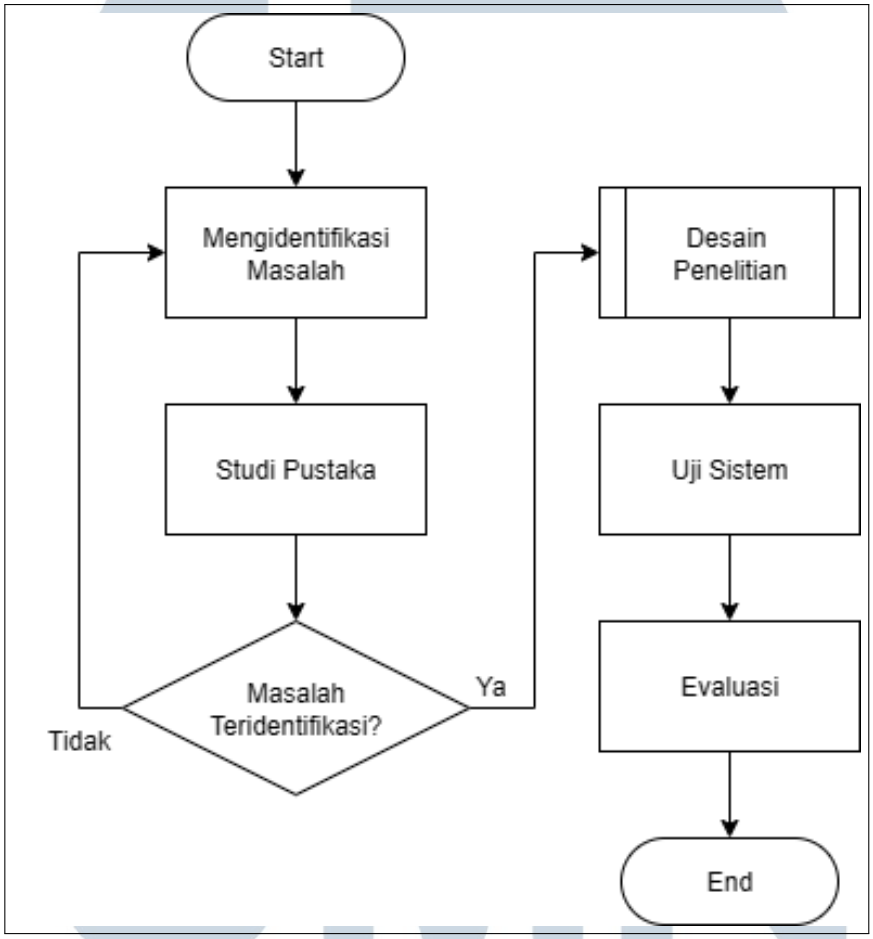
BAB 3
METODOLOGI PENELITIAN

Gambar 3.1 menunjukkan alur proses metodologi penelitian yang dilakukan dalam rancang bangun aplikasi ITS berbasis SAKT.



Gambar 3.1. Diagram alur proses metodologi penelitian

3.1 Identifikasi Masalah

Pada tahapan ini dilakukan identifikasi masalah. Identifikasi masalah dilakukan dari sumber yang ada internet yang berupa artikel, jurnal, dan sumber data lainnya. Pada tahun 2021 sebuah perusahaan *cybersecurity* dari Russia bernama Kaspersky melakukan sebuah survey yang ditujukan kepada murid-murid yang melakukan proses belajar mengajar secara online/daring pada masa pandemi COVID-19 [6]. Hasil dari survey tersebut mendapat bahwa dari semua partisipan

survey, 48% dari mereka memilih matematika sebagai mata pelajaran yang paling sulit untuk dipelajari secara online.Selain itu, berdasarkan data dari National Center for Education (NCES) pada tahun 2024 yang melakukan assesmen dari kemampuan matematika dari pelajar kelas 8 di Amerika Serikat, ditemukan bahwa 28% dari pelajar mahir dalam mata pelajaran matematika, tetapi 39% dari pelajar memiliki kemampuan matematika yang kurang [7].

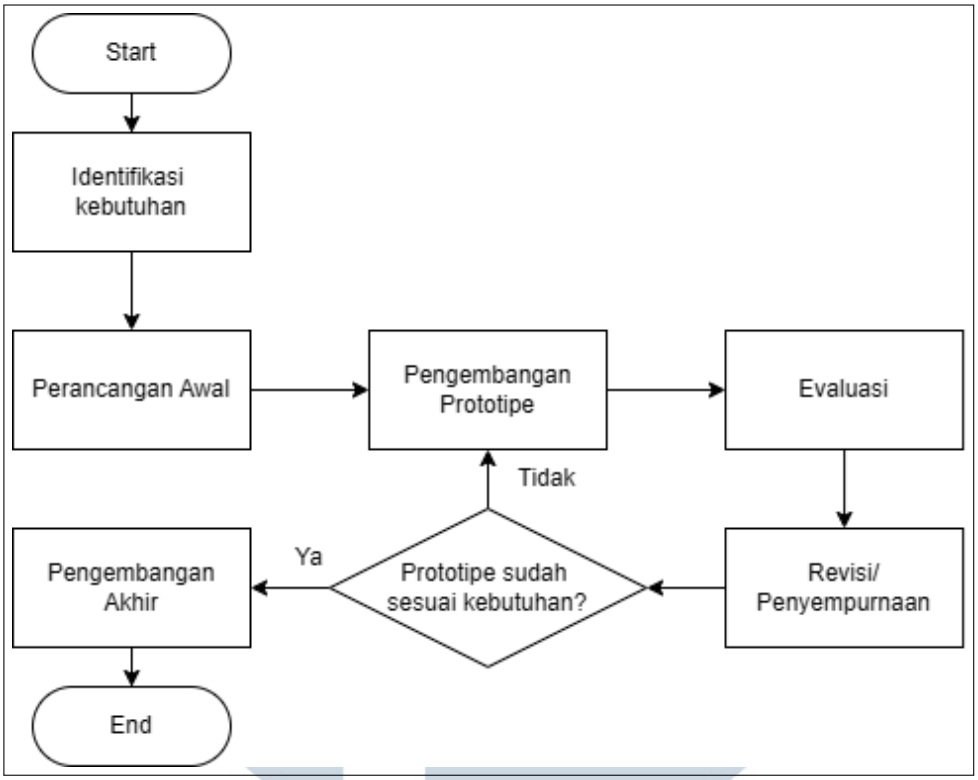
Berdasarkan informasi yang didapat dari kedua sumber yang disebutkan, masalah yang teridentifikasi adalah ditemukannya kesenjangan pemahaman pada pelajar sekolah. Kesenjangan ini muncul secara signifikan dalam mata pelajaran matematika. Oleh karena itu, diperlukan sistem pembelajaran yang dapat membantu proses belajar yang dapat meningkatkan performa akademik pelajar.

3.2 Studi Pustaka

Pada tahapan ini dilakukan studi pustaka. Studi ini dilakukan untuk mengumpulkan dan mempelajari literatur yang relevan. Literatur yang relevan dalam penelitian ini adalah literatur mengenai Intelligent Tutoring System (ITS), Self Attentive Model for Knowledge Tracing (SAKT), dan pembuatan aplikasi berbasis *web* menggunakan Next.js dan React. Selain itu, literatur mengenai metode *Prototyping* dalam melakukan desain penelitian dan literatur mengenai metode *Structured Analysis* dan *Structured Design* (SA/SD) dalam melakukan desain sistem. Literatur-literatur tersebut didapat dari berbagai artikel, jurnal, dan penelitian sebelumnya pada topik-topik yang relevan dalam penelitian ini.

3.3 Desain Penelitian

Tahap ini dilakukan dengan merancang arsitektur aplikasi, UI/UX, dan membuat model SAKT menggunakan *dataset* ASSISTment 2015 berdasarkan studi literatur. Proses desain dapat dilihat pada Gambar 3.2.



Gambar 3.2. Diagram metodologi *prototyping*

Dalam penelitian ini, model pengembangan yang digunakan adalah model *prototyping*, dengan fokus pada pembuatan *proof of concept* sistem Intelligent Tutoring System (ITS) berbasis model *Self-Attentive Knowledge Tracing* (SAKT). Pemilihan model ini dikarenakan pendekatan *prototyping* memungkinkan pengembangan sistem secara bertahap dan fleksibel, sesuai dengan kebutuhan eksploratif dan eksperimental dari penelitian ini. Evaluasi berfokus pada validasi alur kerja sistem, integrasi model SAKT, dan kemampuan sistem memberikan prediksi serta umpan balik secara logis berdasarkan data yang diterima dari pelajar. Aplikasi yang dihasilkan dari penelitian ini diharapkan dapat menunjukkan potensi penerapan pendekatan SAKT dalam pengembangan ITS, terutama pada pembelajaran matematika.

3.4 Identifikasi Kebutuhan Sistem

Proses identifikasi kebutuhan sistem dilakukan untuk merumuskan permasalahan yang dihadapi oleh pelajar serta menentukan fitur dan fungsi utama yang dibutuhkan dalam pengembangan aplikasi Intelligent Tutoring System (ITS). Berdasarkan hasil studi literatur dan survei yang dilakukan oleh Kaspersky

pada masa pandemi COVID-19, ditemukan bahwa matematika merupakan mata pelajaran yang dianggap paling sulit untuk dipelajari secara daring [6]. Selain itu, data dari National Center for Education Statistics (NCES) tahun 2024 menunjukkan adanya kesenjangan signifikan dalam kemampuan matematika pelajar kelas 8, di mana sebagian besar pelajar masih menunjukkan performa belajar yang rendah [7].

