web service* pada aplikasi *chatbot* Jacob.

A. Flowchart Utama

Flowchart utama mengambarkan skema kerja dari penggunaan aplikasi *chatbot* Jacob. Pengunaan diawali dengan pemilihan bahasa yang akan digunakan pada aplikasi *chatbot* Jacob. Pada saat *user* membuka halaman awal pada aplikasi *chatbot* Jacob, *user* akan dihadapkan denngan dua pilihan bahasa, yaitu Bahasa Inggris dan Bahasa Indonesia. Pada pengunaan Bahasa Indonesia setelah *input user*

diproses pada modul Jacob dan *user* memilih untuk melakukan pencarian informasi secara daring, maka *chatbot* Jacob akan melakukan proses *text mining* dalam Bahasa Indonesia. Sebaliknya, pada Bahasa Inggris, *chatbot* Jacob akan menggunakan Modul Adaptive Learning, dimana modul tersebut melakukan proses *text mining* dalam Bahasa Inggris. *Flowchart* mengenai pemilihan bahasa yang digunakan pada aplikasi *chatbot* Jacob dapat dilihat pada Gambar 3.5.

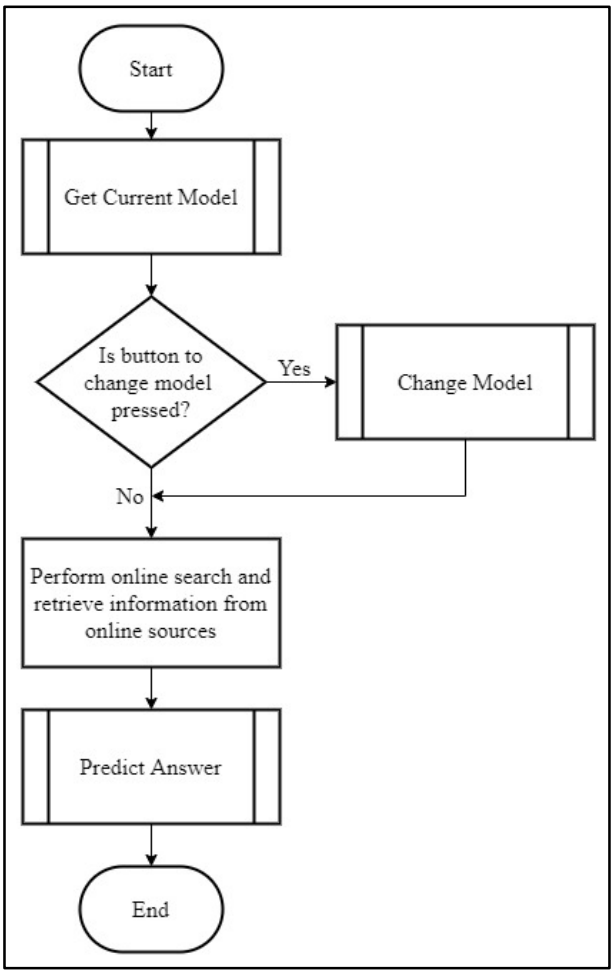


Gambar 3.5. *Flowchart* utama

B. Flowchart Web Service

Flowchart web service mengambarkan skema kerja dari *web service* yang dibangun pada penelitian ini. Skema kerja dimulai dari pengambilan *current model* yang sedang digunakan oleh *user* pada aplikasi *chatbot* Jacob. Setelah mendapatkan

current model yang digunakan, maka *user* dapat memilih untuk mengubah *model* yang ingin digunakan atau tidak mengubah *model*. Jika user memilih untuk mengubah *model*, maka *web service* akan mengubah *model* pada aplikasi *chatbot* Jacob. Selanjutnya *web service* akan melakukan pencarian dan mengumpulkan informasi secara daring. Setelah informasi mengenai pertanyaan *user* didapatkan, maka *web service* akan melakukan prediksi jawaban. *Flowchart web service* dapat dilihat pada Gambar 3.6.

