IMPLEMENTASI TELEGRAM CHATBOT BERBASIS MENU UNTUK INFORMASI AKADEMIK E-CLASS

Skripsi

oleh:

BENEDIKTUS PURNOMO AJI
71180333

PROGRAM STUDI INFORMATIKA FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN DUTA WACANA
2023

IMPLEMENTASI TELEGRAM CHATBOT BERBASIS MENU UNTUK INFORMASI AKADEMIK E-CLASS

Skripsi

Diajukan kepada Program Studi Informatika Fakultas Teknologi Informasi
Universitas Kristen Duta Wacana
Sebagai Salah Satu Syarat dalam Memperoleh Gelar
Sarjana Komputer

Disusun oleh

BENEDIKTUS PURNOMO AJI
71180333

PROGRAM STUDI INFORMATIKA FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN DUTA WACANA
2023

PERNYATAAN KEASLIAN SKRIPSI

Saya menyatakan dengan sesungguhnya bahwa skripsi dengan judul:

IMPLEMENTASI TELEGRAM CHATBOT BERBASIS MENU UNTUK INFORMASI AKADEMIK E-CLASS

yang saya kerjakan untuk melengkapi sebagian persyaratan menjadi Sarjana Komputer pada pendidikan Sarjana Program Studi Informatika Fakultas Teknologi Informasi Universitas Kristen Duta Wacana, bukan merupakan tiruan atau duplikasi dari skripsi kesarjanaan di lingkungan Universitas Kristen Duta Wacana maupun di Perguruan Tinggi atau instansi manapun, kecuali bagian yang sumber informasinya dicantumkan sebagaimana mestinya.

Jika dikemudian hari didapati bahwa hasil skripsi ini adalah hasil plagiasi atau tiruan dari skripsi lain, saya bersedia dikenai sanksi yakni pencabutan gelar kesarjanaan saya.

Yogyakarta, 18 April 2023

BENEDIKTUS PURNOMO AJI
71180333

HALAMAN PERSETUJUAN

Judul Skripsi	:	IMPLEMENTASI TELEGRAM CHATBOT BERBASIS MENU UNTUK INFORMASI AKADEMIK E-CLASS
Nama Mahasiswa	:	BENEDIKTUS PURNOMO AJI
NIM	:	71180333
Mata Kuliah	:	Skripsi (Tugas Akhir)
Kode	:	TI0366
Semester	:	Genap
Tahun Akademik	:	2022/2023

Telah diperiksa dan disetujui di
Yogyakarta,
Pada tanggal 18 April 2023

Dosen Pembimbing I	Dosen Pembimbing II
Willy Sudiarto Raharjo, S.Kom., M.Cs	Dr. Phil. Lucia Dwi Krisnawati, S.S., M.A.

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS SECARA ONLINE UNIVERSITAS KRISTEN DUTA WACANA YOGYAKARTA

Saya yang bertanda tangan di bawah ini:

NIM	:	71180333
Nama	:	Benediktus Purnomo Aji
Prodi / Fakultas	:	Teknologi Informasi / Informatika
Judul Tugas Akhir	:	Implementasi Telegram *Chatbot* Berbasis Menu Untuk Layanan Informasi Akademik E-Class

bersedia menyerahkan Tugas Akhir kepada Universitas melalui Perpustakaan untuk keperluan akademis dan memberikan **Hak Bebas Royalti Non Ekslusif** (*Non-exclusive Royalty-free Right*) serta bersedia Tugas Akhirnya dipublikasikan secara online dan dapat diakses secara lengkap (*full access*).

Dengan Hak Bebas Royalti Nonekslusif ini Perpustakaan Universitas Kristen Duta Wacana berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk *database*, merawat, dan mempublikasikan Tugas Akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenar-benarnya.

Yogyakarta, 18 April 2023

Yang menyatakan,

(71180333 – BenediktusPurnomo Aji)

HALAMAN PENGESAHAN

IMPLEMENTASI TELEGRAM CHATBOT BERBASIS MENU UNTUK LAYANAN INFORMASI AKADEMIK E-CLASS

Oleh: BENEDIKTUS PURNOMO AJI / 71180333

Dipertahankan di depan Dewan Penguji Skripsi
Program Studi Informatika Fakultas Teknologi Informasi
Universitas Kristen Duta Wacana - Yogyakarta
Dan dinyatakan diterima untuk memenuhi salah satu syarat memperoleh gelar
Sarjana Komputer
pada tanggal 30 Maret 2023

Yogyakarta, 30 Maret 2023
Mengesahkan,

Dewan Penguji:

1. Willy Sudiarto Raharjo, S.Kom., M.Cs
2. Dr. Phil. Lucia Dwi Krisnawati, S.S., M.A.
3. Budi Susanto, S.Kom., M.T.
4. Kristian Adi Nugraha, S.Kom., M.T.

Dekan

(Restyandito, S.Kom., MSIS.,Ph.D.)

Ketua Program Studi

(Gloria Virginia, S.Kom., MAI, Ph.D.)

Karya sederhana ini dipersembahkan

kepada Tuhan, Keluarga Tercinta,

dan Kedua Orang Tua

Hidup ini Bagai skripsi,

Banyak bab dan revisi yang harus dilewati.

Namun akan selalu berakhir indah, bagi yang pantang menyerah.

Alit Susanto

KATA PENGANTAR

Segala puji dan syukur kepada Tuhan yang maha kasih, karena atas segala rahmat, bimbingan, dan bantuan-Nya maka akhirnya Skripsi dengan judul IMPLEMENTASI TELEGRAM CHATBOT BERBASIS MENU UNTUK LAYANAN INFORMASI AKADEMIK ini telah selesai disusun.

Penulis memperoleh banyak bantuan dari kerja sama baik secara moral maupun spiritual dalam penulisan Skripsi ini, untuk itu tak lupa penulis ucapkan terima kasih yang sebesar-besarnya kepada:

1. Tuhan yang maha kasih,
2. Orang tua yang selama ini telah sabar membimbing dan mendoakan penulis tanpa kenal untuk selama-lamanya,
3. Restyandito, S.Kom., MSIS., Ph.D. selaku Dekan FTI, yang senantiasa memberikan arahan dan contoh yang baik bagi seluruh mahasiswa
4. Gloria Virginia, S.Kom., MAI, Ph.D. selaku Kaprodi Informatika, yang telah mengusahakan dinamika perkuliahan yang terbaik seluruh mahasiswa
5. Willy Sudiarto Raharjo, S.Kom., M.Cs selaku Dosen Pembimbing 1, yang telah memberikan ilmunya dengan penuh kesabaran bagi penulis,
6. Dr. Phil. Lucia Dwi Krisnawati, S.S., M.A., selaku Dosen Pembimbing 2 yang telah memberikan ilmu dan kesabaran dalam membimbing penulis,
7. Keluarga tercinta: yang telah mendukung perjalanan studi dari awal hingga akhir
8. Lain-lain yang telah mendukung moral, spiritual, dan dana untuk belajar selama ini.

Laporan skripsi ini tentunya tidak lepas dari segala kekurangan, untuk itu segala kritikan dan saran sangat diharapkan. Semoga skripsi ini dapat bermanfaat bagi pembaca semua dan lebih khusus lagi bagi pengembagan ilmu komputer dan teknologi informasi.

Yogyakarta, 18 April 2023

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

INTISARI

IMPLEMENTASI TELEGRAM CHATBOT BERBASIS MENU UNTUK LAYANAN INFORMASI AKADEMIK E-CLASS

Oleh

BENEDIKTUS PURNOMO AJI
71180333

Layanan informasi akademik merupakan kebutuhan yang wajib hadir di semua tingkatan pendidikan. Di Universitas Kristen Duta Wacana, layanan ini tersedia dalam bentuk website bernama e-class. Terdapat setidaknya dua kekurangan dari website e-class yaitu tampilan yang belum *mobile friendly* dan juga belum ada fitur notifikasi untuk informasi yang baru.

Penelitian ini membangun sebuah aplikasi *chatbot* berbasis menu menggunakan telegram untuk informasi akademik pada website e-class. *Chatbot* ini menyediakan informasi akademik berupa jadwal perkuliahan, pengumuman, materi, tugas, nilai, diskusi hingga ke presensi perkuliahan. Selain itu *chatbot* juga akan mengirimkan notifikasi ketika terdapat informasi baru.

Penelitian dilakukan menggunakan *User Acceptance Test* dengan metode *Alpha/Beta Testing* terhadap chatbot untuk layanan informasi akademik pada website e-class. *Alpha Testing* dilakukan terhadap tiga responden dari program studi yang berbeda menunjukkan hasil 97,70% (Sangat Layak). Sementara *Beta Testing* yang dilakukan terdahap 30 responden menunjukkan hasil *User Acceptance Index* sebesar 81,33% dan termasuk ke dalam kategori sangat layak.

Kata-kata kunci: *chatbot*, telegram, e-class, notifikasi

ABSTRACT

IMPLEMENTATION OF MENU BASED TELEGRAM CHATBOT FOR ACADEMIC INFORMATION IN E-CLASS

By

BENEDIKTUS PURNOMO AJI
71180333

Academic information services are a mandatory requirement at all levels of education. At Duta Wacana Christian University, this service is available in the form of a website called e-class. There are at least two drawbacks of e-class which are the interface is not mobile friendly and there is also no notification feature for new information.

This research builds a menu based chatbot using telegram for academic information on e-class. This chatbot provides academic information such as class schedules, assignments, announcements, materials, grades, discussions, and lecture attendance. In addition, the chatbot will also send notifications when there is a new information.

The research was conducted using User Acceptance Test with the Alpha/Beta Testing method for chatbots for academic information services on e-class websites. Alpha Testing conducted on three respondents from different study programs showed a result of 97.70% (Very Eligible). While the Beta Testing conducted on 30 respondents showed a User Acceptance Index result of 81.33% and was included in the very eligible category.

Keywords: chatbot, telegram, e-class, notification

BAB I
PENDAHULUAN

1.1. Latar Belakang Masalah

Layanan informasi akademik merupakan kebutuhan yang wajib hadir di semua tingkatan pendidikan. Di level universitas, layanan ini biasanya berupa sebuah website yang memungkinkan mahasiswa maupun dosen untuk melihat, menggunggah, dan mengunduh segala hal yang berkaitan dengan akademik. Contohnya adalah jadwal perkuliahan, tugas, pengumuman, nilai, daftar kehadiran, dan lain sebagainya. Di Universitas Kristen Duta Wacana, layanan ini tersedia dalam bentuk website bernama e-class.

Seperti website pada umumnya, e-class dapat diakses melalui berbagai platform yang ada, termasuk melalui *smartphone*. Permasalahan yang muncul adalah tampilan website e-class saat ini belum *mobile friendly*. Masih terdapat tombol dan tulisan yang terlalu kecil sehingga menyulitkan pengguna saat mengakses melalui *smartphone*. Selain itu, tidak adanya fitur notifikasi ketika terdapat tugas atau pengumuman baru juga menimbulkan permasalahan. Permasalahan fitur notifikasi sudah pernah diteliti oleh (Efendy, Nugraha, & Sebastian, 2019). Namun penelitian tersebut memiliki kekurangan yaitu sistem yang dikembangkan tidak dapat menangani proses login oleh dua pengguna atau lebih secara bersamaan karena rentan terjadi kesalahan pengambilan data dari e-class. Salah satu solusi dari kedua permasalahan ini adalah dengan membuat sebuah *chatbot* yang dapat mengemas informasi akademik di website e-class menjadi lebih mudah diterima pengguna, termasuk mengirimkan notifikasi saat terdapat tugas atau pengumuman baru.

Dalam penelitian ini penulis akan membangun sebuah aplikasi *chatbot* berbasis menu menggunakan telegram bot API untuk informasi akademik pada website e-class. Di chatbot tersebut, pengguna akan diberikan berbagai pilihan untuk mengakses informasi akademik yang ada. Informasi tersebut seperti jadwal perkuliahan, tugas, pengumuman, nilai, hingga ke materi perkuliahan. Aplikasi

pesan telegram dipilih sebagai media dalam penelitian ini karena karakteristiknya yang ringan dan multi-platform (dapat digunakan di semua sistem operasi) sehingga memudahkan dalam pengembangan kedepan. Selain itu, dukungan telegram bot API dan komunitasnya akan membantu dalam penelitian ini.

1.2. Perumusan Masalah

Berdasarkan uraian pada latar belakang di atas, maka masalah yang akan diteliti pada skripsi ini adalah hasil uji *User Acceptance Test* dengan metode *Alpha/Beta Testing* terhadap telegram *chatbot* untuk layanan informasi akademik pada website e-class.

1.3. Batasan Masalah

Penelitian ini akan dibatasi oleh hal-hal berikut:

1. Website yang akan digunakan adalah e-class.
2. Telegram *chatbot* akan menyediakan informasi berupa jadwal perkuliahan, materi, pengumuman, tugas, diskusi, nilai dan presensi.

1.4. Tujuan Penelitian

Tujuan penelitian ini adalah membangun sebuah telegram *chatbot* untuk layanan informasi akademik pada website e-class dengan menggunakan metode *webhook*.

1.5. Manfaat Penelitian

Hasil dari penelitian pada skripsi ini dapat digunakan sebagai pembanding dengan penelitian sebelumnya untuk fitur notifikasi yang ada pada website e-class. Apabila ketepatan pengiriman notifikasi chatbot cukup baik, maka nantinya dapat dikembangkan menjadi aplikasi e-class yang lebih kompleks.

1.6. Sistematika Penulisan

Dalam penulisan ini, terdapat beberapa bab dan sub bab penulisan laporan, yaitu meliputi Pendahuluan, Tinjauan Pustaka dan Landasan Teori, Metodologi Penelitian, Hasil dan Pembahasan, serta Kesimpulan dan Saran.