Permasalahan utama yang diidentifikasi adalah rendahnya efektivitas pembelajaran daring dalam membantu pelajar memahami materi matematika secara personal dan adaptif. Dalam konteks ini, pelajar membutuhkan sistem pembelajaran yang mampu memberikan soal secara bertahap sesuai dengan tingkat kemampuan mereka serta dapat menyesuaikan materi berdasarkan performa individu.

Berdasarkan identifikasi masalah tersebut, tujuan utama dari pengembangan sistem ini adalah merancang dan membangun sebuah aplikasi ITS berbasis model *Self-Attentive Knowledge Tracing* (SAKT) yang mampu memprediksi performa pelajar dan memberikan soal lanjutan secara adaptif. Aplikasi ini diharapkan dapat menjadi proof of concept yang menunjukkan potensi penerapan model SAKT dalam mendukung proses belajar matematika secara mandiri dan berbasis teknologi, dengan fitur utama berupa pelacakan progres, adaptasi kesulitan soal, serta penyimpanan riwayat belajar secara sistematis.

3.5 Perancangan Awal

Tahap kedua dalam proses penelitian ini adalah perancangan awal dari sistem aplikasi *Intelligent Tutoring System* (ITS) untuk mata pelajaran matematika. Perancangan sistem bertujuan untuk menggambarkan secara rinci struktur, alur kerja, dan antarmuka dari aplikasi sebelum dikembangkan dalam bentuk implementasi nyata.

Dalam merancang sistem *Intelligent Tutoring System* (ITS) berbasis model *Self-Attentive Knowledge Tracing* (SAKT), pendekatan *Structured Analysis* dan *Structured Design* (SA/SD) digunakan dalam tahapan desain sistem. Pendekatan ini dipilih karena mampu memisahkan secara jelas antara aspek analisis fungsional sistem dengan aspek desain teknis, sehingga pengembangan sistem dapat dilakukan secara bertahap, terstruktur, dan mudah dipahami.

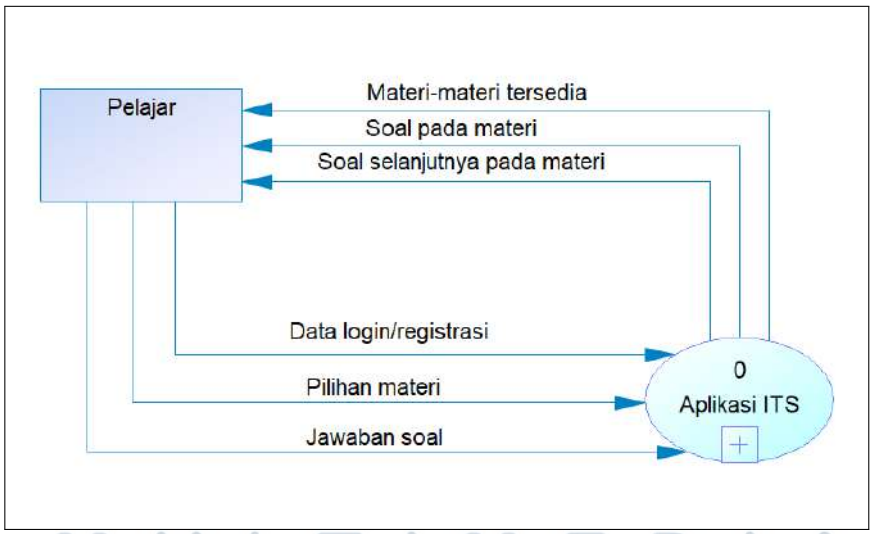
Metode SA/SD dianggap lebih sesuai dibandingkan pendekatan *Object-Oriented Design* (OOD) pada penelitian ini karena fokus pengembangan ITS ini lebih menekankan pada alur data, proses-proses utama, interaksi data antar proses, dan hanya terfokus pada sisi pelajar, bukan pada pemodelan objek dan pewarisan

kelas seperti dalam OOD. Selain itu, sistem ini bersifat prosedural dan terdiri dari rangkaian proses yang jelas, sehingga pendekatan terstruktur lebih tepat untuk menggambarkan alur pengerjaan soal, proses login, registrasi, serta interaksi dengan model *machine learning* SAKT.

Structured Analysis digunakan untuk memodelkan sistem dalam bentuk hierarki proses dan aliran data menggunakan *Data Flow Diagram* (DFD). Sementara itu, *Structured Design* digunakan untuk menerjemahkan hasil analisis menjadi perancangan sistem melalui *Entity Relationship Diagram* (ERD), dan *Flowchart* proses.

3.5.1 *Data Flow Diagram* (DFD)

Proses Structured Design dilakukan dengan membuat *Data Flow Diagram* atau DFD. DFD dibuat sebagai gambaran aliran data pada suatu sistem dengan penggunanya. DFD yang terdiri dari dua level, yaitu DFD Level 0 dan DFD Level 1. *Data Flow Diagram* Level 0 dibuat sebagai gambaran dasar dari keseluruhan sistem dan interaksinya dengan entitas eksternal. Pada penelitian ini, hanya ada 1 entitas eksternal yang berinteraksi dengan sistem, yaitu pelajar.

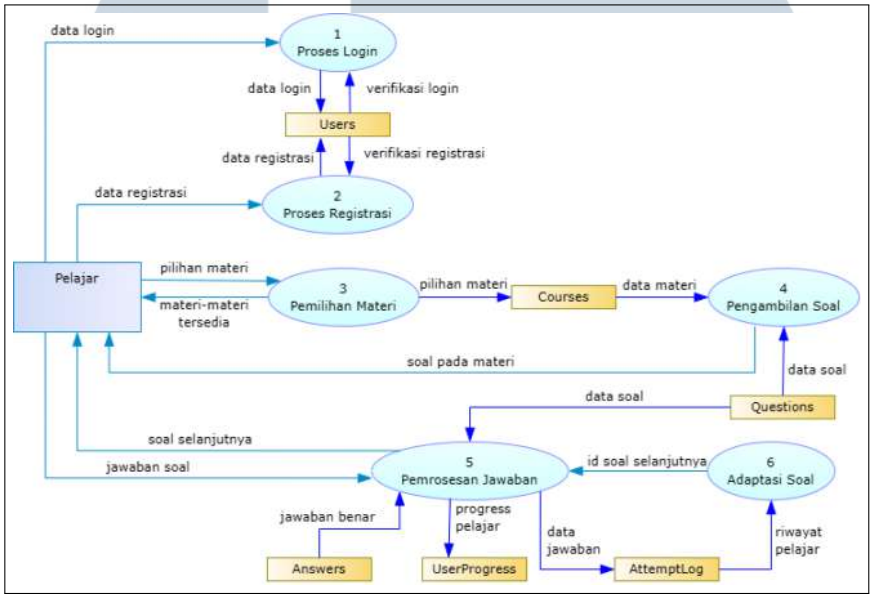


Gambar 3.3. *Data Flow Diagram* Level 0 ITS

Data Flow Diagram Level 0 ditampilkan pada Gambar 3.3. Pada diagram tersebut, digambarkan interaksi antara sistem dengan pelajar melalui pengiriman dan penerimaan data. Pelajar dapat mengirimkan data *login* atau registrasi, pilihan materi, dan jawaban soal. Sedangkan, aplikasi ITS dapat mengirimkan daftar materi

yang tersedia dan soal berdasarkan amteri yang dipilih oleh pelajar.

Data Flow Diagram Level 1 dibuat untuk menguraikan sistem menjadi beberapa proses utama yang saling terhubung melalui aliran data. Dalam penelitian ini, proses-proses tersebut meliputi otentikasi pengguna (*login*/registrasi), pemilihan materi, pengerjaan soal, dan pengiriman data ke model *machine learning* untuk mendapatkan prediksi kinerja pelajar.



Gambar 3.4. *Data Flow Diagram Level 1* aplikasi ITS

Data Flow Diagram Level 1 ditampilkan pada Gambar 3.4. Pada diagram tersebut, ditunjukkan proses-proses utama yang diimplementasikan pada sistem ITS. Selain itu, digambarkan juga alur perpindahan data antara penyimpan data, proses sistem, dan pelajar. Proses-proses utama yang ada pada sistem berdasarkan DFD Level 1 yang sudah dibuat adalah proses autentikasi berupa *login* dan registrasi akun, pemilihan materi, pengambilan soal, pemrosesan jawaban, dan proses adaptasi soal. Selain itu, terdapat beberapa penyimpan data/*data store* yang perlu dibuat, yaitu *Users* untuk menyimpan data akun, *Courses* untuk menyimpan data soal, *Questions* untuk menyimpan data pertanyaan, *Answers* untuk menyimpan data jawaban, *AttemptLog* untuk menyimpan data riwayat pelajar, dan *UserProgress* untuk menyimpan kemajuan/*progress* dari pelajar pada materi.

Terdapat 4 alur perpindahan data yang digambarkan pada DFD Level 1, yaitu:

- Alur *login*

  Alur ini dimulai dengan pelajar mengirimkan data *login* berupa *email*

dan *password* ke aplikasi. Data ini diproses pada proses *login* dengan membandingkan data yang diterima dengan data yang disimpan pada *data store Users*. Verifikasi *login* kemudian dikirimkan ke sistem dan pelajar diarahkan ke halaman dasbor.

- Alur registrasi
  Alur ini dimulai dengan pelajar mengirimkan data registrasi berupa *email* dan *password* ke aplikasi. Data ini diproses pada proses registrasi dengan menyimpan data yang diterima pada *data store Users*. Verifikasi registrasi kemudian dikirimkan ke sistem dan pelajar diarahkan ke halaman dasbor.

- Alur pemilihan materi
  Alur ini dimulai dengan pelajar mengirimkan pilihan materi ke sistem. Sistem meneruskan pilihan materi ke *data store Courses* untuk mendapatkan data materi yang dipilih. Setelah itu, data materi dikirimkan kembali ke sistem pada proses pengambilan soal. Berdasarkan data materi yang didapat, proses tersebut mengambil data-data soal yang ada pada materi dari *data store Questions* dan sistem menampilkannya kepada pelajar.