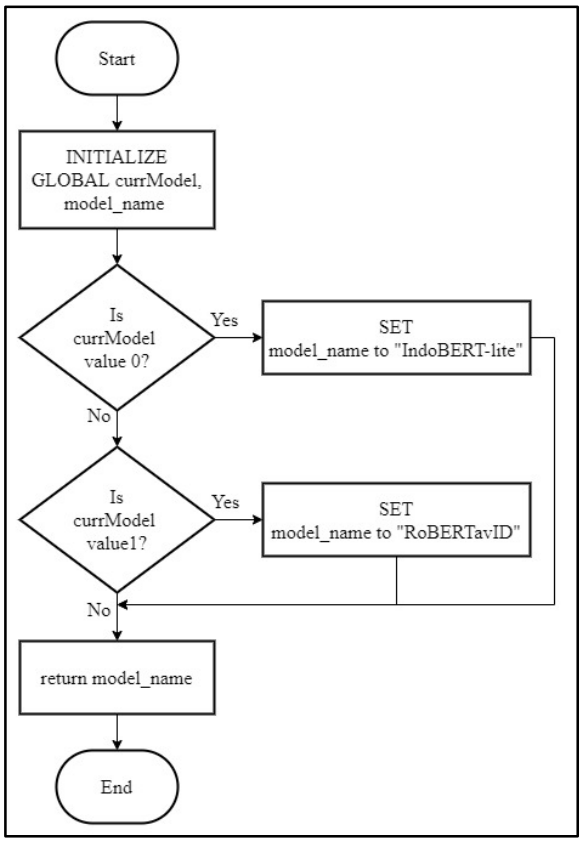


Gambar 3.6. *Flowchart web service*

B.1. Flowchart Get Current Model

Pengambilan *current model* ditujukan untuk menggunakan *pre-trained*

model yang dipilih sebagai *model* untuk melakukan prediksi jawaban dari informasi yang didapatkan secara daring. *Web service* akan mengirimkan *response* berupa *string* nama dari *pre-trained model* yang sedang digunakan yaitu “IndoBERT-lite” atau “RoBERTavID”. *Flowchart* pada Get Current Model dapat dilihat pada Gambar 3.7.

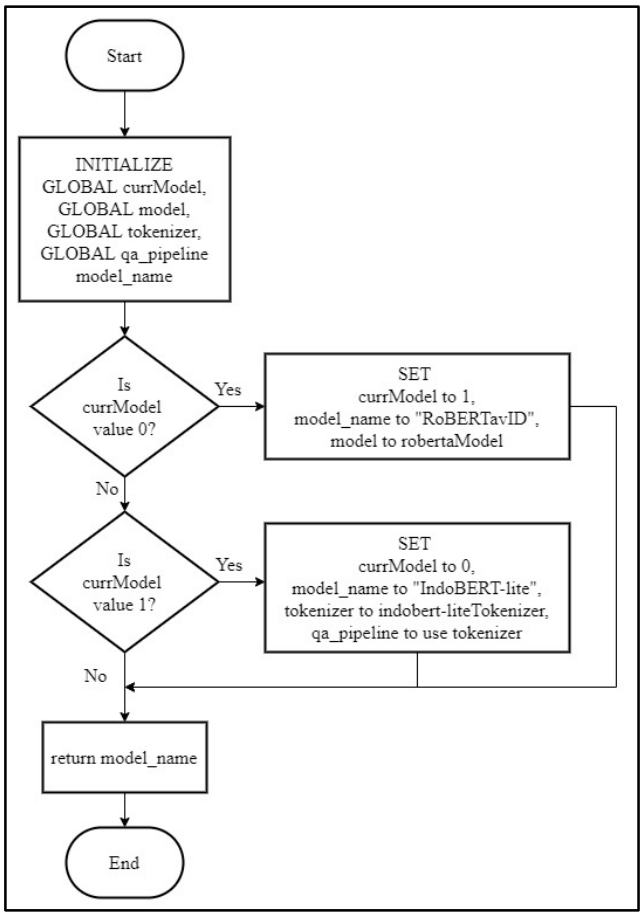


Gambar 3.7. *Flowchart* Get Current Model

B.2. Flowchart Change Model

Pada *resource* Change Model, terdapat penggunaan *tonkenizer* dan *pipeline* pada *pre-trained model* IndoBERT-lite untuk memuat ulang *pre-trained model* saat memberikan prediksi jawaban. *Response* yang akan diberikan oleh *web service* kepada aplikasi *chatbot* Jacob berupa nama *pre-trained model* yaitu “RoBERTavID” atau “IndoBERT-lite”. *Flowchart* mengenai *resource* Change

Model pada *web service* dapat dilihat pada Gambar 3.8.