- BAB I Pendahuluan

  Bab ini akan menjelaskan tentang latar belakang masalah, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metolodogi penelitian, dan sistematika penulisan.
- BAB II Tinjauan Pustaka dan Landasan Teori

  Bab ini akan berisi tentang tinjauan pustaka dan landasan teori. Tinjauan pustaka akan menjelaskan tentang penelitian yang sudah ada sebelumnya yang digunakan sebagai referensi dalam penelitian ini. Landasan teori akan menjelaskan tentang teori dan definisi yang akan digunakan sebagai referensi.
- BAB III Metodologi Penelitian

  Bab ini akan menjelaskan tentang perancangan penelitian, arsitektur sistem, diagram alir, *sequence diagram,* perancangan basis data, perancangan antarmuka pengguna, dan perancangan pengujian sistem.
- BAB IV Implementasi dan Pembahasan

  Bab ini akan menjelaskan tentang implementasi sistem, pengujian dan analisis, serta pembahasan.
- BAB V Kesimpulan dan Saran

  Bab ini akan menjelaskan tentang kesimpulan yang didapatkan dan saran untuk penelitian sejenis yang akan dilakukan kedepannya.

BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Sejumlah penelitian tentang penggunaan telegram *chatbot* untuk mendukung layanan informasi akademik sudah banyak dilakukan. Penelitian ini dirasa penting karena akan semakin membantu terciptanya *smart campus* yang mulai dilirik oleh banyak universitas. Pada umumnya penelitian yang dilakukan terkait penggunaan telegram *chatbot* ini berupa pengemasan informasi akademik agar lebih mudah dipahami melalui *smartphone*. Namun ada juga penelitian yang dilanjutkan sampai merancang sistem automatisasi layanan.

Penelitian tentang pemanfaatan telegram *chatbot* untuk layanan informasi akademik pernah dilakukan oleh Sastrawangsa (2017). Dalam penelitiannya, *chatbot* yang dikembangan merupakan *chatbot* berbasis *command* atau perintah. Hal ini mengharuskan pengguna untuk menuliskan perintah terlebih dahulu saat ingin menggunakan layanan *chatbot*. Perintah tersebut akan diolah oleh *server* dari *chatbot* yang kemudian akan mencari data yang diminta pengguna pada *database*. Data tersebut kemudian akan dikirimkan kepada pengguna oleh *server chatbot*. Penelitian ini juga memberikan fokus kepada rancangan sistem automatisasi layanan berupa pencarian informasi kampus dan permintaan terhadap form.

Selain *chatbot* berbasis *command* atau perintah, terdapat pula *chatbot* yang dibuat berbasis menu. *Chatbot* berbasis menu ini pernah diteliti oleh Leonardo, Herianto, & Irawan (2020) untuk kasus STMIK Hang Tuah di Pekanbaru. Perbedaan utama antara *chatbot* berbasis *command* dan berbasis menu adalah pada *chatbot* berbasis menu, pengguna hanya tinggal menekan pilihan-pilihan menu yang sudah disediakan oleh *chatbot*. Dalam kata lain, pengguna tidak perlu mengetikan perintah untuk setiap layanan yang diinginkan karena sudah terdapat menu yang menggantikannya. Penelitian ini juga menerapkan metode *webhook* sehingga setiap informasi yang diminta merupakan informasi yang realtime.

Yehezkiel, (2019) pernah melakukan penelitian untuk mengubah tampilan e-class agar menjadi lebih *mobile friendly* dengan cara membuat versi *mobile*. Hasil penelitian menunjukkan peningkatkan nilai *performance metrics* sebesar 10-60% dan hasil pengukuran *System Usability Scale* (SUS) yang masuk kategori diterima. Namun dalam penelitian ini sistem operasi android menjadi batasan pengembangan e-class versi *mobile* sehingga smartphone dengan sistem operasi selain android belum dapat menggunakanya.

Usaha membuat notifikasi untuk e-class juga sudah pernah dilakukan oleh Efendy, Nugraha, & Sebastian (2019). Pada penelitian tersebut, notifikasi e-class dibuat menggunakan bantuan dari *Firebase Cloud Messaging*. Tahapan yang dilakukan adalah dengan melakukan *scraping* website e-class dengan menggunakan nim dan password pengguna. Data dari tahap ini kemudian disaring dan diteliti apakah terdapat data baru (berbeda) di dalamnya. Jika terdapat data baru, maka *server* akan mengirimkan notifikasi pada smartphone pengguna. Penelitian ini juga mengimplementasikan fitur *chat room* yang memungkinkan komunikasi antar pengguna di e-class seperti dosen dan asistennya atau sesama mahasiswa. Terdapat dua kelemahan dari penelitian ini yaitu sistem yang dikembangkan tidak dapat melakukan login oleh dua pengguna atau lebih secara bersamaan. Selain itu, sama seperti penelitian yang dilakukan oleh Yehezkiel, (2019), penelitian ini masih terbatas pada sistem operasi android saja.

Berdasarkan penelitian-penelitian sebelumnya, penelitian kali ini akan memberikan fokus kepada pembuatan *chatbot* di telegram untuk informasi akademik yang tersedia pada website e-class. Informasi yang dimaksudkan berupa jadwal perkuliahan, materi, pengumuman, tugas, nilai, diskusi, dan presensi. *Chatbot* ini juga memiliki fitur yang dapat mengirimkan notifikasi kepada penggunanya ketika terdapat materi, pengumuman, tugas, nilai, diskusi, dan presensi baru di e-class. Penelitian ini akan mencoba mengimplementasikan hasil penelitian sebelumnya yaitu layanan informasi akademik (Leonardo, Herianto, & Irawan, 2020) dan notifikasi untuk e-class (Efendy, Nugraha, & Sebastian, 2019) ke dalam aplikasi *chatbot* di telegram. Pemilihan telegram sebagai media implementasi berdasarkan pada karakteristik dari aplikasi telegram yang ringan,

multi-platform (Android, iOS, Windows, macOS, dan Linux), dan ketersediaan API yang memadai.

2.2 Landasan Teori

2.2.1 Chabot

Menurut Khan & Das (2018) chatbot merupakan sebuah program yang dapat memproses masukkan dari pengguna berupa bahasa natural (bahasa yang digunakan sehari-hari) dan menghasilkan respons yang sesuai kepada pengguna. Hal ini tentu menjadi keunggulan dibandingkan jenis tampilan lainnya, karena pengguna dapat berinteraksi dengan bahasa sehari-hari tanpa perlu menuliskan perintah khusus. Namun dari sisi pengembangan chatbot, pengolahan respon pengguna dengan bahasa sehari-hari merupakan proses yang paling sulit. Perlu dilakukan parsing teks tersebut supaya mendapatkan maksud yang sesungguhnya dan kemudian mengirimkan respon balik yang tepat.

Khan & Das (2018) dalam bukunya menjelaskan beberapa jenis elemen tampilan yang dapat digunakan pada chatbot sebagai alternatif dari penggunaan bahasa natural. Beberapa jenis elemen tampilan tersebut adalah:

1. *Carousel*

   Carousel merupakan sebuah koleksi dari beberapa item yang dapat diakses secara horizontal. Item pada *carousel* biasanya terdiri dari judul, gambar, dan tombol. Contoh penggunaan *carousel* adalah pada saat ingin menampilkan katalog film yang sedang ditayangkan di bioskop.

2. *Quick Replies*

   Quick Replies atau balasan cepat merupakan sebuah tampilan pada chatbot yang terdiri dari beberapa pilihan balasan. Balasan tersebut biasanya berupa teks yang ketika diklik oleh pengguna akan mengirimkan respon yang sudah ditentukan kepada chatbot.

3. *Buttons*

   Buttons atau tombol dapat digunakan pada sebuah *chatbot* untuk mendapatkan respon dari penggunanya. Tampilan ini biasanya terdiri

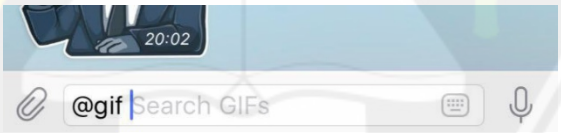
dari beberapa tombol dengan teks singkat di dalamnya untuk menjelaskan aksi dari tombol tersebut.

4. *Web Views*

   Web Views merupakan jenis elemen yang dapat digunakan untuk menampilkan halaman HTML. Jenis elemen ini biasa digunakan untuk menangani respon pengguna yang kompleks. Contohnya adalah ketika ingin memilih kursi saat akan menonton di bioskop. Informasi yang ingin ditampilkan adalah seluruh kursi, kursi yang tidak tersedia, dan kursi yang tersedia. Informasi seperti ini akan lebih mudah diimplementasikan dengan elemen *web views*.

2.2.2 Telegram Bots

Telegram Bots merupakan sebuah *bot* yang berjalan di aplikasi telegram. *Bot* ini memungkinkan pengguna untuk untuk berinteraksi dengan dua cara yaitu mengirimkan pesan atau perintah langsung kepada bot dan mengirimkan *inline queries*. Untuk cara yang kedua, *bot* tersebut biasa dinamakan *inline bots* (Bots: An introduction for developers, 2022). Contoh penggunaan *inline bots* adalah sebagai berikut:



Gambar 2.1. Contoh inline bots pada Telegram
(Bots: An introduction for developers, 2022)

Kata @gif merupakan username dari *bot*, sedangkan kata berikutnya merupakan kueri yang nantinya akan ditangani oleh *bot* tersebut.

Seperti yang dilansir pada grammY (2022), terdapat dua metode yang dapat digunakan untuk menjalankan telegram bots yaitu:

1. *Long Polling*

   Long Polling merupakan sebuah metode dimana *server bot* akan secara aktif mengirimkan *request* kepada *server* telegram. Jika terdapat pesan

baru pada *server* telegram, maka pesan tersebut akan dikirimkan ke server bot untuk kemudian diolah responsnya. Namun jika tidak terdapat pesan baru, maka koneksi antara *server* bot dan *server* telegram akan tetap dibuka sampai waktu *timeout* yang telah ditentukan.

2. *Webhook*

   Metode *webhook* berjalan dengan cara menanamkan *bot* yang dibuat pada sebuah URL yang dapat diakses secara publik. Server telegram akan mengirimkan pesan kepada *bot* lewat URL yang sudah disediakan ketika terdapat pesan baru di *server*. Jika tidak terdapat pesan baru, maka *server* telegram tidak akan mengirimkan apa-apa. Ketentuan tambahan jika menggunakan metode ini adalah perlunya penggunaan sertifikat SSL (*Secure Socket Layer*).

2.2.3 Web Scraping

Menurut Broucke & Baesens (2018), *web scraping* merupakan sebuah aktivitas untuk mengunduh, mengurai, dan mengatur data dari sebuah situs secara otomatis. Aktivitas ini biasa dilakukan dengan menggunakan sebuah *scraper* atau *bot*. Kegiatan *web scraping* sebenarnya sudah muncul sejak lama dengan istilah "*screen scraping*" merujuk pada kegiatan untuk mengekstrak atau mengambil data dari representasi visual seperti website.

Proses pengambilan data dari sebuah website sebenarnya sudah dapat dilakukan dengan menggunakan API (*Application Programming Interface*). Kendala yang muncul biasanya adalah tidak semua website mempunya API gratis yang dapat digunakan secara publik. Bahkan jika website tersebut mempunyai API tetap saja terdapat batasan dalam penggunaannya. Batasan tersebut seperti API yang berbayar, *limit request*, atau batasan data yang dapat diperoleh. Batasan-batasan tersebut dapat diatasi dengan menggunakan *web scraping* (Mitchell, 2018).

2.2.4 Puppeteer

Puppeteer merupakan sebuah *library* pada Node.js yang menyediakan API tingkat tinggi untuk mengontrol jalannya sebuah browser berbasis Chromium melalui protokol *DevTools*. Menurut Kondratiuk, (2021), beberapa hal yang dapat dilakukan menggunakan puppeteer adalah sebagai berikut:

1. *Task Automation*

   Mengingat kemampuan dari puppeteer untuk mejalankan sebuah browser dan perintah-perintahnya, ini membuka peluang untuk digunakan sebagai *task automation.* Contoh skenarionya adalah untuk membuka browser, mengunjungi sebuah situs dan kemudian mengisi form yang diminta pada situs tersebut secara otomatis.

2. *Web Scraping*

   Puppeteer dapat menjalankan berbagai perintah untuk mengambil data-data yang dibutuhkan dari sebuah situs. Perintah-perintah tersebut dituliskan dengan menggunakan *syntax* seperti JQuery.

3. *Content Generation*

   Content Generation merupakan sebuah istiliah untuk kegiatan membuat sebuah konten dari data yang sudah ada. Dalam puppeteer, dua hal yang dapat dilakukan terkait *content generation* adalah melakukan *screenshot* dan membuat pdf. Untuk melakukan *screenshot* pada sebuah halaman situs dapat menggunakan perintah *page.screenshot([options])*. Sedangkan untuk membuat sebuah pdf dapat dilakukan dengan menuliskan perintah *page.pdf([options]).*

4. *End-to-End Testing*

   End-to-End Testing merupakan sebuah metode untuk menguji fungsionalitas dan performa dari sebuah aplikasi mulai dari awal hingga akhir dalam kondisi mendekati kondisi sesungguhnya. Puppeteer mempunyai arsitektur dan perilaku yang sangat mirip dengan *browser* sesungguhnya sehingga cocok digunakan untuk metode *testing* ini.

2.2.5 Cloud Firestore

Firebase Cloud Firestore atau lebih dikenal dengan Cloud Firestore merupakan sebuah produk database NoSQL yang disediakan oleh Google. Cloud Firestore hadir sebagai produk lanjutan dari versi terdahulunya yaitu Firebase Realtime Database dengan model data yang lebih intuitif (Rosyana, Onno, & RZ., 2021).

Menurut Yahiaoui (2017), Cloud Firestore menawarkan beberapa kelebihan dibandingkan dengan produk terdahulunya yaitu:

1. Model data yang disimpan di Cloud Firestore lebih rapi dan terstruktur. Data pada Cloud Firestore disimpan dalam sebuah dokumen pada sebuah koleksi, berbeda dengan Realtime Database yang menyimpan semua data dalam sebuah hirerarki JSON.
2. Skalabilitas yang baik

   Cloud Firestore memang didesain untuk menangani skalabilitas dari sebuah database. Artinya semakin banyak data yang ada pada database tidak akan mempengaruhi performa dari database.
3. Fitur kueri yang lengkap

   Fitur kueri pada Cloud Firestore merupakan fitur canggih yang dibangun dengan fokus pada performa. Artinya pengguna dapat memilih untuk mengambil beberapa dokumen pada sebuah koleksi lalu memberikan filter dan sorting. Performanya pun sesuai dengan proporsi dari hasil kueri yang dilakukan bukan dari data yang ada pada database.
4. Dukungan *realtime* dan *offline*

   Cloud Firestore menawarkan dukungan *realtime* terhadap data yang ada pada database tentu dengan bantuan fitur query yang lengkap. Fitur *offline* juga hadir dengan menggunakan bantuan *cache.* Ketika perangkat sudah kembali terhubung ke internet maka Cloud Firestore akan menyinkronkan perubahan yang sudah terjadi ke dalam database mereka.