- Alur pengerjaan soal
  Alur ini dimulai dengan pelajar mengirimkan jawaban untuk sebuah soal kepada sistem. Sistem memroses jawaban yang didapat dengan membandingkan jawaban yang diterima dengan jawaban benar yang disimpan pada *data store Answers*. Setelah itu, sistem menyimpan data percobaan menjawab ke *data store AttemptLog*. Sistem melanjutkan alur dengan melakukan adaptasi soal. Proses adaptasi ini mengambil riwayat menjawab pelajar dari *data store AttemptLog*. Setelah itu, sistem menyimpan data *progress* pelajar ke *data store User Progress*. Data *progress* tersebut dikembalikan ke sistem pada proses pengambilan soal dan soal selanjutnya ditampilkan ke pelajar.

3.5.2 Struktur *database*

Database dirancang untuk menyimpan berbagai data penting yang dibutuhkan oleh aplikasi, seperti data pelajar, data soal dan keterampilan/*skills*, jawaban siswa, hasil prediksi dari model SAKT, serta data log aktivitas siswa. Pembangunan *database* dilakukan berdasarkan Entity Relationship Diagram (ERD) pada Gambar 3.5. Dalam implementasinya, sistem menggunakan Microsoft SQL

Server (MSSQL) sebagai sistem manajemen basis data dan Sequelize sebagai Object-Relational Mapping (ORM) untuk mengelola interaksi antara aplikasi dan *database*. Sequelize memungkinkan untuk memodelkan tabel *database* MSSQL dalam bentuk objek JavaScript, sehingga mempermudah proses pembuatan, pembacaan, pembaruan, dan penghapusan data (CRUD) tanpa harus menulis perintah *query* SQL secara langsung. Tabel-tabel yang dibuat pada *database* ITS untuk penelitian ini adalah:

1. Tabel *user*

   Tabel ini menyimpan data pelajar/*user*, seperti nama pelajar, email pelajar, dan kata sandi akun pelajar. Desain dari tabel *user* serta deskripsi dari setiap kolomnya dapat dilihat pada Tabel 3.1.

   Tabel 3.1. Desain tabel *user*

Nama Kolom	**Tipe Data**	**Deskripsi**
id (PK)	CHAR(36)	Teks sepanjang 36 karakter yang di-*generate* secara acak sebagai tanda pengenal unik *user* pada sistem
name	NVARCHAR(255)	Nama akun pelajar
email	NVARCHAR(255)	Email yang didaftarkan untuk akun pelajar
password	NVARCHAR(255)	Kata sandi akun pelajar
createdAt	DATETIMEOFFSET(7)	Tanggal pembuatan akun
updatedAt	DATETIMEOFFSET(7)	Tanggal terakhir akun diperbaharui

2. Tabel *courses*

   Tabel ini menyimpan informasi mengenai materi-materi yang tersedia dalam aplikasi ITS, seperti nama materi dan deskripsi dari materi. Desain dari tabel *course* serta deskripsi dari setiap kolomnya dapat dilihat pada Tabel 3.2.

Tabel 3.2. Desain tabel *courses*

Nama Kolom	**Tipe Data**	**Deskripsi**
id (PK)	CHAR(36)	Teks sepanjang 36 karakter yang di-*generate* secara acak sebagai tanda pengenal unik materi pada sistem
title	NVARCHAR(255)	Judul *course*
description	NVARCHAR(255)	Deskripsi *course*
createdAt	DATETIMEOFFSET(7)	Tanggal pembuatan *course*
updatedAt	DATETIMEOFFSET(7)	Tanggal terakhir *course* diperbaharui

3. Tabel *questions*

   Tabel ini menyimpan informasi mengenai soal-soal yang ditampilkan di dalam sebuah materi, seperti teks pertanyaan, persamaan matematika yang harus diselesaikan, dan materi yang bersangkutan. Desain dari tabel *questions* serta deskripsi dari setiap kolomnya dapat dilihat pada Tabel 3.3 dan dilanjutkan pada Tabel 3.4.

Tabel 3.3. Desain tabel *questions*

Nama Kolom	**Tipe Data**	**Deskripsi**
id (PK)	CHAR(36)	Teks sepanjang 36 karakter yang di-*generate* secara acak sebagai tanda pengenal unik soal pada sistem
questionText	NVARCHAR(255)	Teks pertanyaan
equation	NVARCHAR(255)	Persamaan matematika yang perlu diselesaikan
order	INT	Posisi pertanyaan dalam materi

Tabel 3.4. Desain tabel *questions* (lanjutan)

Nama Kolom	Tipe Data	Deskripsi
courseId	CHAR(36)	*Id course*/materi yang memuat pertanyaan
skillId	INT	*Id skill* sebagai tanda pengenal unik
createdAt	DATETIMEOFFSET(7)	Tanggal pembuatan soal
updatedAt	DATETIMEOFFSET(7)	Tanggal terakhir soal diperbaharui

4. Tabel *answers*

   Tabel ini menyimpan informasi mengenai jawaban-jawaban dari soal, seperti teks jawaban yang benar dan soal yang bersangkutan. Desain dari tabel *answers* serta deskripsi dari setiap kolomnya dapat dilihat pada Tabel 3.5.

Tabel 3.5. Desain tabel *answers*

Nama Kolom	Tipe Data	Deskripsi
id (PK)	CHAR(36)	Teks sepanjang 36 karakter yang di-*generate* secara acak sebagai tanda pengenal unik jawaban pada sistem
questionId	CHAR(36)	*Id question* yang berkaitan dengan jawaban
answerText	NVARCHAR(255)	Teks jawaban
isCorrect	BOOLEAN	Teks jawaban yang disimpan benar atau tidak
createdAt	DATETIMEOFFSET(7)	Tanggal pembuatan jawaban
updatedAt	DATETIMEOFFSET(7)	Tanggal terakhir jawaban diperbaharui

5. Tabel *attemptLog*

   Tabel ini menyimpan riwayat percobaan menjawab seorang pelajar pada sebuah soal. Riwayat ini terdiri dari *id user*/pelajar, *id* soal yang dikerjakan, serta kebenaran dari percobaan menjawab. Desain dari tabel *attemptLog* serta deskripsi dari setiap kolomnya dapat dilihat pada Tabel 3.6.

Tabel 3.6. Desain tabel *attemptLog*

Nama Kolom	**Tipe Data**	**Deskripsi**
id (PK)	CHAR(36)	Teks sepanjang 36 karakter yang di-*generate* secara acak sebagai tanda pengenal unik riwayat pada sistem
userId	CHAR(36)	*Id user* yang melakukan percobaan menjawab
questionId	CHAR(36)	*Id question* yang dijawab
correct	INT	Percobaan menjawab benar atau tidak
createdAt	DATETIMEOFFSET(7)	Tanggal percobaan menjawab
updatedAt	DATETIMEOFFSET(7)	Tanggal percobaan menjawab diperbaharui

6. Tabel *userProgress*

   Tabel ini menyimpan *progress*/kemajuan seorang pelajar dalam materi. Data yang disimpan pada tabel ini adalah *id user*/pelajar, *id* materi, *progress*/kemajuan pelajar pada materi dalam bentuk angka, dan *id* dari soal yang terakhir kali ditampilkan kepada pelajar pada materi. Desain dari tabel *userProgress* serta deskripsi dari setiap kolomnya dapat dilihat pada Tabel 3.7.

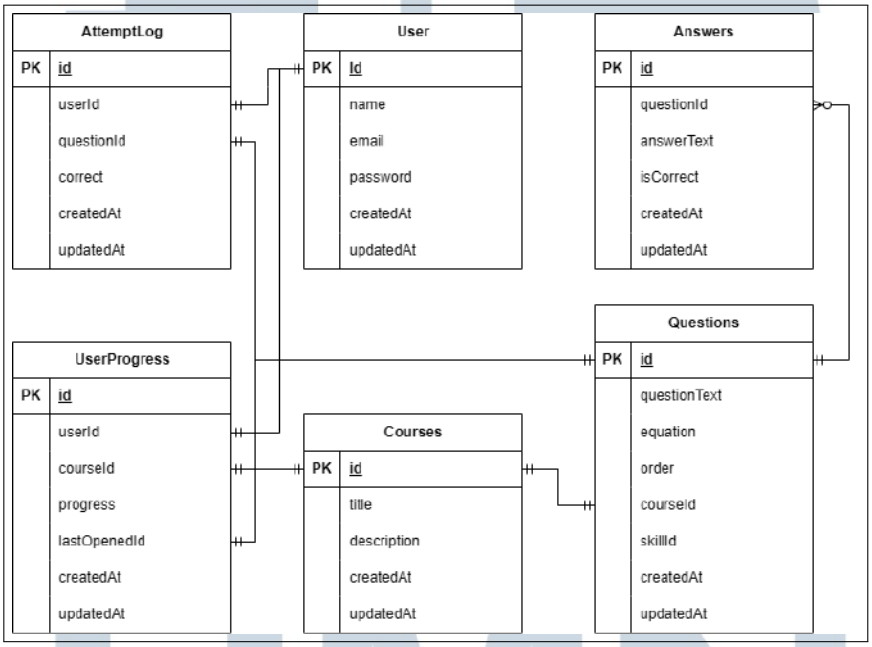
Tabel 3.7. Desain tabel *userProgress*

Nama Kolom	**Tipe Data**	**Deskripsi**
id (PK)	CHAR(36)	Teks sepanjang 36 karakter yang di-*generate* secara acak sebagai tanda pengenal unik pada sistem
userId	CHAR(36)	*Id user*
courseId	CHAR(36)	*Id course* yang dipelajari oleh *user*/pelajar
progress	INT	Kemajuan *user* dalam *course*
lastOpenedId	NVARCHAR(255)	*Id question* yang terakhir dibuka
createdAt	DATETIMEOFFSET(7)	Tanggal pembuatan *userProgress*
updatedAt	DATETIMEOFFSET(7)	Tanggal terakhir *userProgress* diperbaharui

Untuk berinteraksi dengan *database*, aplikasi menggunakan RESTful API. RESTful API adalah jenis antarmuka pemrograman aplikasi (API) yang menggunakan prinsip arsitektur *Representational State Transfer* (REST) untuk memungkinkan komunikasi antara klien dan server melalui protokol HTTP [24]. REST API memanfaatkan metode HTTP standar seperti GET, POST, PUT, dan DELETE untuk mengakses dan memanipulasi sumber daya yang diidentifikasi dengan URL unik (*endpoint*). Data yang dipertukarkan biasanya dalam format yang ringan dan mudah diproses seperti JSON atau XML. Karena sifatnya yang fleksibel, ringan, dan *stateless* (tidak menyimpan status klien di server), RESTful API digunakan untuk mengintegrasikan aplikasi dan layanan dalam berbagai *platform* secara efisien.