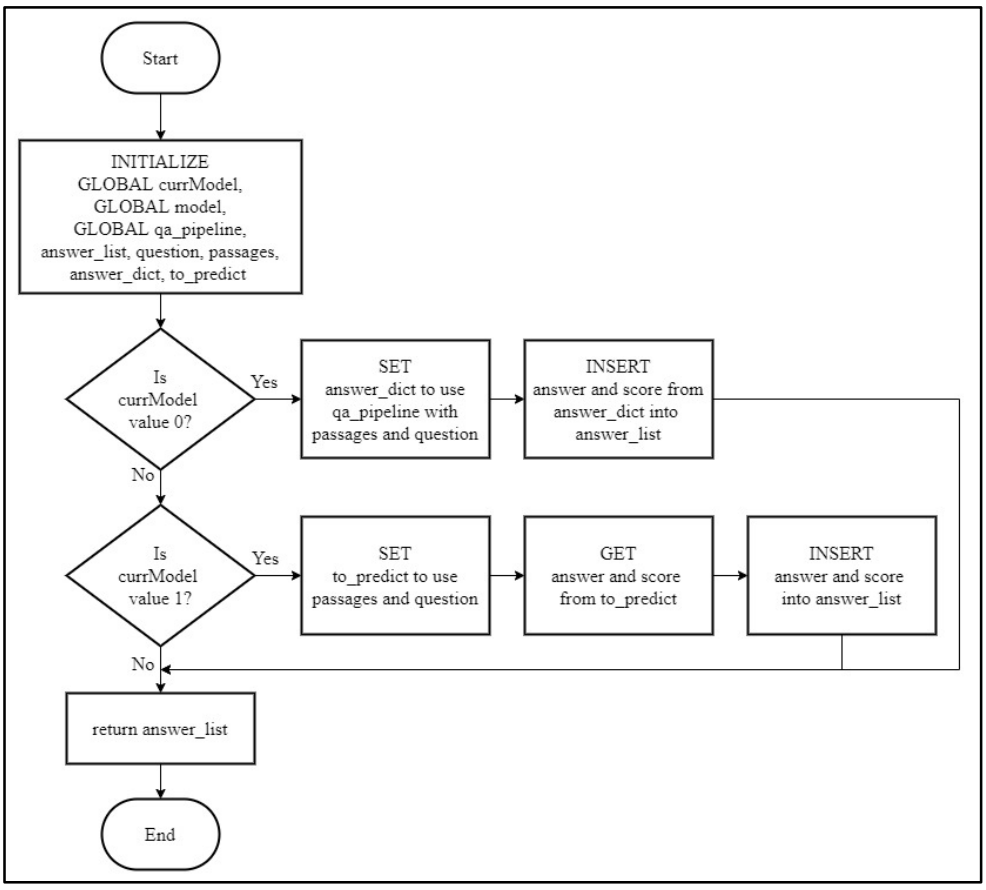


Gambar 3.8. *Flowchart* Change Model

B.3. Flowchart Predict Answer

Pada *resource* Predict Answer, dilakukan perubahan dengan menghilangkan penggunaan *pre* maupun *max tokens* yang digunakan *web service* pada modul Adaptive Learning. Penghilangan *tokens* dilakukan dikarenakan kedua *pre-trained* model yang digunakan pada penelitian ini tidak menggunakan *tokens* untuk melakukan prediksi jawaban. Variabel answer_list berupa *array* digunakan untuk menyimpan jawaban hasil prediksi dan nilai akurasi yang dihasilkan dari *pre-trained model* saat memberikan prediksi jawaban. Variabel question dan variabel passages yang digunakan memiliki fungsi yang sama dengan yang ada pada modul