5. Dukungan terhadap produk Firebase lainnya

   Cloud Firestore dapat bekerja dengan baik terhadap produk Firebase lainnya seperti Firebase Authentication dan Cloud Function.

2.2.6 User Acceptance Test

Menurut Hambling & Goathem (2013), *User Acceptance Test* (UAT) adalah sebuah kegiatan uji coba terhadap sebuah software yang dilakukan oleh pengguna. Tujuan utama dari dilakukannya UAT adalah untuk memastikan bahwa versi terbaru dari software atau aplikasi yang dikembangkan memenuhi kebutuhan yang diperlukan dan dapat melakukannya dengan baik. Terdapat empat bentuk dari *user acceptance test* yang kerap digunakan yaitu:

1. *Alpha/Beta Testing*

   Alpha Testing merupakan sebuah pengujian terhadap aplikasi yang dikembangkan yang dilakukan oleh tim internal atau representasi dari pengguna. Sedangkan *Beta Testing* merupakan sebuah pengujian terhadap aplikasi yang dilakukan oleh pengguna dengan kondisi lingkungan pengujian yang mendekati situasi sesungguhnya. Pengujian ini biasa dilakukan sebelum aplikasi akhirnya siap digunakan kepada publik.

2. *Contract Acceptance Testing*

   Contract Acceptance Testing merupakan sebuah pengujian yang dilakukan berdasarkan pada kebutuhan sistem yang dimuat dalam kontrak kerja. Pengujian ini dilakukan untuk memastikan bahwa setiap kebutuhan yang terdapat pada kontrak sudah terpenuhi.

3. *Factory/Site Acceptance Testing*

   Factory/Site Acceptance Testing merupakan sebuah pengujian yang biasa dilakukan setelah implementasi aplikasi di lokasi kerja. Pengujian jenis ini biasa dilakukan pada aplikasi skala besar yang memerlukan proses instalasi yang kompleks.

4. *Field Testing*

   Field Testing atau uji lapangan merupakan sebuah pengujian yang dilakukan pada aplikasi yang akan digunakan dalam kasus tertentu di lapangan. Contohnya adalah pemasangan alat pendeteksi asap di suatu gedung. Pemasangan alat ini perlu diikuti dengan sebuah *field testing* agar mengetahui seberapa baik alat ini dapat bekerja pada kasus tertentu.

BAB III
METODOLOGI PENELITIAN

3.1 Perancangan Penelitian

Penelitian ini dilakukan dengan menggunakan Telegram bot API, Node.js untuk menangani *server*, dan Cloud Firestore untuk menangani data dari pengguna. Tahapan pertama adalah dengan mendaftarkan *chatbot* yang akan digunakan ke *bot* BotFather. Setelah itu dilanjutkan dengan pengembangan *server* berbasis Node.js yang terkoneksi dengan Cloud Firestore.

Proses utama yang dapat dilakukan oleh *chatbot* ini adalah memberikan data sesuai dengan pilihan menu yang dipilih oleh pengguna. Setiap pilihan yang diambil akan direspon dengan *server* yang kemudian akan mengambil data dari Cloud Firestore. Namun bagi pengguna yang baru pertama kali menggunakan *chatbot*, akan dilakukan proses *web scraping* pada e-class terlebih dahulu. Data tersebut kemudian akan disimpan ke dalam Cloud Firestore. Proses penyimpanan data di database dipilih untuk menghindari *scraping* data secara terus menerus pada website e-class.

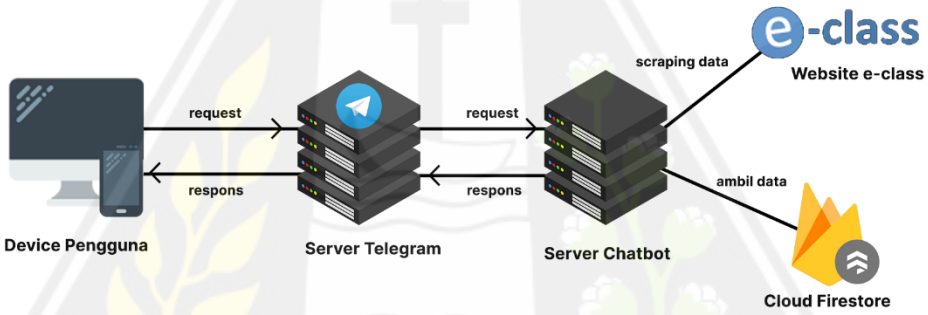
Proses lain yang dapat dilakukan pada *chatbot* ini adalah mengirimkan notifikasi ketika terdapat materi, pengumuman, tugas, nilai, diskusi, dan presensi baru di website e-class. Tahapannya adalah dengan melakukan *scraping* data pada e-class secara periodik menggunakan bantuan dari *library* puppeteer. Server kemudian juga akan mengambil data yang ada pada Cloud Firestore. Kedua data tersebut kemudian akan dibandingkan. Saat mendeteksi ada perubahan data, *server* akan langsung mengirimkan notifikasi berupa data yang berubah kepada pengguna tersebut.

Untuk menguji *chatbot* yang dibuat dapat berjalan sesuai dengan rencana maka akan dilakukan uji *User Acceptance Testing* (UAT). Salah satu metode yang ada dalam UAT adalah *Alpha/Beta Testing*. Metode *Alpha/Beta Testing* dipilih karena memungkinkan untuk menguji produk (*chatbot*) dengan pengguna sesungguhnya dalam kondisi seperti di lapangan. Pengujian ini akan dilakukan

dengan melibatkan 30 mahasiswa dari berbagai program studi. Pada proses pengujian, mahasiswa akan menjalankan beberapa tugas yang diberikan. Tugas tersebut seperti login, logout, melihat materi, pengumuman, tugas, diskusi, presensi dan nilai serta menerima notifikasi saat terdapat materi, tugas, nilai, diskusi dan pengumuman baru. Hasil dari setiap tugas tersebut kemudian akan dievaluasi untuk menentukan fitur tersebut dapat berjalan sesuai rencana atau tidak.

3.2 Arsitektur Sistem

Arsitektur dari sistem *chatbot* yang dikembangkan dapat dilihat pada gambar dibawah ini.



Gambar 3.1. Arsitektur Sistem

Gambar 3.1 merupakan arsitektur sistem *chatbot* yang dikembangkan dalam penelitian kali ini. Secara sederhana, pengguna akan terhubung ke dalam sistem menggunakan antarmuka *chatbot* melalui aplikasi Telegram yang terdapat pada perangkat masing-masing pengguna. *Server telegram* terhubung dengan *server chatbot* menggunakan metode *webhook.* Metode ini memungkinkan dua sistem untuk berkomunikasi menggunakan protokol HTTP. Khusus untuk *server telegram,* protokol HTTP yang digunakan harus memiliki SSL/TLS.

Server chatbot ditanamkan pada sebuah URL yang terenkripsi dan kemudian dikaitkan dengan *server telegram. Server chatbot* akan terhubung dengan website eclass secara periodik pada proses *scraping. Server chatbot* juga akan terhubung dengan database *cloud firestore* seandainya data yang diminta oleh pengguna sudah terdapat pada database.

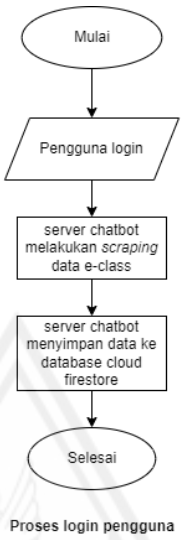
Perangkat dari pengguna akan terhubung dengan *server telegram* menggunakan antarmuka *telegram chatbot.* Segala *request* dari pengguna akan diterima dan diolah oleh *server telegram.* Selanjutnya *request* tersebut akan dikirimkan kepada *server chatbot* yang sudah dikembangkan. *Request* tersebut akan diolah terlebih dahulu untuk menentukan respon yang akan dikirimkan kembali kepada pengguna.

Pada *request* berupa login, *server chatbot* akan melakukan proses *scraping* pada website eclass dengan menggunakan *library puppeteer.* Sistem akan mengakses website eclass menggunakan NIM dan *password* yang sudah dikirimkan oleh pengguna. Saat website eclass berhasil diakses, sistem akan menyimpan informasi mata kuliah seperti (kode nama kuliah, nama mana kuliah, grup, sks, jadwal, ruang, dan dosen pengampu) ke dalam *database*. Selanjutnya sistem akan melakukan *scraping* untuk setiap menu pada mata kuliah tersebut. Menu diakses dari menu materi, pengumuman, tugas, nilai, presensi, dan yang terakhir diskusi. Hasil dari proses *scraping* tersebut kemudian akan disimpan pada *cloud firestore.*

Sedangkan pada *request* lainnya, *server chatbot* hanya akan mengambil data yang sudah tersedia pada *cloud firestore* untuk dikirimkan kembali kepada pengguna.

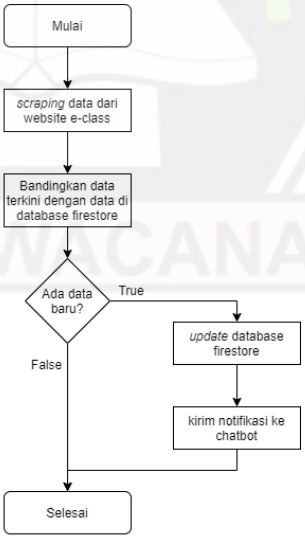
3.3 Diagram Alir

Terdapat tiga proses utama yang dapat dilakukan oleh *chatbot* yang dikembangkan dalam penelitian ini. Ketiga proses tersebut adalah melakukan proses login, melakukan pengolahan pilihan menu yang tersedia, dan melakukan *scraping* data eclass secara berkala. Diagram alir untuk setiap proses tersebut dapat dilihat pada gambar dibawah ini.



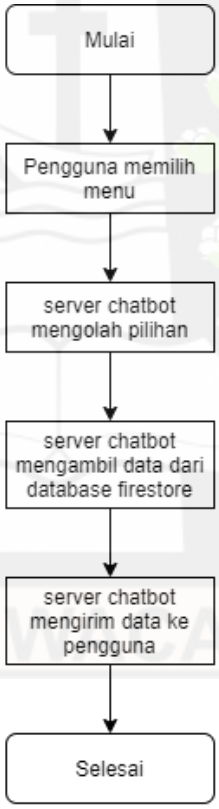
Gambar 3.2. Diagram alir proses login pengguna

Gambar 3.2 merupakan diagram alir proses login pengguna saat akan menggunakan *chatbot*. Pengguna akan memberikan *username* berupa NIM dan *password* yang digunakan untuk mengakses website eclass. *Server chatbot* akan melakukan proses *scraping* data eclass menggunakan NIM dan *password* yang sudah diberikan oleh pengguna. Langkah selanjutnya adalah *server chatbot* akan menyimpan data hasil *scraping* dan juga data informasi login pengguna tersebut ke dalam database *cloud firestore.*



Gambar 3.3. Diagram alir proses pengambilan data dan pengiriman notifikasi

Gambar 3.3 menunjukkan proses pengambilan data dan pengiriman notifikasi. Proses ini dimulai dengan melakukan *scraping* data dari website e-class menggunakan *username* dan *password* yang sudah ada pada *Cloud Firestore*. Data yang diperoleh pada tahap ini kemudian dibandingkan dengan data yang sudah ada pada Cloud Firestore. Data yang dibandingkan meliputi materi, pengumuman, tugas, nilai, diskusi, dan presensi. Jika ditemukan perbedaan (terdapat data baru) maka *server* akan memperbarui data pada Cloud Firestore. Setelah itu *server* akan mengirimkan notifikasi kepada pengguna terkait perubahan yang terjadi. Jika pada proses ini tidak ditemukan perbedaan maka *server* tidak akan memperbarui dan mengirimkan notifikasi kepada pengguna. Proses ini dilakukan secara periodik pada jam perkuliahan untuk semua pengguna yang sudah terdaftar pada database.

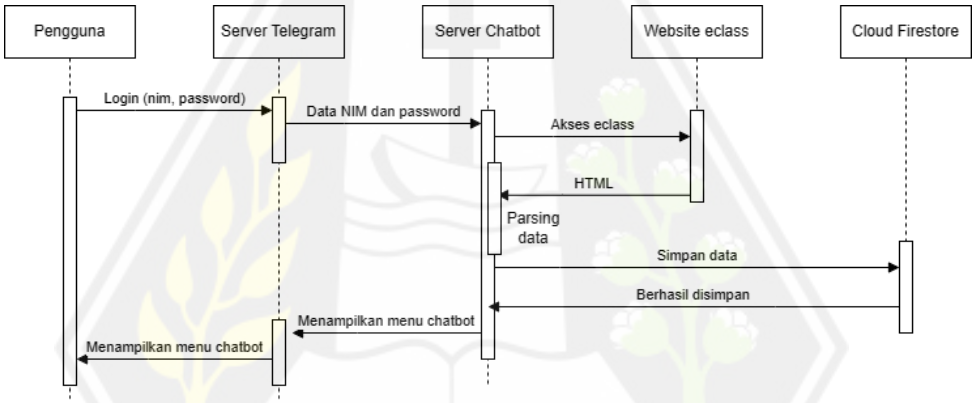


Gambar 3.4. Diagram alir proses pengolahan pilihan pada menu chatbot

Gambar 3.4 menjelaskan diagram alir proses pengolahan pilihan pada menu *chatbot*. Proses ini dimulai setelah pengguna memilih salah satu menu yang tersedia pada *chatbot*. *Server chatbot* akan mengolah pilihan yang dikirimkan pengguna. Pilihan tersebut bisa berupa aksi untuk melihat materi pada salah satu mata kuliah. Proses selanjutnya adalah *server chatbot* akan mengambil data dari Cloud Firestore sesuai dengan data yang diminta oleh pengguna. *Server* kemudian akan mengirimkan data tersebut kepada pengguna.