3.5.3 *Entity Relationship Diagram* (ERD)

Penyimpanan data yang ditunjukkan pada DFD Level 1 dimodelkan pada sebuah *Entity Relationship Diagram* (ERD). ERD digunakan untuk memodelkan struktur basis data yang mendukung sistem ITS. ERD menggambarkan tabel-tabel data yang digunakan dalam sistem, atribut dari setiap tabel, serta relasi antar tabel-tabel tersebut. ERD membantu dalam membuat struktur tabel *database* dan menyusun kebutuhan data yang digunakan oleh sistem.



Gambar 3.5. Entity Relationship Diagram *database* ITS

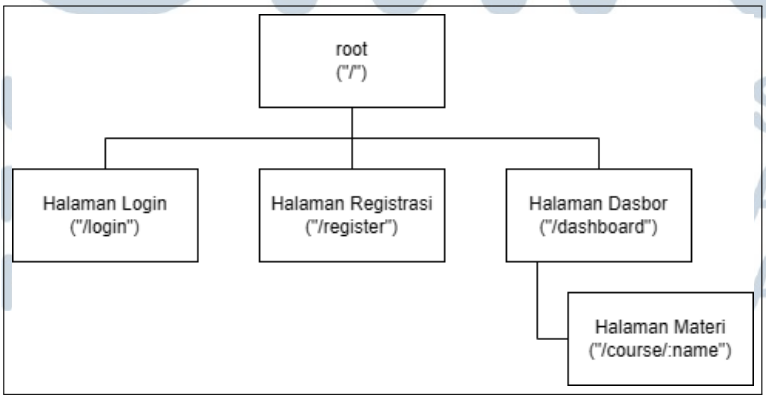
Entity Relationship Diagram (ERD) dari *database* aplikasi dapat dilihat pada Gambar 3.5. Berdasarkan ERD, terdapat 6 tabel yang perlu dibuat pada sistem, yaitu *User*, *Courses*, *Questions*, *Answers*, *AttemptLog*, dan *UserProgress*. Dari tabel-tabel tersebut, relasi antara mereka adalah:

- Tabel *User* dan *AttemptLog* memiliki relasi *one-to-many* dimana satu pelajar dapat memiliki data *attempt log* lebih dari satu. Relasi ini merepresentasikan riwayat percobaan menjawab seorang pelajar pada soal yang berbeda-beda.
- Tabel *User* dan *UserProgress* memiliki relasi *one-to-many* dimana satu pelajar dapat mempunyai lebih dari satu data *progress*. Relasi ini merepresentasikan *progress* pelajar pada berbagai materi yang tersedia pada ITS.

- Tabel *Courses* dan *UserProgress* memiliki relasi *one-to-many* dimana satu materi dapat muncul pada beberapa data *progress*. Relasi ini merepresentasikan satu materi yang dapat diakses oleh beberapa pelajar.
- Tabel *Courses* dan *Questions* memiliki relasi *one-to-many* dimana satu materi memiliki banyak soal.
- Tabel *Questions* dan *Answers* memiliki relasi *one-to-many* dimana satu soal dapat memiliki beberapa jawaban. Relasi ini dibuat untuk mengakomodasi adanya soal dengan pilihan ganda.
- Tabel *Questions* dan *AttemptLog* memiliki relasi *one-to-many* dimana satu soal dapat muncul pada lebih dari satu data riwayat. Relasi ini merepresentasikan bahwa setiap soal dapat dikerjakan berulang kali oleh berbagai pelajar.

3.5.4 Rancangan *sitemap*/peta situs

Sitemap atau peta situs merupakan representasi struktur navigasi dari aplikasi yang dikembangkan. Pada penelitian ini, sitemap disusun untuk menggambarkan alur halaman yang dapat diakses oleh pelajar dalam aplikasi *Intelligent Tutoring System* (ITS). *Sitemap* membantu dalam merancang antarmuka pengguna dengan menunjukkan hubungan hierarkis antarhalaman serta bagaimana pengguna berpindah dari satu halaman ke halaman lainnya. Karena aplikasi ini berfokus pada sisi siswa, maka sitemap yang disusun hanya mencakup halaman-halaman yang relevan bagi siswa, seperti halaman *login*, registrasi, dasbor pemilihan materi, dan halaman materi.

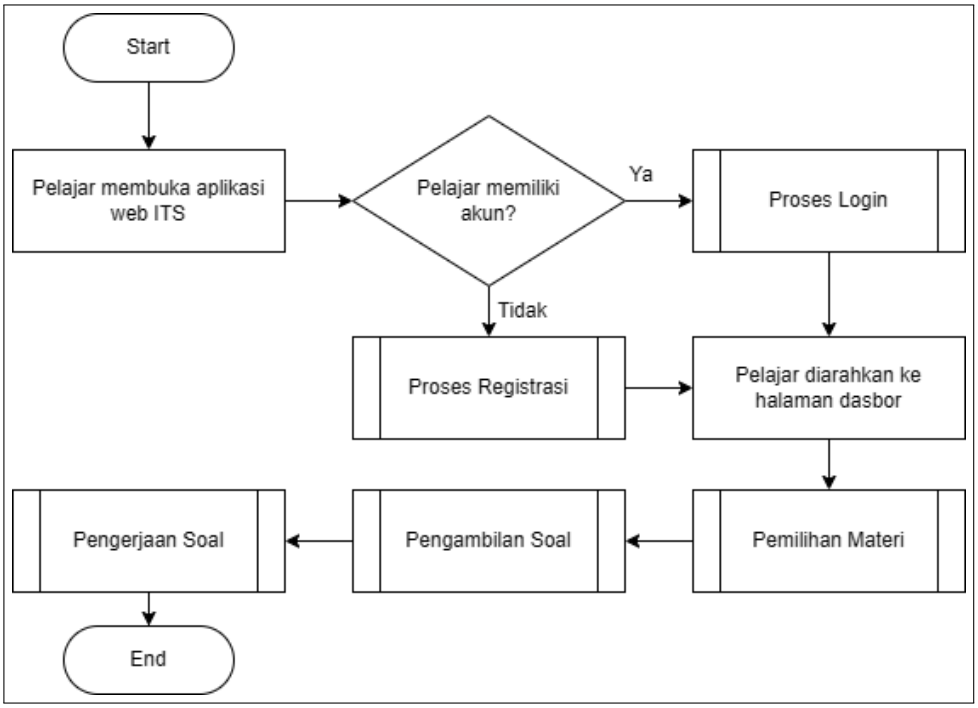


Gambar 3.6. *Sitemap* aplikasi ITS

Sitemap yang dibuat untuk pengembangan aplikasi ITS ini dapat dilihat pada Gambar 3.6. Pada aplikasi ITS ini, jalur URL *root* (”/”) tidak merepresentasikan sebuah halaman beranda. Sebaliknya, jalur tersebut berfungsi sebagai mekanisme pengalihan */redirect* yang mengarahkan pelajar yang belum terautentikasi ke halaman *login* (”/login”), dan pengguna yang telah terautentikasi ke halaman dasbor (”/dashboard”). Selain itu, terdapat juga halaman registrasi (”/register”) yang dapat diakses secara independen seperti halaman *login* dan halaman dasbor. Terdapat juga halaman materi (”/course/:name”) yang merupakan *child node* dari halaman dasbor karena halaman materi hanya dapat diakses dari halaman dasbor.

3.5.5 *Flowchart sistem*

Flowchart sistem digunakan untuk menggambarkan alur kerja aplikasi ITS, dimulai dari proses autentikasi, pemilihan materi, pengerjaan soal, dan proses adaptasi kesulitan soal menggunakan model SAKT. *Flowchart* dirancang berdasarkan DFD Level 1 yang dapat dilihat pada Gambar 3.4. *Flowchart* dibuat untuk memberikan visualisasi alur sistem secara jelas untuk memudahkan proses implementasi.