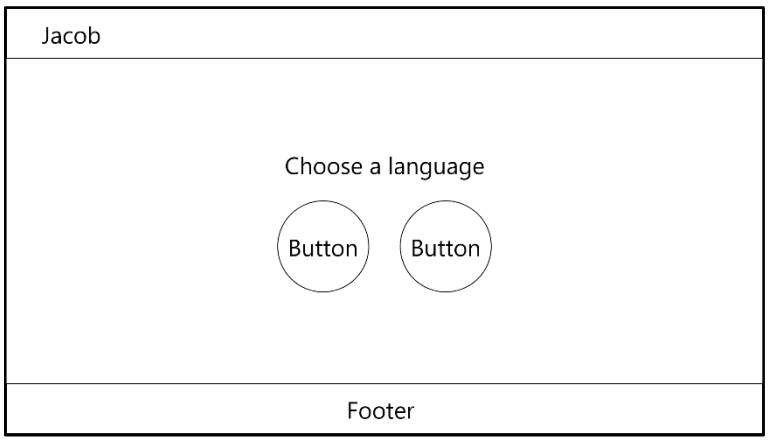
Adaptive Learning, yaitu merupakan pertanyaan *user* dan informasi hasil pencarian yang dilakukan. Kemudian terdapat variabel answer_dict yang digunakan oleh *pre-trained model* IndoBERT-lite untuk menyimpan jawaban dan nilai akurasi hasil prediksi. Sedangkan varibael to_predict digunakan oleh *pre-trained model* RoBERTa untuk menyimpan *question* dan *passages* dalam bentuk *array*, yang selanjutnya akan dilakukan prediksi jawaban. Variabel answers dan probabilities digunakan pada *pre-trained model* RoBERTa untuk menyimpan jawaban dan nilai akurasi hasil presiksi. Pada akhirnya *web service* akan mengirim jawaban hasil prediksi tersebut ke aplikasi *chatbot* Jacob. *Flowchart* mengenai *resource* Predict Answer dapat dilihat pada Gambar 3.9.



Gambar 3.9. *Flowchart* Predict Answer

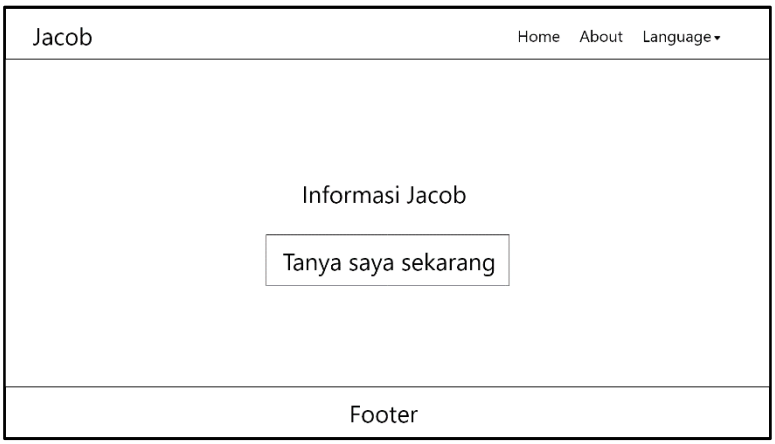
3.3.5 Rancangan Antarmuka Halaman

Pada subbab ini, dilakukan perancangan antarmuka halaman yang meliputi antramuka pemilihan bahasa dan antarmuka aplikasi *chatbot* Jacob Bahasa Indonesia. Pada antarmuka halaman pemilihan bahasa, terdapat dua tombol yang digunakan untuk memilih bahasa yang ingin digunakan pada aplikasi *chatbot* Jacob. Rancangan antarmuka pilihan bahasa dapat dilihat pada Gambar 3.10.



Gambar 3.10. Rancangan antarmuka halaman pilihan bahasa pada *chatbot* Jacob

Pada Gambar 3.11. menunjukkan antarmuka halaman aplikasi *chatbot* Jacob dalam Bahasa Indonesia. Pada bagian *navbar*, terdapat tombol untuk mengganti bahasa yang dapat digunakan saat telah memilih bahasa pada halaman awal.



Gambar 3.11. Rancangan antarmuka *chatbot* Jacob dalam Bahasa Indonesia