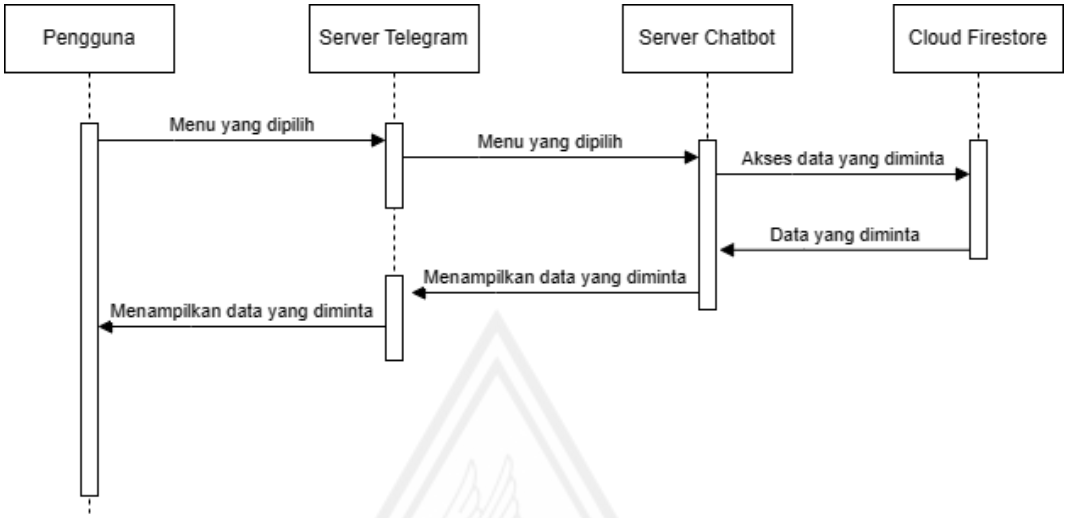
3.4 Sequence Diagram

Sequence diagram untuk proses utama pada chatbot dapat dilihat pada gambar berikut:



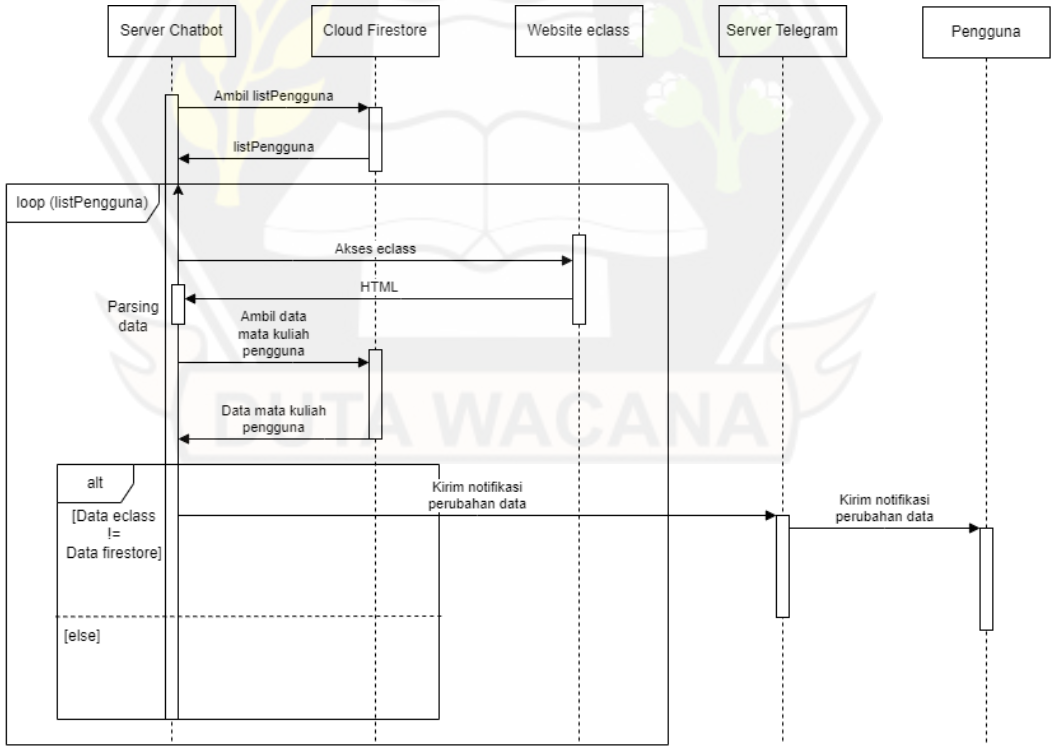
Gambar 3.5. *Sequence diagram* proses login

Gambar 3.5 menampilkan *sequence diagram* pada proses login. Dapat dilihat bahwa koneksi antara *server telegram* dan *server chatbot* hanya terjadi saat mengirimkan data login dan juga saat menampikan menu chatbot sesudah login berhasil. Koneksi antara website eclass dan *server chatbot* juga hanya terjadi sewaktu proses *scraping* data saja. Selanjutnya *server chatbot* akan menyimpan data tersebut dalam database *cloud firestore.*



Gambar 3.6. *Sequence diagram* proses pengolahan menu

Gambar 3.6 menampilkan *sequence diagram* pada proses pengolahan menu *chatbot*. Data mata kuliah yang diminta oleh pengguna diambil dari database *cloud firestore* sehingga tidak perlu melakukan proses *scraping* setiap kali pengguna ingin mendapatkan data pada eclass.

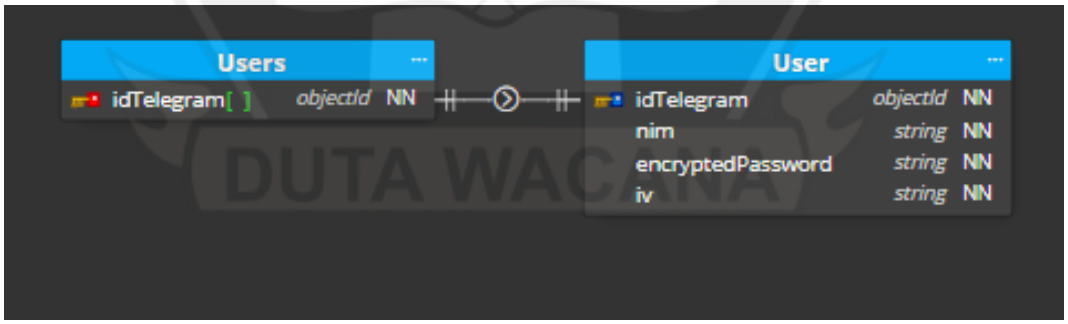


Gambar 3.7. *Sequence Diagram* proses *scraping* berkala

Gambar 3.7 menampilkan *sequence diagram* untuk proses *scraping* secara berkala yang dilakukan oleh *server chatbot.* Proses ini dilakukan dengan interval tiga jam. Langkah pertama *server chatbot* akan mengambil listPengguna dari database. Kemudian untuk setiap pengguna, akan dilakukan pengaksesan akun eclass. Selanjutnya *server chatbot* akan mengambil data pengguna tersebut pada database. Kedua data tersebut kemudian dibandingkan. Jika terdapat perbedaan, maka *server chatbot* akan memberikan notifikasi kepada pengguna melalui bantuan *server telegram*. Namun jika tidak terdapat perbedaan, *server chatbot* akan lanjut ke akun pengguna selanjutnya. Untuk membandingkan data, *server chatbot* akan mengecek hasil hash dari kedua data yang sudah dimiliki.

3.5 Perancangan Basis Data

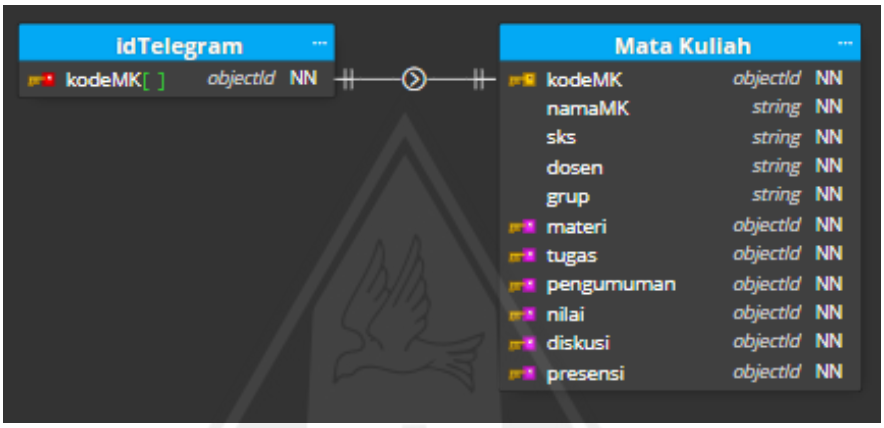
Basis data yang akan digunakan dalam penelitian adalah basis data non relasional atau *NoSQL.* Basis data jenis *NoSQL* memungkinkan pengguna menyimpan data dalam bentuk pasangan nilai-kunci (*key-value*) atau berbentuk dokumen seperti JSON. Penelitian ini membagi basis data yang digunakan ke dalam dua buah koleksi besar yaitu koleksi *users* dan koleksi unik untuk setiap *user.* Koleksi *users* merupakan sebuah koleksi berisi daftar pengguna chatbot. Sedangkan koleksi unik setiap *user* merupakan sebuah koleksi berisi daftar mata kuliah yang sedang diambil oleh *user*.



Gambar 3.8. Rancangan koleksi users

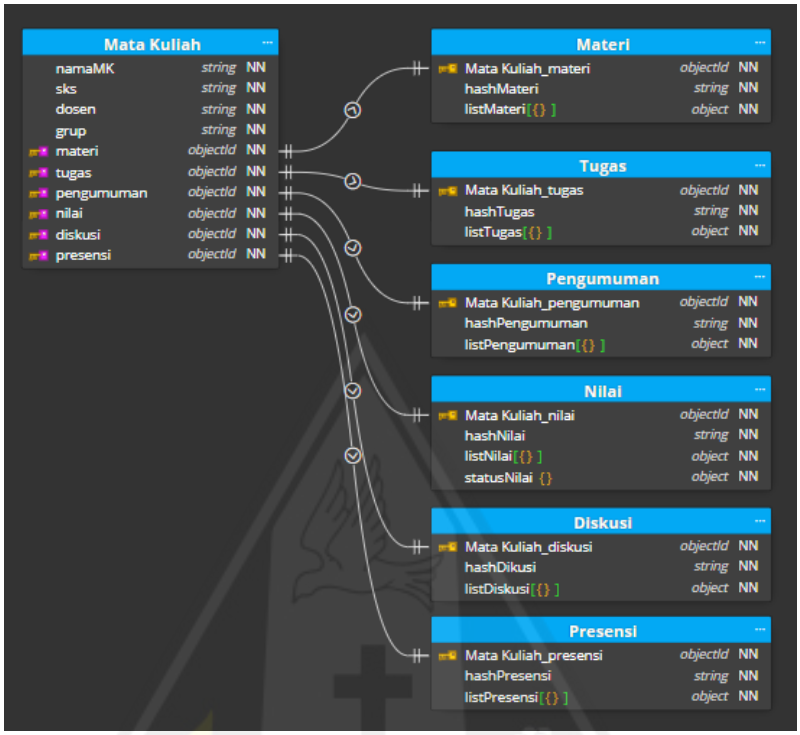
Gambar 3.8 merupakan rancangan koleksi *users* yang akan disimpan di *Cloud Firestore.* Koleksi *users* terdiri dari daftar (list) pasangan nilai-kunci untuk setiap

user chatbot. Kunci yang digunakan dalam list tersebut dalah *id* telegram, sedangkan nilainya adalah sebuah *object* berisikan nim, *encrypted password* eclass dari pengguna, dan *iv (initialization vector)*.



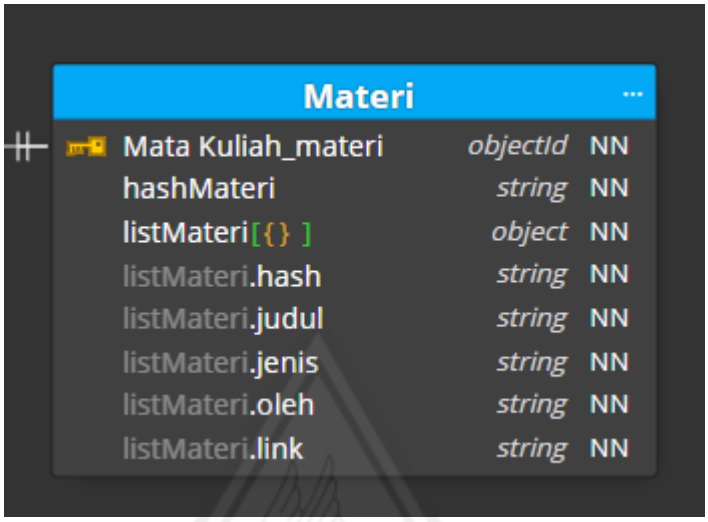
Gambar 3.9. Rancangan koleksi setiap user

Gambar 3.9 merupakan rancangan dari koleksi unik setiap *user.* Untuk setiap *id telegram* yang berada di dalam list koleksi *users,* akan memiliki koleksi sendiri untuk menyimpan data akademik eclass miliknya. Koleksi ini disimpan dengan nama *id telegram* dari pengguna. Di dalam koleksi ini terdapat daftar kode mata kuliah yang diambil. Kode mata kuliah ini merupakan kunci untuk mengakses *object* mata kuliah tersebut.