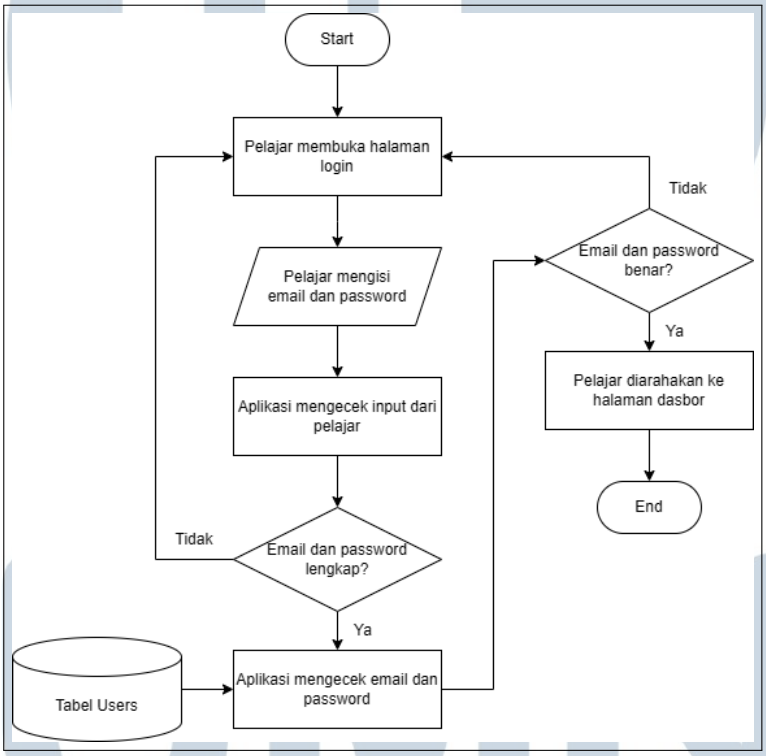
Gambar 3.7. *Flowchart* sistem aplikasi ITS

Gambar 3.7 menunjukkan *flowchart* yang dibuat untuk menggambarkan alur

kerja sistem aplikasi ITS. Alur sistem dapat dibagi menjadi beberapa bagian utama, yaitu:

1. Proses autentikasi

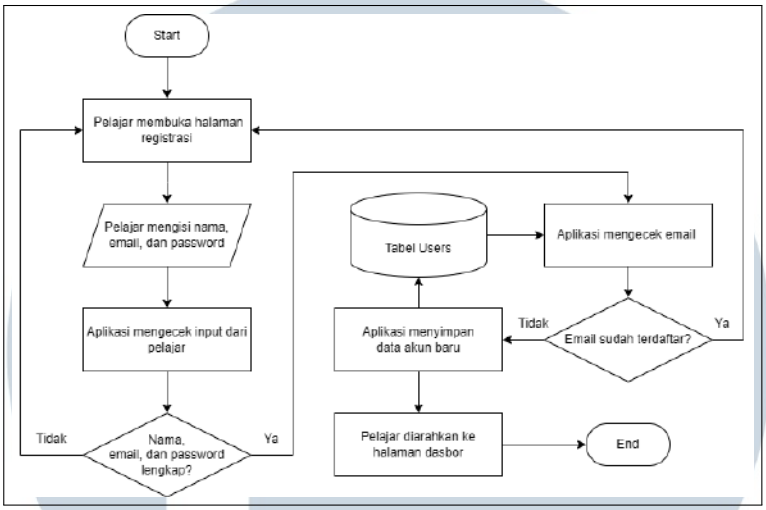
   Autentikasi adalah verifikasi identitas pengguna sebelum diberikan akses ke sistem, layanan, atau informasi tertentu. Pada penelitian ini, proses autentikasi dilakukan untuk memastikan bahwa hanya pelajar terdaftar yang dapat mengkases aplikasi ITS. Proses autentikasi pada aplikasi ITS terbagi menjadi dua proses, yaitu proses *login* dan proses registrasi.



Gambar 3.8. *Flowchart* proses *login* aplikasi ITS

Flowchart dari proses *login* dapat dilihat pada Gambar 3.8. Ketika pelajar membuka halaman *web* aplikasi ITS, aplikasi menampilkan halaman *login* dimana pelajar dapat masuk ke akun mereka yang sudah didaftarkan sebelumnya. Pada halaman *login*, tersedia kolom-kolom *input* dimana pelajar dapat memasukkan *email* dan *password* sesuai dengan akun yang terdaftar pada aplikasi. Pelajar dapat masuk ke akun mereka dengan mengirimkan *email* dan *password* kepada aplikasi. Setelah itu, aplikasi melakukan cek autentikasi dengan mencari kombinasi *email* dan *password* ke tabel *Users* untuk melakukan verifikasi apakah kombinasi *email* dan *password* yang

diterima benar. Jika benar, pelajar akan diarahkan ke halaman dasbor. Jika tidak, pelajar akan diminta untuk memasukkan *email* dan *password* kembali.

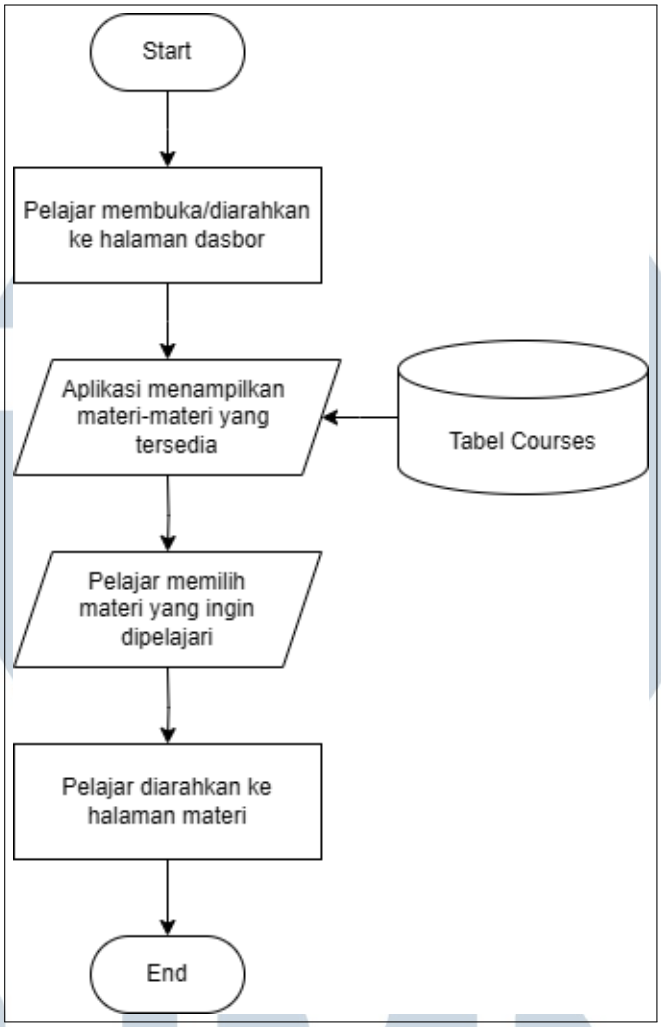


Gambar 3.9. *Flowchart* proses registrasi aplikasi ITS

Flowchart dari proses registrasi dapat dilihat pada Gambar 3.9. Apabila pelajar tidak dapat melakukan *login* ke aplikasi karena belum memiliki akun yang terdaftar pada aplikasi, pelajar dapat membuka halaman registrasi akun. Pada halaman tersebut, pelajar dapat mendaftarkan akun baru dengan memasukkan nama, *email*, dan *password* pada kolom-kolom *input* yang tersedia. Jika ketiga data tersebut berhasil diterima oleh aplikasi, maka aplikasi melakukan cek ke tabel *User* untuk memastikan bahwa *email* yang diterima belum digunakan untuk mendaftarkan akun sebelumnya dan mencegah adanya *email* yang terdaftar untuk lebih dari satu akun/duplikat. Jika data lengkap dan *email* tersedia, maka data akun baru akan disimpan di dalam tabel *User* dan pelajar akan secara langsung diarahkan ke halaman dasbor dengan akun yang baru didaftarkan tersebut. Jika ada data yang kurang atau tidak sesuai, maka pelajar akan diminta untuk mengisi nama, *email*, dan *password* kembali.

2. Pemilihan materi

   Proses pemilihan materi dimulai dengan pelajar membuka halaman dasbor setelah berhasil melakukan *login* atau registrasi akun baru.

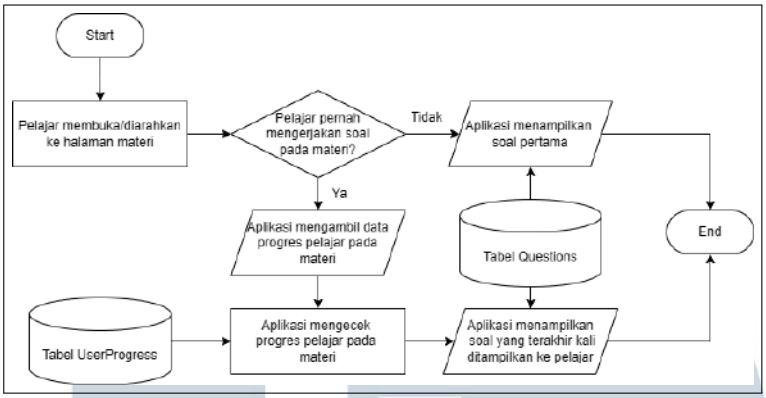


Gambar 3.10. *Flowchart* pemilihan materi aplikasi ITS

Flowchart dari proses pemilihan materi yang ada pada halaman dasbor dapat dilihat pada Gambar 3.10. Ketika halaman dasbor dibuka, aplikasi ITS mengambil semua data materi yang disimpan pada tabel *Courses*. Setelah itu, aplikasi ITS menampilkan semua materi yang tersedia berdasarkan data yang diambil dari tabel *Courses* dalam bentuk daftar dan pelajar dapat memilih materi yang ingin mereka pelajari. Setelah pelajar memilih salah satu materi dari daftar materi yang tersedia, pelajar diarahkan ke halaman materi yang mereka pilih.

3. Pengambilan soal

   Setelah pelajar diarahkan ke halaman materi, aplikasi memroses pengambilan soal yang ada pada materi sesuai dengan ada atau tidaknya data *progress* pelajar pada materi yang mereka pilih.

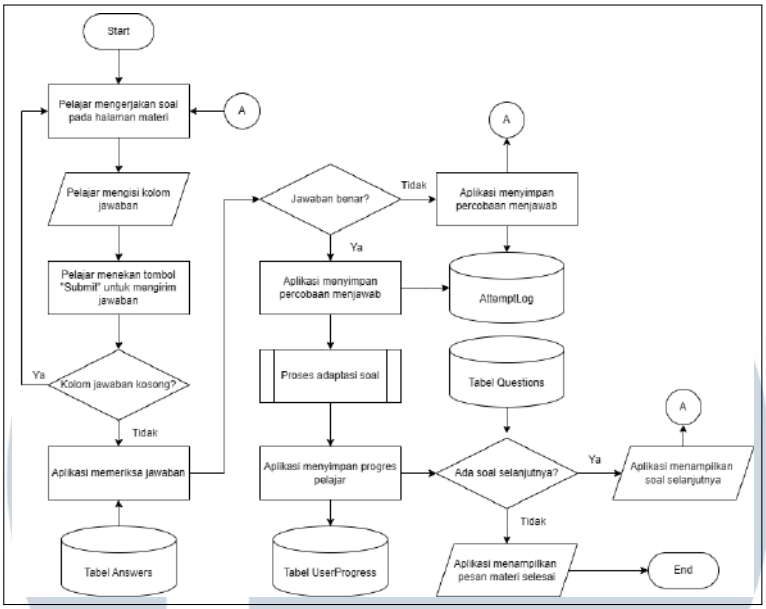


Gambar 3.11. *Flowchart* proses pengambilan soal pada aplikasi ITS

Flowchart pada Gambar 3.11 menggambarkan proses pengambilan soal pada halaman materi. Ketika halaman suatu materi dibuka oleh pelajar seletah melakukan pemilihan, aplikasi akan mengecek apabila pelajar sudah pernah mengerjakan soal pada materi tersebut dengan mencari data *progress* yang sesuai pada tabel *UserProgress* menggunakan data akun pelajar dan data materi yang dipilih. Data *progress* yang disimpan pada tabel *UserProgress* menyimpan informasi mengenai sejauh apa pelajar dalam materi yang dipilih, seperti soal yang terakhir ditampilkan kepada pelajar dan sedalam apa pelajar dalam menyelesaikan materi. Keberadaan data *progress* tersebut mempengaruhi soal yang ditampilkan pada halaman materi. Jika data *progress* ada dan berhasil didapatkan oleh aplikasi, aplikasi akan mengambil soal terakhir yang ditampilkan kepada pelajar pada materi dari tabel *Questions* dan menampilkan soal tersebut. Jika data *progress* tidak ada karena pelajar baru pertama kali membuka materi yang dipilih, aplikasi akan mengambil dan menampilkan soal pertama yang ada pada materi dari tabel *Questions*.

4. Menjawab soal

   Pada halaman materi, pelajar dapat mengerjakan soal yang ditampilkan oleh aplikasi dan memasukkan jawaban yang mereka dapat pada kolom jawaban yang tersedia.

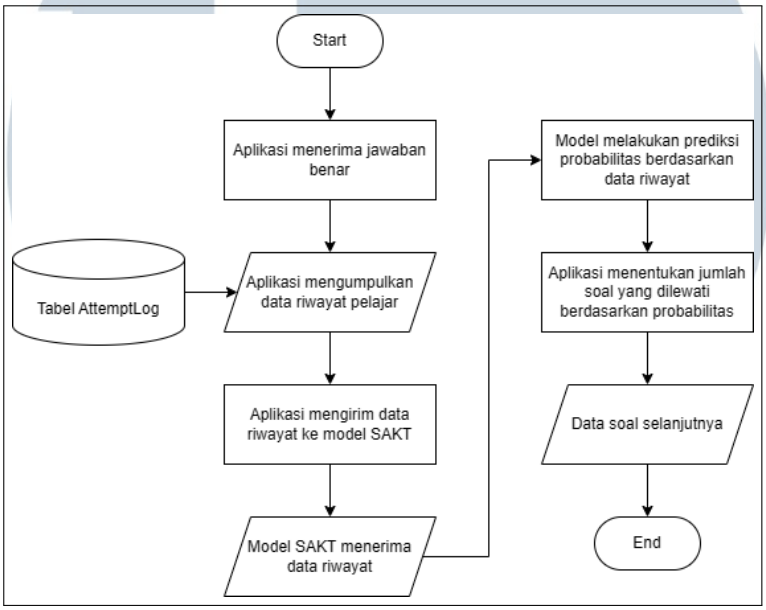


Gambar 3.12. *Flowchart* pemrosesan jawaban pada aplikasi ITS

Flowchart pada Gambar 3.12 menggambarkan pemrosesan jawaban pada aplikasi ITS ketika menerima jawaban dari pelajar. Ketika pelajar menjawab pertanyaan dengan memasukkan jawaban pada kolom jawaban pada halaman materi, aplikasi mengecek kebenaran dari jawaban yang diterima dengan menyocokkan jawaban yang diterima dengan jawaban yang benar untuk soal tersebut berdasarkan data jawaban yang tersimpan pada tabel *Answers*. Setiap kali pelajar mencoba menjawab pertanyaan, percobaan tersebut disimpan dalam bentuk riwayat pada tabel *AttemptLog* yang ada pada *database*. Data riwayat yang disimpan meliputi data pelajar, data materi, data soal yang dijawab, dan kebenaran dari jawaban yang dikirimkan oleh pelajar. Jika jawaban benar, maka riwayat yang disimpan menunjukkan bahwa percobaan menjawab pelajar adalah benar. Sebaliknya, jika percobaan menjawab salah, maka riwayat yang disimpan menunjukkan percobaan salah. Proses ini dilanjutkan dengan melakukan adaptasi menggunakan model SAKT berdasarkan riwayat menjawab pelajar. Setelah proses adaptasi menentukan jumlah soal yang dilewati, aplikasi akan menyimpan *progress*/kemajuan dari pelajar pada materi dan informasi soal selanjutnya yang ditampilkan pada tabel *UserProgress*. Apabila semua soal pada materi sudah dijawab dengan benar oleh pelajar, maka pesan ”Materi ini sudah selesai” ditampilkan kepada pelajar.

5. Adaptasi soal

Setiap kali aplikasi menerima jawaban benar, akan dilakukan prediksi performa. Prediksi ini dilakukan menggunakan model SAKT yang menerima riwayat percobaan menjawab pelajar dan melakukan kalkulasi berdasarkan riwayat tersebut untuk memprediksi probabilitas pelajar menjawab soal selanjutnya dengan benar.



Gambar 3.13. *Flowchart* proses adaptasi soal pada aplikasi ITS

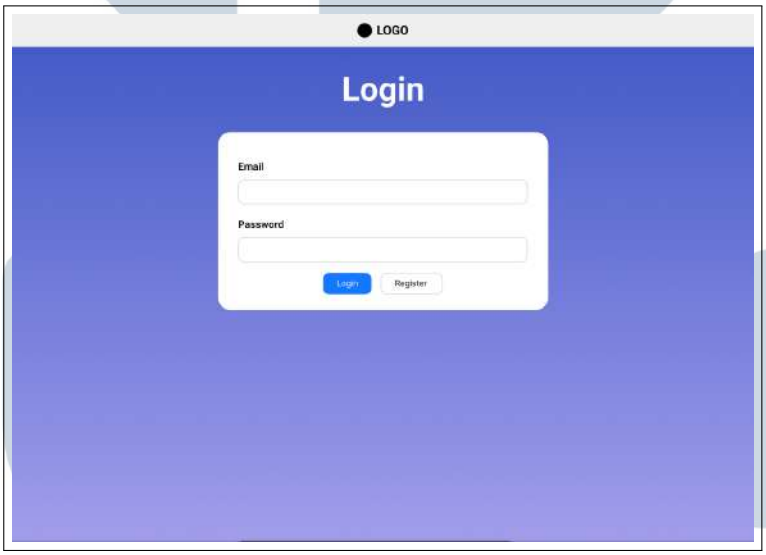
Flowchart dari proses adaptasi soal menggunakan model SAKT dapat dilihat pada Gambar 3.13. Setelah aplikasi menerima jawaban benar, aplikasi mengumpulkan riwayat menjawab pelajar dari tabel *AttemptLog* dan mengirimkan riwayat tersebut ke model SAKT untuk melakukan prediksi probabilitas pelajar tersebut menjawab soal selanjutnya dengan benar. Setelah mendapatkan prediksi probabilitas dari model SAKT, aplikasi akan menyesuaikan soal selanjutnya yang ditampilkan. Jika probabilitas yang diterima tinggi, maka aplikasi akan melewati beberapa soal dan menampilkan soal berikutnya yang berada lebih jauh di dalam materi. Jika probabilitas rendah, maka jumlah soal yang dilewati akan berkurang. Jumlah soal yang dilewati ini diteruskan ke pemrosesan jawaban pada Gambar 3.12 untuk menentukan apakah ada soal selanjutnya atau tidak pada materi berdasarkan data soal yang disimpan pada tabel *Questions*.

3.5.6 Rancangan tampilan antarmuka

Rancangan tampilan antarmuka pengguna (UI) dibuat untuk merepresentasikan tampilan akhir aplikasi *web* ITS. Desain ini mencakup halaman *login*, halaman dasbor pelajar, tampilan soal, dan halaman hasil evaluasi. Perancangan dilakukan menggunakan aplikasi Figma, platform desain UI/UX berbasis *web* yang memungkinkan perancangan secara interaktif dan responsif. Figma juga memberikan gambaran nyata terhadap tampilan dan fungsionalitas aplikasi sebelum dikembangkan secara teknis. Dalam penelitian ini, rancangan UI dibuat dalam sebuah *mockup* dimana rancangan dibuat semirip mungkin dengan hasil akhir tampilan UI.