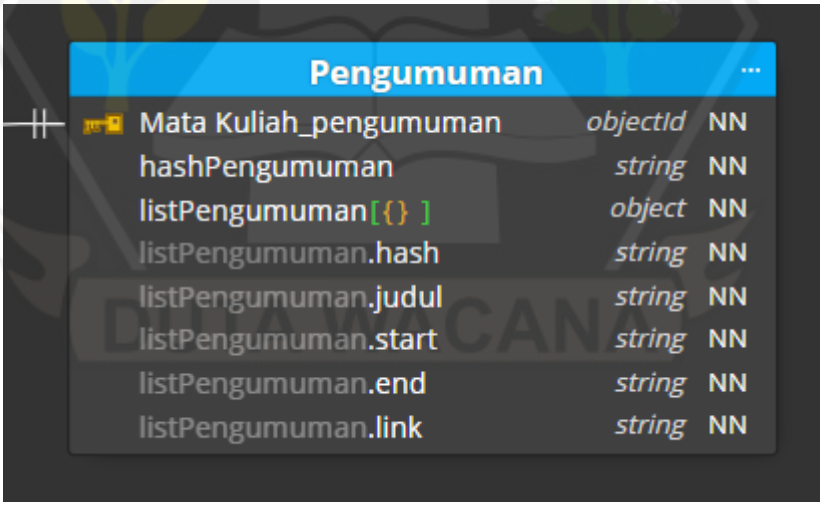
Gambar 3.10. Rancangan dokumen mata kuliah milik user

Gambar 3.10 merupakan penjelasan rancangan dokumen mata kuliah untuk setiap *object* mata kuliah pada Gambar 3.9. Dokumen mata kuliah milik *user* terdiri dari informasi mata kuliah yang disimpan dalam bentuk pasangan nilai-kunci seperti kode mata kuliah, nama, grup, sks, dosen pengampu dan jadwal. Terdapat enam kunci dalam dokumen ini yang memiliki nilai sebuah *object* yaitu materi, pengumuman, tugas, nilai, presensi, dan diskusi.



Gambar 3.11. Rancangan field materi mata kuliah

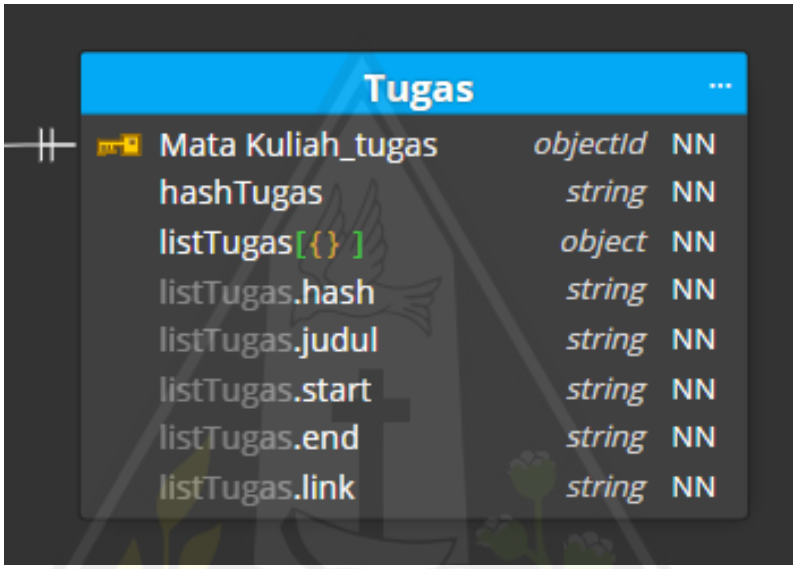
Gambar 3.11 merupakan rancangan *field* materi pada Gambar 3.10. Rancangan ini terdiri dari hashMateri dan listMateri. HashMateri akan digunakan untuk membandingkan data baru. Jika hash berbeda maka terdapat data baru pada materi mata kuliah. Sedangkan listMateri merupakan sebuah list dari setiap *object* materi yang ada. *Object* tersebut terdiri dari hash, judul, jenis, oleh, dan link dari materi.



Gambar 3.12. Rancangan field pengumuman mata kuliah

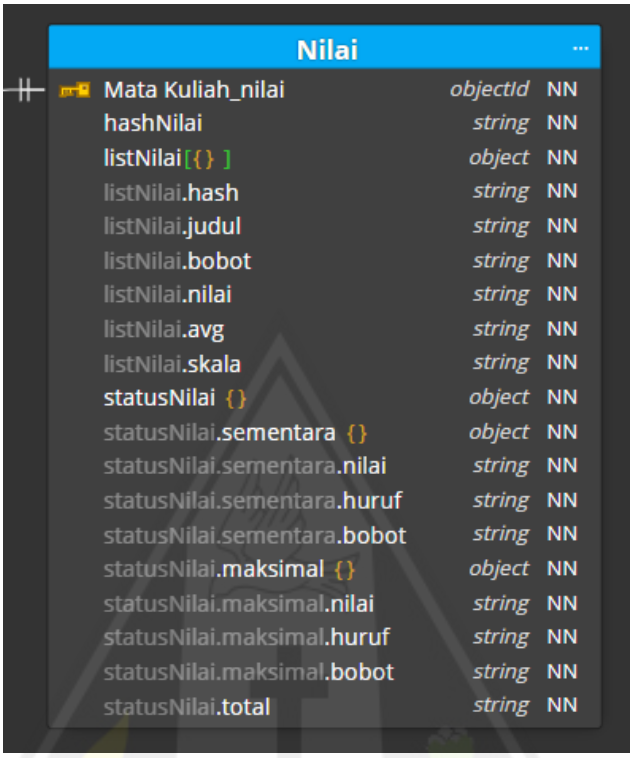
Gambar 3.12 merupakan rancangan *field* materi pada Gambar 3.10. Rancangan ini terdiri dari hashPengumuman dan listPengumuman.

HashPengumuman akan digunakan untuk membandingkan data baru. Jika hash berbeda maka terdapat data baru pada pengumuman mata kuliah. Sedangkan listPengumuman merupakan sebuah list dari setiap *object* pengumuman yang ada. *Object* tersebut terdiri dari hash, id, judul, *start*, *end*, dan link dari pengumuman.



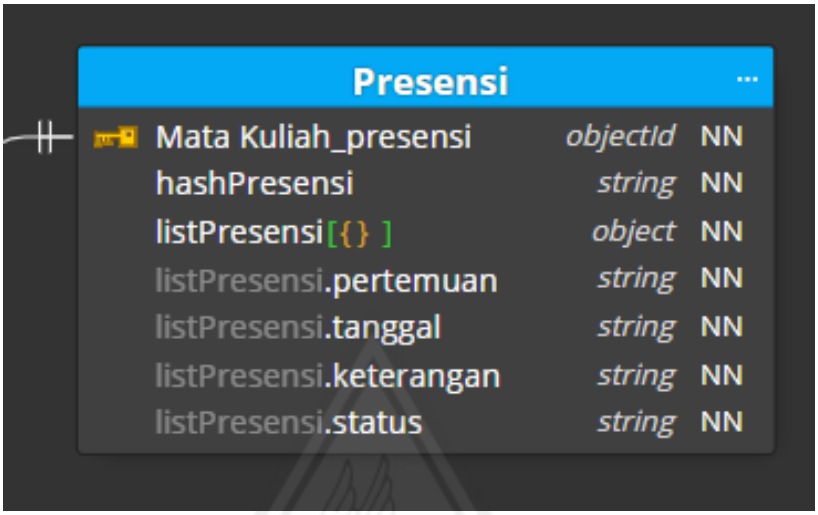
Gambar 3.13. Rancangan field tugas mata kuliah

Gambar 3.13 merupakan rancangan *field* materi pada Gambar 3.10. Rancangan ini terdiri dari hashTugas dan listTugas. HashTugas akan digunakan untuk membandingkan data baru. Jika hash berbeda maka terdapat data baru pada tugas mata kuliah. Sedangkan listTugas merupakan sebuah list dari setiap *object* tugas yang ada. *Object* tersebut terdiri dari hash, judul, *start*, *end*, dan link dari tugas.



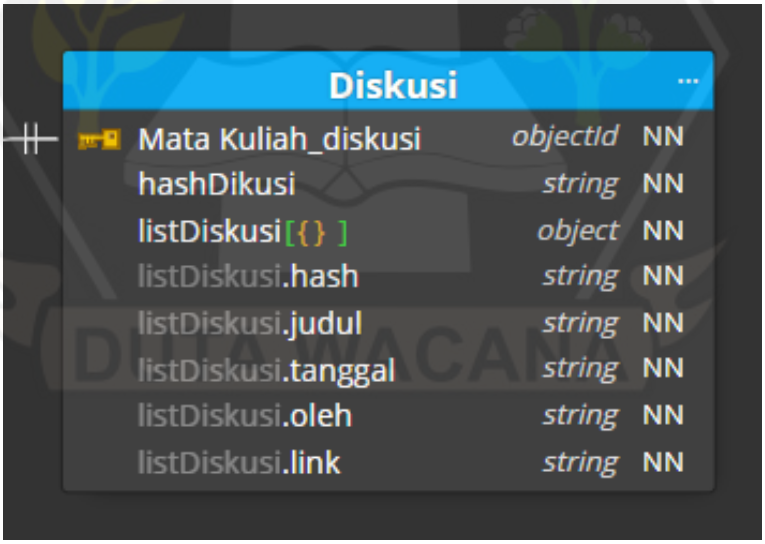
Gambar 3.14. Rancangan field nilai mata kuliah

Gambar 3.14 merupakan rancangan *field* materi pada Gambar 3.10. Rancangan ini terdiri dari hashNilai, listNilai, dan statusNilai. HashNilai akan digunakan untuk membandingkan data baru. Jika hash berbeda maka terdapat data baru pada nilai mata kuliah. Sedangkan listNilai merupakan sebuah list dari setiap *object* nilai yang ada. *Object* tersebut terdiri dari hash, judul, bobot, nilai, *avg*, skala, dan link dari nilai. Terakhir terdapat statusNilai yang digunakan untuk menyimpan informasi total item yang sudah dinilai, capaian sementara (bobot, nilai, dan huruf) serta capaian maksimal (bobot, nilai, dan huruf).



Gambar 3.15. Rancangan field presensi mata kuliah

Gambar 3.15 merupakan rancangan *field* presensi pada Gambar 3.10. Rancangan ini terdiri dari hashPresensi dan listPresensi. HashPresensi akan digunakan untuk membandingkan data baru. Jika hash berbeda maka terdapat data baru pada presensi mata kuliah. ListPresensi merupakan sebuah list dari setiap *object* presensi yang ada. *Object* tersebut terdiri dari tanggal, pertemuan, keterangan dan status kehadiran pada pertemuan tersebut.



Gambar 3.16. Rancangan field diskusi mata kuliah

Gambar 3.16 merupakan rancangan *field* diskusi pada Gambar 3.10. Rancangan ini terdiri dari hashDiskusi dan listDiskusi. HashDiskusi akan digunakan untuk membandingkan data baru. Jika hash berbeda maka terdapat data

baru pada diskusi mata kuliah. Sedangkan listDiskusi merupakan sebuah list dari setiap *object* diskusi yang ada. *Object* tersebut terdiri dari hash, id, oleh, judul, tanggal, dan link dari diskusi.

3.6 Perancangan Antarmuka Pengguna

3.6.1 Rancangan antarmuka *login*

Gambar 3.17. Rancangan antarmuka *login*

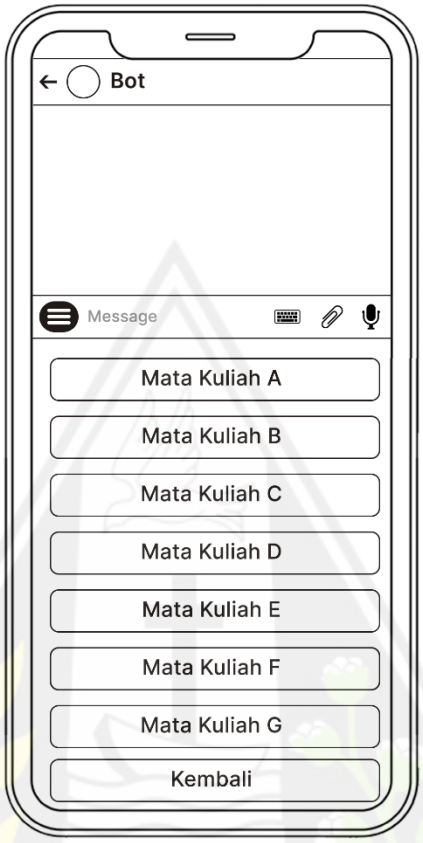
Gambar 3.17 merupakan rancangan antarmuka untuk tampilan awal chatbot sebelum pengguna melakukan proses login. Terdapat dua menu yang disediakan yaitu menu login dan menu bantuan. Menu login akan memunculkan pesan untuk mengirimkan *username* dan *password* e-class dari pengguna. Sedangkan menu bantuan akan mengirimkan pesan berupa informasi fitur-fitur yang terdapat dalam chatbot.

3.6.2 Rancangan antarmuka menu awal

Gambar 3.18. Racangan antarmuka menu awal

Gambar 3.18 merupakan rancangan awal untuk tampilan menu awal pada chatbot yang dikembangkan. Terdapat tiga buah menu yaitu daftar mata kuliah, bantuan, dan logout. Tombol daftar mata kuliah akan mengirimkan balasan menu baru berupa daftar mata kuliah seperti yang ditampilkan pada Gambar 3.19. Menu bantuan akan mengirimkan balasan berupa pesan berisi cara menggunakan chatbot. Menu logout akan memunculkan perintah untuk mengakhiri penggunaan *chatbot*.

3.6.3 Rancangan antarmuka menu daftar mata kuliah



Gambar 3.19. Rancangan antarmuka menu daftar mata kuliah

Gambar 3.19 merupakan rancangan awal dari tampilan daftar mata kuliah. Pada tampilan ini, jumlah menu akan mengikuti banyaknya mata kuliah yang sedang diambil oleh pengguna. Sebagai contoh pengguna mengambil tujuh mata kuliah maka aka nada tujuh menu untuk setiap mata kuliah yang pengguna ambil dan satu menu untuk kembali ke tampilan awal dari chatbot. Menu untuk setiap mata kuliah yang ditampilkan akan memberikan pengguna pilihan untuk mengakses materi, tugas, pengumuman dan nilai seperti yang ditampilkan pada Gambar 3.20.

3.6.4 Rancangan antarmuka menu mata kuliah

Gambar 3.20. Rancangan antarmuka menu mata kuliah untuk daftar materi

Gambar 3.20 merupakan rancangan awal dari tampilan ketika salah satu mata kuliah pada Gambar 3.19 dipilih. Pada tampilan ini terdapat lima buah menu. Menu yang pertama adalah menu untuk mengakses daftar materi. Gambar 3.20 merupakan contoh ketika pengguna mencoba mengakses daftar materi untuk mata kuliah A. Menu yang kedua adalah untuk mengakses daftar tugas. Menu pengumuman untuk mengakses pengumuman dan menu nilai untuk mengakses nilai dari mata kuliah tersebut. Menu terakhir adalah menu kembali. Fungsinya adalah untuk kembali ke halaman daftar mata kuliah seperti yang terdapat pada Gambar 3.19.

3.6.5 Rancangan notifikasi

Gambar 3.21. Rancangan notifikasi

Gambar 3.21merupakan rancangan awal untuk notifikasi dari chatbot yang dikembangkan. Terdapat dua bagian yang akan ditampilkan pada pesan notifikasi ini. Bagian pertama adalah informasi mengenani jenis item yang mengalami perubahan (materi, pengumuman, tugas, dan nilai) pada mata kuliah. Bagian kedua adalah isi dari item tersebut.

3.7 Perancangan Pengujian Sistem

Pengujian sistem pada penelitian ini akan menggunakan pengujian *User Acceptance Testing* (UAT). Metode yang akan digunakan adalah *Alpha/Beta Testing.* Metode ini akan dilakukan dalam dua tahap yaitu *Alpha Testing* dan *Beta Testing.* Pada tahap pertama yaitu *Alpha Testing,* pengujian *chatbot* akan dilakukan dengan di sisi pengembang oleh perwakilan dari pengguna. Pengujian ini dilakukan

dalam kondisi lingkungan yang dikendalikan untuk mengetahui fungsionalitas fitur *chatbot*. Rencana pengujian dapat dilihat pada Tabel 3.1.

Tabel 3.1. Rancangan Pengujian *Alpha Testing*

No	Fitur	Skenario	Hasil yang diharapkan
1	*Login*	Melakukan *login*	Muncul pesan sedang login mohon menunggu diikuti dengan pesan Selamat Datang di Bot Eclass
		Melakukan *login* dengan username atau password salah	Muncul pesan *username* atau *password* salah
		Melakukan login, saat sudah *login*	Muncul pesan silakan *logout* terlebih dahulu
2	Daftar mata kuliah	Melihat menu daftar mata kuliah	Muncul pesan list mata kuliah beserta menu untuk setiap mata kuliah yang ada
		Melihat menu daftar mata kuliah, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu
3	Info mata kuliah	Melihat menu info mata kuliah	Muncul pesan informasi berupa kode, nama, grup, dosen pengampu, jadwal, dan ruang dari mata kuliah
		Melihat menu info mata kuliah, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu
		Melihat menu info mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan
4	Materi	Melihat menu materi	Jika ada materi, muncul pesan berupa judul, jenis, pembuat, dan link dari setiap materi yang ada
			Jika tidak ada materi, muncul pesan belum ada materi

No	Fitur	Skenario	Hasil yang diharapkan
		Melihat menu materi, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu
		Melihat menu materi mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan
5	Pengumuman	Melihat menu pengumuman	Jika ada pengumuman, muncul pesan berupa judul, tanggal mulai, tanggal berakhir dan link dari setiap pengumuman yang ada
			Jika tidak ada pengumuman, muncul pesan belum ada pengumuman
		Melihat menu pengumuman, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu
		Melihat menu pengumuman mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan
6	Tugas	Melihat menu tugas	Jika ada tugas, muncul pesan berupa judul, tanggal mulai, tanggal berakhir dan link dari setiap tugas yang ada
			Jika tidak ada tugas, muncul pesan belum ada tugas
		Melihat menu tugas, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu
		Melihat menu tugas mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan
7	Nilai	Melihat menu nilai	Jika ada nilai, muncul pesan informasi nilai sementara dan nilai maksimal dan pesan berupa judul, bobot,

No	Fitur	Skenario	Hasil yang diharapkan
			nilai, rata-rata, skala dan link dari setiap nilai yang ada
			Jika tidak ada nilai, muncul pesan belum ada nilai
		Melihat menu nilai, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu
		Melihat menu nilai mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan
8	Diskusi	Melihat menu diskusi	Jika ada diskusi, muncul pesan berupa judul, tanggal, pembuat dan link dari setiap diskusi yang ada
			Jika tidak ada diskusi, muncul pesan belum ada diskusi
		Melihat menu diskusi, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu
		Melihat menu diskusi mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan
9	Presensi	Melihat menu presensi	Jika ada presensi, muncul pesan berupa judul, tanggal, materi, pembuat dan status kehadiran dari setiap presensi yang ada
			Jika tidak ada presensi, muncul pesan belum ada presensi
		Melihat menu presensi, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu
		Melihat menu presensi mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan

No	Fitur	Skenario	Hasil yang diharapkan
10	Notifikasi	Terdapat data baru di eclass	Muncul pesan informasi data yang baru
11	*Logout*	Melakukan *Logout*	Muncul pesan Selamat Datang di Bot Eclass
		Melakukan *logout*, saat pengguna belum *login*	Muncul pesan silakan *login* terlebih dahulu

Kriteria penilaian ketercapaian yang akan digunakan dalam pengujian *Alpha Testing* adalah dengan menggunakan dua pilihan jawaban seperti pada Tabel 3.2.

Tabel 3.2. Kriteria Penilaian *Alpha Testing*

Ketercapaian	Nilai
Ya	1
Tidak	0

Penilaian tersebut kemudian dihitung dengan menggunakan rumus:

$$\textbf{Persentase} = \frac{\textbf{jumlah skor yang diperoleh}}{\textbf{jumlah skor maksimum}} \times \mathbf{100}\ \%$$

Nilai presentase ini kemudian akan digunakan untuk mengetahui seberapa besar ketercapaian *chatbot* yang dikembangkan.

Tahap berikutnya adalah dengan melakukan *Beta Testing.* Pengujian *Beta Testing* dilakukan dengan menggunakan responden dari berbagai program studi dalam kondisi lingkungan yang tidak dikendalikan oleh pengembang. Responden akan diberi beberapa tugas untuk mengetahui menu-menu yang ada dalam chatbot. Tugas tersebut dapat dilihat pada Tabel 3.3.

Tabel 3.3. Tugas Pengujian *Beta Testing*

No	Tugas
1	Melakukan login
2	Melihat daftar mata kuliah
3	Melihat info mata kuliah

No	Tugas
4	Melihat materi
5	Melihat tugas
6	Melihat pengumuman
7	Melihat nilai
8	Melihat diskusi
9	Melihat presensi
10	Melakukan logout

Pengukuran dilakukan dengan menggunakan kuesioner kepada responden. Kuesioner tersebut terdiri dari 10 pernyataan dengan lima pilihan jawaban yaitu Sangat Setuju (SS), Setuju (S), Netral (N), Tidak Setuju (TS), dan Sangat Tidak Setuju (STS). Kuesioner tersebut dapat dilihat pada tabel 3.4.

Tabel 3.4. Rancangan Kuesioner Pengujian *Beta Testing*

No	Pertanyaan
1	*Chatbot* ini mudah digunakan
2	Menu yang ada dalam chatbot ini mudah dipahami
3	Ikon yang digunakan dalam setiap menu mudah dipahami
4	Navigasi dalam *chatbot* ini mudah
5	Saya membutuhkan bantuan dari orang lain untuk menggunakan *chatbot* ini
6	Saya merasa menu yang ada dalam *chatbot* ini sudah berjalan dengan sesuai
7	Saya dapat menemukan apa yang saya cari dengan cepat di *chatbot* ini
8	Saya merasa respon *chatbot* sudah tepat dan cepat
9	Saya akan menggunakan *chatbot* ini lagi
10	Saya mungkin akan merekomendasikan *chatbot* ini ke teman atau kolega

Kriteria penilaian yang akan digunakan dalam pengujian *Beta Testing* adalah menggunakan skala *Likert* dengan lima pilihan jawaban seperti pada Tabel 3.5.

Tabel 3.5. Kriteria Penilaian *Beta Testing*

Kriteria	Nilai
Sangat Setuju	5
Setuju	4
Netral	3
Tidak Setuju	2
Sangat Tidak Setuju	1

Setelah data dari responden diperoleh, proses selanjutnya adalah menghitung persentase jawaban responden terhadap setiap elemen pada kuesioner. Penghitungan persentase dilakukan dengan menggunakan rumus:

$$\boldsymbol{Persentase} = \frac{\boldsymbol{jumlah\ skor\ yang\ diperoleh}}{\boldsymbol{jumlah\ skor\ maksimum}} \times \mathbf{100}\ \%$$

Data persentase setiap elemen tersebut kemudian perlu diolah kembali untuk mendapatkan persentase *Acceptance Index* dari pengujian *Beta Testing* menggunakan rumus:

$$\boldsymbol{Acceptance\ Index} = \frac{\boldsymbol{Rata - rata\ persentase\ setiap\ elemen}}{\boldsymbol{total\ elemen} \times \boldsymbol{skor\ maksimum}} \times \mathbf{100}\ \%$$

Data persentase *Acceptance Index* kemudian diinterpretasikan menggunakan bantuan Tabel 3.6.

Tabel 3.6. Pedoman Persentase *Acceptance Index*

Angka (%)	Kategori
0 - 20	Sangat Tidak Layak
21 - 40	Tidak Layak
41 - 60	Cukup
61 - 80	Layak
81 - 100	Sangat Layak

BAB IV
IMPLEMENTASI DAN PEMBAHASAN

4.1 Implementasi Sistem

4.1.1 *Setup server chatbot*

Server chatbot yang dikembangkan pada penelitian ini menggunakan metode *webhook*. Metode ini memiliki dua syarat yang wajib dipenuhi agar *chatbot* dapat berjalan sesuai dengan rencana yaitu:

1. Memiliki *URL*

   Server chatbot yang dikembangkan wajib ditanamkan pada sebuah *URL* yang dapat diakses oleh publik.

2. Memiliki sertifikat *SSL*

   Server chatbot yang ditanamkan pada sebuah *URL* wajib memiliki sertifikat *SSL*. Hal ini diperlukan agar semua komunikasi antara *server chatbot* dan *server telegram* terjadi dalam keadaan terenkripsi.

Terdapat beberapa cara untuk memenuhi kedua persyaratan tersebut. Namun dalam penelitian ini, cara yang dipilih adalah menggunakan *tunneling* dengan bantuan dari *ngrok.* Cara *tunneling* dengan *ngrok* dipilih karena memudahkan dalam pengaturan *server* (*setting domain, hostname, SSL,* dan lainnya). *Ngrok* akan melakukan proses *tunneling* agar *localhost* dari *server chatbot* dapat diakses oleh publik menggunakan *URL* dengan sertifikat *SSL*. Proses *setup* dari *ngrok* dapat dilihat pada Gambar 4.1.

```
setup:
    koneksi ke akun ngrok
    if (sukses):
        jalankan runBot
    else (gagal):
        setup
```

Gambar 4.1 *Pseudocode* proses *setup ngrok*

Proses *setup* akan dipanggil pertama kali saat sistem mulai dijalankan. Jika proses ini berhasil, maka akan menjalankan fungsi *runBot*. Jika proses ini gagal maka *server* akan menjalankan Kembali proses *setup*. Proses ini dapat berjalan berulang kali jika ternyata hasilnya selalu gagal. Pada tahap ini, sistem sudah dapat diakses oleh publik melalui *URL* acak yang disediakan oleh *ngrok*. Ketika publik mengakses *URL* tersebut, maka respons yang didapatkan sesuai dengan yang terjadi pada *localhost* milik *server* dengan *port* yang sama.

Proses selanjutnya adalah menjalankan fungsi *runBot.* Fungsi *runBot* merupakan fungsi utama pada sistem yang memuat semua fungsi untuk menangani *request* dari pengguna telegram. *Pseudocode* untuk fungsi *runBot* dapat dilihat pada Gambar 4.2.

```
runBot:
    launch(mengatur URL untuk webhook)
    start
    hears(menu 1)
    .
    .
    .
    hears (menu n)
```

Gambar 4.2 *Pseudocode* fungsi *runBot*

Pada saat fungsi *runBot* dipanggil, fungsi yang dijalankan pertama kali adalah fungsi *launch.* Pada fungsi *launch* ini terjadi proses pengkaitan *URL* yang sudah dibuatkan oleh *ngrok* kepada *server telegram.* Mulai pada proses ini, setiap *request* dari pengguna yang diterima oleh *server telegram* akan langsung diteruskan kepada *server chatbot* melalui *URL* buatan *ngrok.* Fungsi yang terdapat pada *runBot* dapat dikategorikan kedalam dua bagian besar yaitu

1. Fungsi *start*

   Fungsi ini akan dijalankan saat pengguna pertama kali berinteraksi dengan *chabot.* Fungsi *start* mengembalikan informasi tentang hal yang dapat dilakukan oleh *chatbot.*

2. Fungsi *hears*

   Fungsi *hears* akan dijalankan setelah fungsi *start.* Jumlah fungsi ini dapat disesuaikan dengan jumlah aksi yang dimiliki oleh *chatbot.* Fungsi *hears* akan menangkap sebuah *request* yang sesuai dengan parameter *trigger* yang sudah diatur sebelumnya. Sebagai contoh ketika pengguna ingin melakukan proses login, maka fungsi *hears* dengan parameter login saja yang akan dijalankan. Meskipun terdapat fungsi *hears* dengan parameter *trigger* lainnya seperti daftar mata kuliah, materi, tugas, logout dan lainnya.