1. Halaman Login

   Halaman ini memuat beberapa *field* dan tombol-tombol yang dapat digunakan seorang pelajar untuk masuk ke akun mereka. *Mockup* dari halaman ini dapat dilihat pada Gambar 3.14.

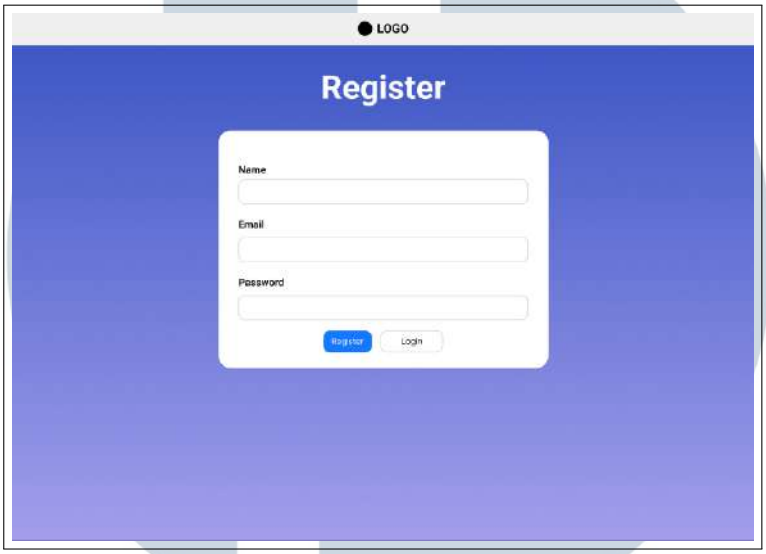


Gambar 3.14. *Mockup* halaman login

Pada halaman ini terdapat *field* email dan *field* kata sandi/*password* yang dapat diisi oleh pelajar sesuai dengan email dan *password* yang sudah mereka daftarkan sebelumnya. Selain itu, terdapat tombol ”Login” dan ”Register”. Pelajar dapat masuk ke akun mereka dengan menekan tombol ”Login” setelah mengisi email dan kata sandi/*password* yang benar. Jika pelajar ingin mendaftarkan akun baru, mereka dapat menekan tombol ”Register” untuk membuka halaman register.

2. Halaman Register

   Halaman ini memuat beberapa *field* dan tombol-tombol yang dapat digunakan seorang pelajar untuk mendaftarkan akun baru. Mockup dari halaman ini dapat dilihat pada Gambar 3.15.

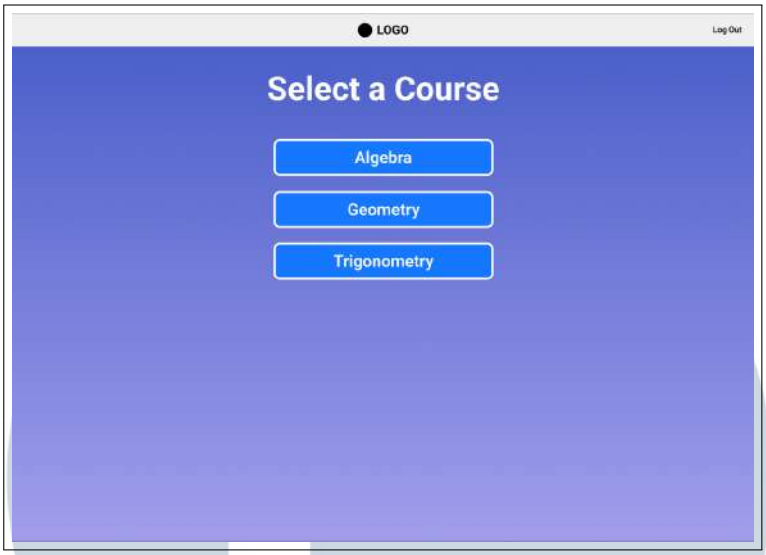


Gambar 3.15. *Mockup* halaman register

   Pada halaman ini terdapat *field* nama, *field* email, *field* kata sandi/*password* yang dapat diisi oleh pelajar untuk mendaftarkan akun baru. Selain itu, terdapat tombol ”Register” dan ”Login”. Pelajar dapat masuk ke akun mereka dengan menekan tombol ”Register” setelah mengisi nama, email, dan kata sandi/*password*. Jika pelajar ingin masuk ke akun yang sudah didaftarkan sebelumnya, mereka dapat menekan tombol ”Login” untuk membuka halaman login.

3. Halaman Dasbor

   Halaman ini memuat tombol-tombol yang dapat ditekan oleh pelajar untuk memilih materi yang ingin mereka pelajari. Mockup dari halaman ini dapat dilihat pada Gambar 3.16.



Gambar 3.16. *Mockup* halaman dasbor

Pada halaman ini, pelajar dapat memilih materi yang ingin mereka pelajari dengan menekan salah satu tombol yang ditampilkan dengan nama materi yang ingin mereka pelajari. Setelah memilih materi, pelajar akan diarahkan ke halaman materi yang dipilih.

4. Halaman *Course*/Materi

   Halaman ini memuat soal yang dapat dikerjakan oleh pelajar dalam mempelajari materi yang mereka pilih dari halaman dasbor. Mockup dari halaman ini dapat dilihat pada Gambar 3.17.

Gambar 3.17. *Mockup* halaman *course*/materi

Pada halaman ini, ditampilkan soal yang dapat dijawab oleh pelajar. Soal tersebut terdiri dari teks soal, persamaaan matematika, dan *field* jawaban. Setelah mengisi jawaban, pelajar dapat menekan tombol ”Submit” untuk mengirimkan jawaban.

Setelah itu, dibuat *high-fidelity prototype* yang menunjukkan alur/*flow* penggunaan aplikasi. Alur ini menunjukkan interaksi yang dapat dilakukan oleh pelajar dengan aplikasi, seperti *input* data dan perpindahan halaman. Tahap ini menambahkan alur interaksi tersebut kepada *mockup* yang sudah dibuat.

Gambar 3.18. Alur interaksi pelajar dengan aplikasi

High-fidelity prototype yang dibuat dapat dilihat pada Gambar 3.18. Interaksi yang dapat dilakukan pelajar dengan aplikasi berdasarkan *prototype* tersebut adalah:

1. Menekan tombol ”Register” pada halaman *login* untuk membuka halaman registrasi.
2. Menekan tombol ”Login” pada halaman registrasi untuk membuka halaman *login*.
3. Menekan tombol ”Login” pada halaman *login* atau tombol ”Register” pada halaman registrasi untuk mengakses halaman dasbor.

4. Pada halaman dasbor dan halaman materi, tersedia tombol *log out* pada kanan atas halaman untuk kembali ke halaman *login*.

5. Materi dapat dipilih dengan menekan salah satu tombol dengan nama materi yang ada pada halaman dasbor dan pelajar diarahkan ke halaman materi yang mereka pilih.

6. Pada halaman materi terdapat tombol ”Return to Dashboard” yang dapat ditekan untuk kembali ke halaman dasbor.

7. Mengisi kolom *input* pada halaman *login* dan registrasi untuk proses autentikasi dan pada halaman materi untuk menjawab soal.

8. Mengirimkan jawaban pada halaman materi dengan menekan tombol ”Submit Answer”.

3.6 Pengembangan Prototipe

Tahap ini merupakan implementasi dari hasil desain sistem sebelumnya, yang berfokus pada pengembangan prototipe aplikasi Intelligent Tutoring System (ITS) berbasis model Self-Attentive Knowledge Tracing (SAKT). Tujuan dari tahap ini adalah membangun versi awal sistem yang telah dirancang, guna mengevaluasi fungsionalitas inti dan integrasi antara *frontend*, *backend*, serta model pembelajaran adaptif. Pembuatan prototipe dilakukan secara bertahap, dimulai dari pembangunan model SAKT, implementasi *backend*, implementasi *frontend*, dan integrasi model SAKT ke sistem.

3.6.1 Pembangunan model SAKT

Tahap pertama dalam pengembangan prototipe adalah membangun sebuah model SAKT yang akan digunakan pada backend aplikasi. Untuk menjalankan fungsi *adaptive learning*, aplikasi ITS menggunakan model SAKT yang dibuat menggunakan bahasa pemrograman Python, *library machine learning* PyTorch, dan dilatih menggunakan *dataset* ASSISTment Skill Builder 2015 [21].

SAKT memiliki keunggulan dalam menangkap pola pembelajaran pelajar tanpa harus mengikuti urutan kronologis secara ketat, sehingga lebih akurat dalam memberikan soal yang sesuai dengan pemahaman pelajar. Probabilitas yang didapat dari hasil prediksi model akan menjadi nilai banding untuk menentukan kesulitan

soal selanjutnya yang diberikan kepada pelajar. Apabila probabilitasnya rendah, maka soal selanjutnya akan lebih mudah dan apabila probabilitasnya tinggi, maka soal selanjutnya akan lebih rumit.

Dalam pembangunan sistem Intelligent Tutoring System (ITS) ini, digunakan model Self-Attentive Knowledge Tracing (SAKT) dari repositori publik GitHub yang dikembangkan oleh Theophile Gervet dengan judul ”Learner Performance Prediction using Transformers” [23]. Repositori tersebut menyediakan implementasi berbagai model *deep learning* untuk prediksi performa belajar siswa, salah satunya adalah model SAKT. Model SAKT pada repositori ini dikembangkan berdasarkan jurnal SAKT oleh Pandey dan Karypis (2019) dan dievaluasi menggunakan skor AUC, sama seperti model SAKT pada jurnal SAKT oleh Pandey dan Karypis (2019) [19]. Dari repositori tersebut, terdapat arsitektur model SAKT, fungsi bantu untuk melakukan preprocessing data, dan script untuk proses pelatihan model. Repositori ini juga sudah memiliki *dataset* ASSISTment 2015 yang sudah diproses, tetapi untuk penelitian ini dataset ASISSTment 2015 akan diproses ulang untuk memastikan bahwa *dataset* yang dipelajari oleh model benar-benar *dataset* yang tersedia secara publik.