4.1.2 Pengolahan Proses *Login*

Hal yang harus dilakukan oleh pengguna untuk dapat menggunakan *chatbot* ini adalah melakukan *login*. *Login* dilakukan dengan mengirimkan NIM mahasiswa sebagai *username* dan *password* eclass. Untuk *password* yang dikirimkan kepada *chatbot* akan dienkripsi sebagai bentuk keamanan dari data pengguna. Proses enkripsi dilakukan dengan menggunakan algoritma aes-256-cbc. *Pseudocode* untuk proses enkripsi dapat dilihat pada Gambar 4.3.

```
encrypt(password):
    key = fix key
    iv = random 16 Bytes
    encryptedPassword = createEncrypt(key)
    return {
        encryptedPassword,
        iv
    }
```

Gambar 4.3. *Pseudocode* proses enkripsi *password*

Pada saat fungsi enkripsi dipanggil, fungsi ini akan menggunakan *key* dan *initialization vector (iv)*. Untuk *key,* sistem akan menggunakan *key* yang sama untuk setiap pengguna, sedangkan untuk *initialization vector* akan dibuat secara acak untuk setiap pengguna. Data yang diperoleh dari proses ini adalah *password* yang

sudah terenkripsi dan juga *iv* dari setiap pengguna. Kedua data ini yang akan disimpan dalam *database cloud firestore.* Sedangkan *key* hanya akan disimpan pada *source code* program. Pemisahan *iv* dan *key* ini dilakukan untuk memberikan ruang keamanan jika terjadi kebocoran akses di *database cloud firestore.* Seandainya kebocoran akses tersebut terjadi, data yang diperoleh hanya *password* yang sudah terenkripsi dan juga *iv.* Namun jika *source code* program juga dapat diakses oleh pihak lain maka *password* dapat didekripsi secara utuh.

Permasalahan ini dapat diselesaikan dengan cara menyimpan *initialization vector* dan juga *key* untuk setiap pengguna dalam *memory* program. Artinya setiap *iv* dan *key* pengguna hanya dapat diakses oleh program saat *server* dinyalakan. Namun saat *server* dimatikan, semua data *iv* dan *key* pengguna akan dihapus. Cara ini belum diterapkan pada sistem karena pada saat pengujian, *server* akan berulang kali dinyalakan dan matikan.

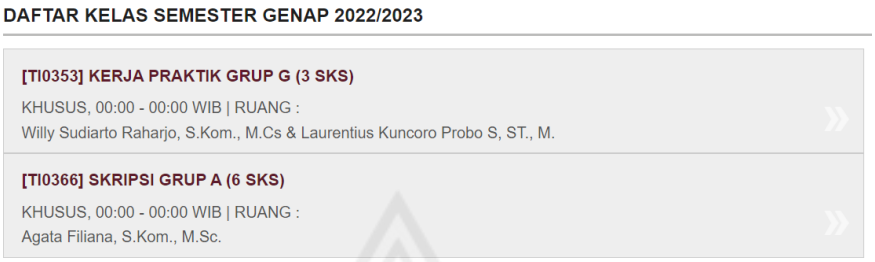
Setiap kali pengguna melakukan login (memanggil fungsi *hears*(*login*)), terjadi proses *scraping* data eclass pengguna untuk mendapatkan informasi akademiknya. *Pseudocode* untuk proses *scraping* dapat dilihat pada Gambar 4.4.

```
login:
    setup headless browser
    goto eclass.ukdw.ac.id
    input username
    input password
    do login
    get querySelectorAll(mata kuliah)
    save info mata kuliah to database
    scraping mata kuliah (async):
        save detail mata kuliah to database
    done
```

Gambar 4.4. *Pseudocode* proses *login*

Proses *scraping* dengan *library puppeteer* dimulai dengan melakukan *setup headless browser* yang akan digunakan. Setelah *setup* selesai, *Headless browser* akan membuka halaman baru dan mengunjungi alamat website eclass yaitu eclass.ukdw.ac.id. Setelah halaman berhasil dimuat, sistem akan mengisikan

username dan *password* dari pengguna dan melakukan proses login seperti biasanya.



Gambar 4.5. Contoh daftar mata kuliah setelah berhasil login

Pada Gambar 4.5 terlihat contoh daftar mata kuliah yang sedang diambil oleh pengguna saat ini. Kedua item kelas tersebut memiliki struktur html yang sama sehingga dapat dilakukan *query selector* untuk mendapatkan elemennya. Struktur html elemen tersebut dapat dilihat pada Gambar 4.6.

```
<div class="kelas_box"> == $0
  <h2>[TI0353] KERJA PRAKTIK GRUP G (3 SKS)</h2>
 ▶ <div class="kelas_arrow" style="float:right">…</div>
 ▼ <div class="kelas_box_sub">
    " KHUSUS, 00:00 - 00:00 WIB | RUANG : "
    <br>
    " Willy Sudiarto Raharjo, S.Kom., M.Cs & Laurentius Kuncoro Probo S, ST., M. "
  </div>
</div>
```

Gambar 4.6 Struktur HTML dari setiap kelas

Sistem akan langsung melakukan *query selector* untuk semua item mata kuliah yang ada pada eclass pengguna. Data yang diambil berupa kode mata kuliah, nama mata kuliah, grup, sks, ruangan, jadwal, dan dosen pengampu. Untuk data kode mata kuliah, nama mata kuliah, grup, dan sks diperoleh dari proses pembersihan data menggunakan fungsi *split.* Data tersebut akan disimpan dalam bentuk objek sebelum kemudian disimpan ke dalam database dalam bentuk koleksi.

Sistem juga akan membuat list kode mata kuliah yang berisikan setiap kode mata kuliah yang didapatkan pada proses *login.* List kode mata kuliah ini akan digunakan untuk proses *scraping* detail setiap mata kuliah secara *asynchronous.*

Proses pengambilan data mata kuliah pada saat *login* sudah selesai. Pengguna akan menerima pesan "Selamat datang di chatbot eclass" dan sudah dapat menggunakan chatbot tersebut.

Pada saat yang bersamaan, sistem akan melakukan proses *scraping* detail mata kuliah seperti materi, tugas, nilai hingga ke daftar presensi secara *asynchronous.* Proses ini menggunakan data dari list kode mata kuliah dan berlangsung dibelakang layar pengguna sehingga pengguna sudah dapat menggunakan *chatbot* meskipun informasi detail mata kuliah belum seluruhnya tersedia. Terdapat *handling* untuk *request* pengguna pada detail mata kuliah yang belum tersedia yaitu dengan mengirimkan pesan "Data X sedang diambil. Silakan tunggu…". Urutan proses *scraping* untuk detail mata kuliah dapat dilihat pada Gambar 4.7.

Gambar 4.7 *Pseudocode* proses *scraping* detail mata kuliah

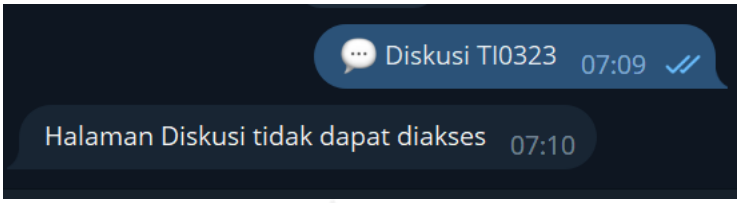
Alur pengambilan data pada setiap mata kuliah milik pengguna dilakukan mulai dari materi hingga yang terakhir adalah diskusi. Fungsi *scraping* mata kuliah ini dipanggil secara secara *asynchronous* untuk setiap kode yang terdapat pada list kode mata kuliah. Fungsi pengambilan data materi hingga data diskusi memiliki struktur yang serupa meskipun elemen yang diambil berbeda. Fungsi tersebut dapat dilihat pada Gambar 4.8.

```
getMateri(kode mata kuliah):
    res = {
            status: false,
            message: "",
            hashMateri: "",
            listMateri: []
        }
    respon = goto halaman mata kuliah
    if respon ok (status code 200-299):
        dataMateri = querySelectorAll()
        dataHash = createHash()
        if dataMateri empty:
            res.status: true,
            res.message: "Belum ada materi"
        else:
            res.status: true,
            res.message: "Sudah ada materi"
            for materi in dataMateri:
                data.judul = ambil judul materi
                data.oleh = ambil pengunggah materi
                data.jenis = ambil jenis materi
                data.link = ambil link url materi
                data.hash = createHash(materi)
                push data to listMateri
    else:
        res.status: false,
        res.message: "Halaman materi tidak dapat
diakses"
    return res
```

Gambar 4.8. *Pseudocode* proses *scraping* detail materi mata kuliah

Setiap pengambilan detail item (materi, tugas, dan lainnya) dari suatu mata kuliah hanya memerlukan kode mata kuliah tersebut. *URL* yang akan diakses untuk *scraping* hanya akan mengubah parameter item yang diakses saja. Contohnya untuk menu pengumuman *URL* yang diakses adalah eclass.ukdw.ac.id/e-class/id/kelas/pengumuman/TI0353. Sedangkan untuk tugas *URL* yang diakses adalah eclass.ukdw.ac.id/e-class/id/kelas/tugas/TI0353. Saat sistem mencoba mengakses halaman tersebut, sistem akan mendapatkan respon berupa dua hal yaitu respon *header* dan juga respon tampilan HTML dari halaman yang diakses. Sistem akan melakukan pengecekan terhadap *status code* yang diterima pada respon *header*. Jika *status code* yang diterima masuk dalam jangkauan 200-299 maka sistem akan melakukan *scraping*. Namun jika *status code* yang diterima diluar jangkauan, maka sistem akan menyimpan respon "Halaman Materi (item) tidak dapat diakses". Pesan ini akan muncul ketika pengguna memilih menu salah satu

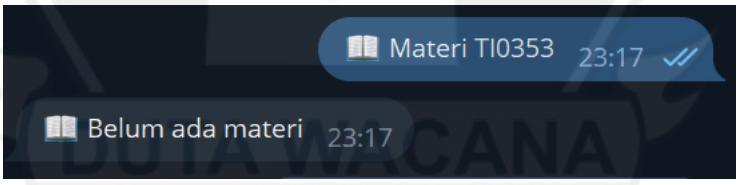
mata kuliah yang datanya tidak dapat diakses oleh *sistem* sewaktu proses *login.* Contohnya dapat dilihat pada Gambar 4.9.



Gambar 4.9. Respon *chatbot* untuk dapat yang tidak dapat diakses

Saat respon *header* yang diterima sistem masuk dalam jangkauan, sistem baru akan melakukan proses *scraping.* Proses ini dimulai dengan mengambil seluruh elemen yang dicari menggunakan fungsi *querySelectorAll* milik *javascript.* Bentuk *query* yang digunakan setiap item tentu sedikit berbeda, contohnya untuk item materi, querynya adalah *#content-right > table:nth-child(10) > tbody > tr.* Sedangkan untuk pengumuman bentuk querynya adalah *.thread.* Hal ini menyesuaikan dari struktur HTML setiap item yang ada pada website eclass. Selanjutnya sistem akan membuat *hash* secara keseluruhan untuk data tersebut. Nantinya *hash* ini akan digunakan sebagai perbandingan untuk mengetahui perubahan informasi.

Jika tidak terdapat elemen yang dikumpulkan melalui *querySelectorAll,* maka sistem akan menyimpan pesan "Belum ada Materi (item)". Pesan ini akan muncul ketika pengguna mengakses menu tetapi data pada menu tersebut kosong. Contohnya dapat dilihat pada Gambar 4.10.



Gambar 4.10. Contoh belum ada materi

Namun jika elemen yang dicari ternyata ditemukan pada halaman tersebut, maka sistem akan melakukan pembersihan data untuk setiap elemen yang ditemukan. Data HTML tersebut perlu dibersihkan terlebih dahulu, karena sistem hanya akan menyimpan teks saja, tanpa kode HTMLnya. Contohnya untuk materi, sistem akan menyimpan data berupa judul, oleh (pengunggah), jenis, dan link *URL* ke materi tersebut. Data ini diperoleh dengan bantuan fungsi *innerText* pada

javascript. Data lain yang disimpan adalah *hash* dari elemen tersebut. Fungsi *hash* pada setiap elemen akan digunakan untuk mengetahui jika terjadi perubahan data pada data yang sudah pernah dipublikasikan pada item tersebut. Contohnya jika terjadi perubahan nilai (bukan nilai baru), maka sistem dapat memberikan notifikasi.

Setelah pengambilan data materi berhasil dilakukan, data tersebut akan langsung disimpan sesuai pada koleksi pengguna di database. Kemudian sistem akan melanjutkan pengambilan data untuk pengumuman dan seterusnya hingga diskusi. Terdapat dua menu mata kuliah yang belum diambil datanya pada sistem yaitu menu asisten dan juga menu peserta. Kedua menu ini tidak diikutsertakan karena dua pertimbangan yaitu data menu tersebut cenderung tidak berubah selama satu semester dan juga pengaruhnya terhadap waktu yang diperlukan hingga data lengkap.

Meskipun proses *scraping* sudah dilakukan secara paralel untuk mata kuliah pertama hingga terakhir, waktu yang diperlukan sampai semua data menjadi lengkap masih cukup lama. Waktu yang diperlukan ini sangat bergantung pada jumlah mata kuliah yang diambil pengguna dan juga koneksi (kelancaran) dari *server chatbot* saat mengunjuni website eclass. Masalah ini dapat diatasi dengan mengubah struktur kode agar menjadi *asynchronous* secara utuh. Struktur kode saat ini masih melakukan *loop* dari mata kuliah pertama hingga terakhir. Proses ini yang menyebabkan waktu tunggu hingga data lengkap menjadi lebih lama. Jika struktur kode sudah diubah tentunya dua menu (asisten dan peserta) dapat juga disertakan dalam *chatbot* tanpa memiliki pengaruh yang berarti terhadap waktu akses hingga data lengkap.

4.1.3 Pengolahan Menu *Chatbot*

Setiap menu yang terdapat pada *chatbot* akan terkoneksi dengan fungsi *hears* pada *server chatbot.* Semua fungsi *hears* memiliki struktur yang serupa meskipun memiliki *trigger* yang berbeda-beda. Struktur fungsi tersebut dapat dilihat pada Gambar 4.11.

```
hears (trigger):
    if userid exist in db:
        Ambil data yang diminta di db
        reply(data)
    else:
        reply(silakan login terlebih dahulu)
```

Gambar 4.11. Pseudocode fungsi hears

Di setiap fungsi *hears* akan selalu terdapat percabangan untuk mengecek *userid* dari pengguna sudah terdaftar pada *database* atau belum. Jika *userid* sudah terdaftar, maka *server* akan mengambil data yang diminta oleh pengguna di *database cloud firetore.* Data tersebut kemudian diformat sesuai sebagai teks untuk dikirimkan kepada pengguna. Namun jika *userid* tidak terdaftar maka, sistem akan mengirimkan pesan supaya pengguna melakukan *login* terlebih dahulu. Setiap *userid* pengguna akan disimpan selama pengguna belum melakukan proses *logout*.