Pada penelitian ini, ditambahkan satu *file script* Python untuk mempermudah proses pelatihan berulang dan merubah parameter/*hyperparameter tuning* untuk mendapatkan model dengan skor AUC terbaik. *Hyperparameter* yang dapat diubah pada *script* pelatihan model dapat dilihat pada Tabel 3.8.

Tabel 3.8. Tabel *hyperparameter* pelatihan model SAKT

Hyperparameter	***Example Value***
max_length	200
embed_size	200
num_attn_layers	1
num_heads	5
encode_pos	”store_true”
max_pos	10
drop_prob	0.2
batch_size	100
lr	1e-3
grad_clip	10
num_epochs	100

Menggunakan script tambahan tersebut, *hyperparameter tuning* dilakukan dengan melakukan perubahan angka pada parameter setiap kali dilakukan pelatihan model baru. Parameter yang digunakan dan skor Area Under the Curve (AUC) dari hasil setiap pelatihan model dicatat pada file teks terpisah. File teks tersebut digunakan untuk memilih parameter-parameter terbaik berdasarkan skor AUC yang didapat.

3.6.2 Implementasi *backend*

Lapisan backend bertanggung jawab untuk mengelola data dan menjalankan logika bisnis aplikasi. Backend menangani proses seperti pendaftaran dan autentikasi pengguna, penyimpanan dan pengambilan soal, pencatatan jawaban siswa, serta integrasi dengan model machine learning (SAKT). Backend juga mengatur koneksi dengan *database*, menjalankan proses CRUD (Create, Read, Update, Delete), dan menyediakan RESTful API yang akan dipanggil oleh *frontend*. Bahasa pemrograman dan *framework* yang digunakan dalam pengembangan *backend* adalah TypeScript dengan Node.js dan Express, serta Sequelize untuk ORM (Object Relational Mapping) yang terhubung ke *database* MSSQL.

3.6.3 Implementasi *frontend*

Frontend adalah antarmuka visual yang digunakan oleh pengguna untuk berinteraksi dengan sistem. Pada sisi ini, rancangan prototype dari Figma diterjemahkan menjadi tampilan halaman *web* interaktif. *Frontend* mengatur tampilan halaman *login*, registrasi, dasbor, dan materi. *Framework* yang digunakan untuk mengembangkan *frontend* adalah Next.js dengan *library* React.js. Untuk tampilan visual, digunakan Ant Design sebagai *library* komponen UI yang responsif dan modern.

3.6.4 Integrasi model SAKT ke sistem

Model *Self-Attentive Knowledge Tracing* (SAKT) yang telah dilatih sebelumnya dengan *dataset* ASSISTments 2015 diintegrasikan dengan sistem melalui sebuah *server* yang menyediakan rute API untuk mengakses model SAKT. *Server* ini di-*deploy* menggunakan *library* Flask pada Python dan memuat model SAKT yang sudah dilatih saat *server* dimulai. Dengan metode ini, proses prediksi dapat dilakukan dengan lebih cepat dibandingkan dengan menggunakan *script*

Python yang harus memuat model SAKT setiap kali dipanggil/prediksi probabilitas ingin dilakukan oleh sistem. *Backend* sistem hanya perlu mengirimkan data yang sesuai ke rute API yang disediakan pada *server* model, menunggu proses prediksi, dan menerima angka probabilitas yang dihasilkan oleh model. Proses ini memungkinkan *backend* untuk memutuskan soal berikutnya yang ditampilkan kepada pelajar secara adaptif berdasarkan performa pelajar tersebut.

3.7 Evaluasi Prototipe

Aplikasi yang sudah selesai dibangun akan diuji dengan tujuan memastikan aplikasi yang dihasilkan dapat memenuhi tujuan dibangunnya aplikasi. Pengujian yang dilakukan adalah metrik skor AUC untuk model SAKT, pengujian fungsionalitas sistem (*Black Box Testing*), pengujian performa sistem (*Performance Benchmarking*).

3.7.1 *Black-Box Testing*

Pengujian black-box digunakan untuk mengevaluasi fungsionalitas dari sistem ITS tanpa melihat struktur internal kode atau implementasi teknisnya. Fokus pengujian ini adalah pada keluaran yang dihasilkan sistem berdasarkan masukan yang diberikan pengguna. Metode ini digunakan untuk memverifikasi bahwa setiap fitur, seperti *login*, registrasi, pemilihan materi, dan pengerjaan soal, berfungsi sesuai dengan spesifikasi awal tanpa kesalahan logika atau kegagalan proses. Pada penelitian ini, *blackbox testing* terdiri dari 16 skenario atau *test case* dengan tujuan akhir untuk membangun sebuah aplikasi ITS yang memenuhi ekspektasi sistem pada semua *test case* atau 100% memenuhi ekspektasi semua skenario.

Selain fitur-fitur dasar yang ada pada aplikasi, adaptivitas sistem juga perlu dievaluasi. Evaluasi adaptivitas dilakukan untuk menilai sejauh mana sistem mampu menyesuaikan tingkat kesulitan soal berdasarkan performa pelajar yang terdeteksi oleh model SAKT. Pengujian ini dilakukan dengan memberikan sejumlah jawaban benar atau salah secara berurutan, lalu mengamati apakah sistem mengatur soal lanjutan sesuai dengan probabilitas yang diprediksi. Hasil evaluasi ini menjadi dasar validasi terhadap fungsi utama ITS sebagai sistem pembelajaran adaptif.

3.7.2 *Area Under the ROC Curve Score* (Skor AUC)

Untuk mengevaluasi kinerja model Self-Attentive Knowledge Tracing (SAKT) yang diimplementasikan dalam aplikasi Intelligent Tutoring System (ITS), dilakukan pengujian dengan menggunakan metrik *Area Under Curve* (AUC). Skor AUC merupakan ukuran yang umum digunakan dalam permasalahan klasifikasi biner untuk menilai kemampuan model dalam membedakan antara dua kelas, yaitu jawaban benar dan salah dari pelajar.

Semakin tinggi nilai AUC, maka semakin baik model dalam memprediksi probabilitas jawaban pelajar secara akurat. Nilai AUC berada pada rentang 0 hingga 1, di mana skor mendekati 1 menunjukkan prediksi yang sangat baik, sedangkan skor mendekati 0.5 mengindikasikan bahwa model tidak lebih baik dari tebakan acak.

3.7.3 *Performance Benchmarking*

Evaluasi performa sistem dilakukan untuk mengukur efisiensi dan kecepatan respon sistem dalam memproses permintaan pengguna, terutama saat berinteraksi dengan model SAKT. Pengujian ini mencakup waktu eksekusi proses prediksi dan waktu respons API dalam menangani permintaan berturut-turut. Tujuan dari *benchmarking* ini adalah memastikan bahwa sistem mampu memberikan umpan balik secara *real-time* tanpa menyebabkan keterlambatan signifikan yang dapat mengganggu pengalaman pengguna.

3.8 Revisi dan Pengembangan Akhir

Berdasarkan hasil evaluasi terhadap prototipe sistem, dilakukan beberapa penyesuaian untuk meningkatkan akurasi fungsi adaptasi dan kualitas pengalaman pengguna. Revisi dilakukan terhadap aspek tertentu seperti tampilan antarmuka pengguna, logika pemilihan soal, serta integrasi dengan model SAKT agar proses prediksi berjalan lebih efisien. Dalam penelitian ini, beberapa iterasi dari aplikasi dibangun dan dievaluasi yang dapat dilihat pada Tabel 3.9.

Tabel 3.9. Tabel iterasi prototipe aplikasi ITS

Prototipe	Deskripsi	Evaluasi
Prototipe 1	Aplikasi *web* dasar hanya dengan fungsi pemberian soal, penerimaan jawaban, dan pengecekan jawaban dengan model SAKT yang diintegrasi langsung pada *backend*	Proses prediksi model setiap kali menerima jawaban benar membutuhkan waktu yang cukup lama
Prototipe 2	Model SAKT dimuat pada sebuah *server* terpisah menggunakan Flask dan ditambahkan *progress bar* untuk mengindikasikan *progress* pelajar dalam materi	Proses prediksi lebih cepat dan hampir instan, tetapi data pelajar masih *hardcode* dan memerlukan fitur autentikasi
Prototipe 3	Dibuat halaman *login* dan registrasi untuk proses autentikasi, dibuat halaman dasbor untuk memberikan pilihan materi, dan ditambahkan tombol serta fungsi *log out*	Proses autentikasi berhasil, tetapi validasi *input* dan pesan *warning* belum bekerja dengan baik
Prototipe 4	Semua kolom *input* diberikan fungsi validasi dan untuk kolom *email*, diberikan juga fungsi pengecekan format *input*.	Semua fitur sudah bekerja sesuai ekspektasi dan dapat dilanjutkan ke penyempurnaan UI

Setelah perbaikan dilakukan, prototipe diperbarui menjadi versi yang lebih stabil dan digunakan sebagai hasil akhir dalam penelitian ini. Pengembangan akhir ini tidak berfokus pada penyempurnaan fitur tambahan, melainkan pada penyempurnaan proses utama untuk mendukung tujuan utama penelitian, yaitu menunjukkan kelayakan penerapan model SAKT dalam sistem pembelajaran adaptif menggunakan ITS pada mata pelajaran matematika.