4.1.4 Proses *Scraping* Secara Berkala

Proses lain yang juga dijalankan oleh sistem adalah proses *scraping* secara berkala untuk mencari tahu apakah terdapat perbedaan data milik pengguna di website eclass. *Pseudocode* proses *scraping* secara berkala dapat dilihat pada Gambar 4.12.

```
Scraping[berkala]:
    get listPengguna
    foreach pengguna:
        login
        get data from eclass
        get koleksi mata kuliah pengguna from database
        foreach koleksi mata kuliah:
            compare data eclass dan koleksi
            if beda:
                sendNotif
```

Gambar 4.12 *Pseudocode* proses *scraping* berkala

Proses ini akan berlangsung dengan interval tiga jam mulai dari pukul 06.00 hingga pukul 18.00 setiap hari. Proses ini mengambil data list pengguna dari

database. Kemudian untuk setiap pengguna pada list tersebut, sistem akan melalukan proses *login*. Setelah berhasil melakukan *login*, sistem akan mengambil data dari eclass tersebut. Proses ini seperti yang terdapat pada Gambar 4.7 dan Gambar 4.8. Selanjutnya sistem akan mengambil data untuk mata kuliah yang sama milik pengguna dari *database*. Kedua data tersebut kemudian dibandingkan untuk mencari tahu perubahan yang terjadi. Proses pembandingan menggunakan bantuan *hash* dari setiap data. Jika *hash* berbeda (terdapat perubahan data), maka sistem akan mengirimkan pesan informasi perubahan data tersebut kepada pengguna. Namun jika tidak terdapat perubahan data, sistem akan melanjutkan ke mata kuliah selanjutnya. Hal ini dilakukan untuk setiap detail mata kuliah dari setiap mata kuliah yang diambil pengguna. Setelah pengguna terakhir dalam listPengguna berhasil diperiksa, sistem akan mengakhiri proses *scraping* secara berkala ini.

4.1.5 Pengolahan Proses *Logout*

Proses terakhir yang dilakukan oleh *chatbot* adalah proses *logout.* Proses ini akan dijalankan ketika pengguna melakukan *logout* pada akunnya. *Pseudocode* untuk proses *logout* dapat dilihat pada Gambar 4.13.

```
logout(userid):
    if userid exist in db:
        delete user info
        delete user mata kuliah
        reply(Selamat datang di bot eclass)
    else:
        reply(Silakan login terlebih dahulu)
```

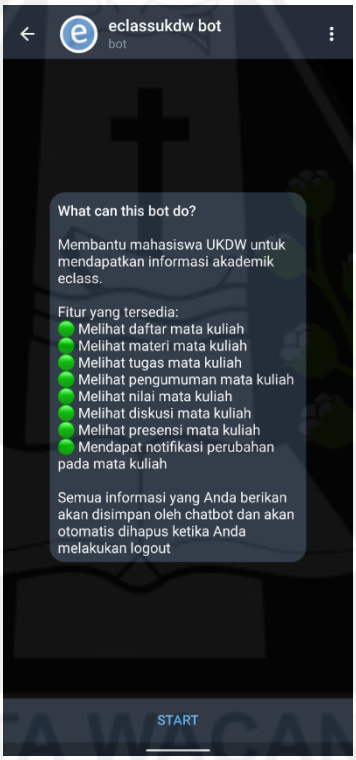
Gambar 4.13. *Pseudocode* proses *logout*

Pertama, sistem akan melakukan pengecekan *userid* dari pengguna. Jika ternyata *userid* belum disimpan dalam *database* (belum *login*), maka sistem akan mengirimkan balasan berupa pesan “Silakan login terlebih dahulu”. Namun jika *userid* pengguna sudah disimpan dalam *database* (sudah *login*), maka sistem akan melakukan koneksi ke *database cloud firestore* untuk menghapus informasi *login* pengguna. Info yang dihapus berupa *encryptedPassword, initialization vector,* dan juga NIM pengguna. Selanjutnya sistem akan menghapus koleksi data mata kuliah

pengguna yang disimpan pada *database.* Setelah proses penghapusnya tersebut berhasil dilakukan, maka sistem akan mengirimkan pesan “Selamat datang di bot eclass” kepada pengguna. Proses *logout* pengguna pada *chatbot* sudah berhasil dilakukan. Artinya semua data pengguna tersebut yang disimpan pada *server chatbot* sudah sepenuhnya dihapus. Sistem sudah tidak memiliki data apapun terkait pengguna tersebut. Pengguna dapat Kembali menggunakan *chatbot* dengan melakukan proses *login* kembali.

4.2 Implementasi Antarmuka

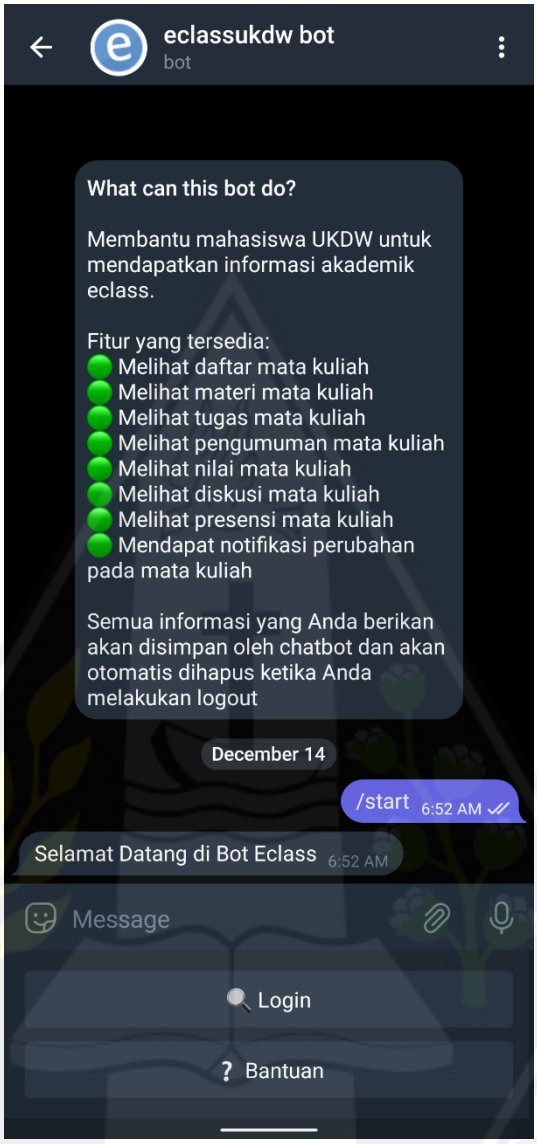
4.2.1 Antarmuka Awal Chatbot



Gambar 4.14. Antarmuka awal *chatbot*

Gambar 4.14 merupakan tampilan awal chatbot. Terdapat deskripsi chatbot, fitur-fitur yang tersedia pada chatbot dan juga informasi bahwa data yang diberikan pada chatbot akan disimpan sementara dan otomatis dihapus ketika pengguna melakukan logout.

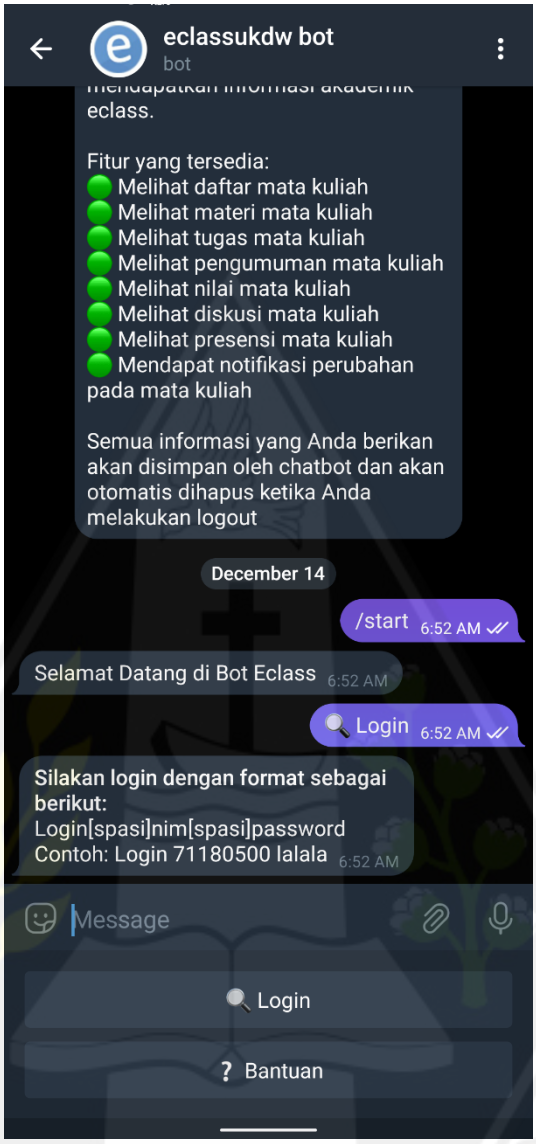
4.2.2 Antarmuka Menu Awal



Gambar 4.15. Antarmuka menu awal

Gambar 4.15 merupakan tampilan menu awal chatbot setelah pengguna menekan tombol *start* padaGambar 4.14. Pada tampilan ini terdapat dua buah menu yaitu login dan bantuan. Untuk menggunakan chatbot, pengguna perlu menekan menu login terlebih dahulu.

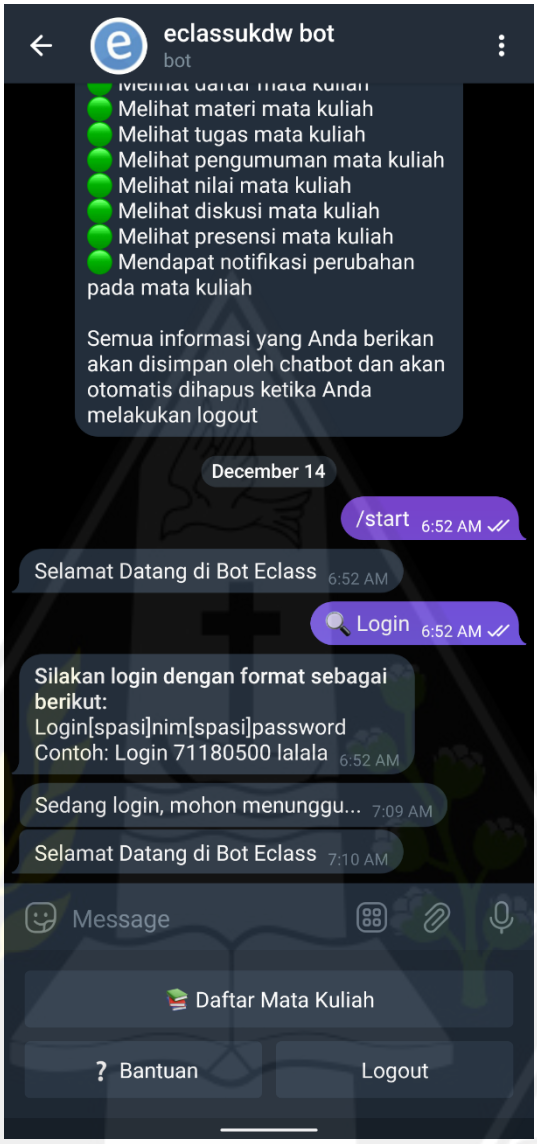
4.2.3 Antarmuka Login



Gambar 4.16. Antarmuka *login*

Gambar 4.16 merupakan informasi cara login yang harus dilakukan oleh pengguna jika ingin menggunakan chatbot. Format pesan yang diberikan oleh pengguna harus sesuai agar pengguna dapat login. Jika format yang diberikan oleh pengguna tidak sesuai maka akan muncul pesan login gagal.

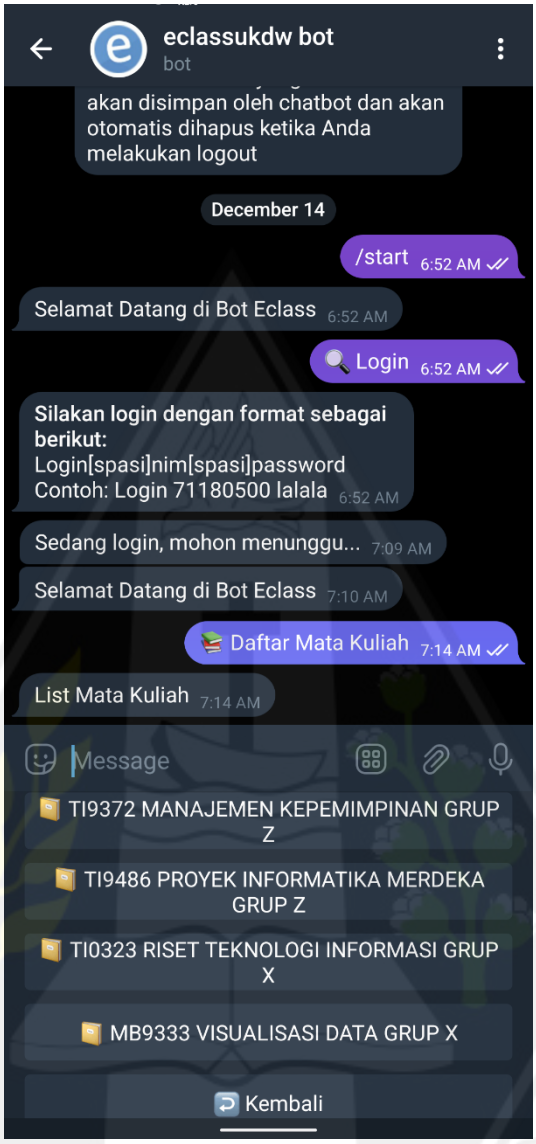
4.2.4 Antarmuka Menu Setelah Login



Gambar 4.17. Antarmuka menu setelah login

Gambar 4.17 adalah tampilan menu setelah pengguna berhasil melakukan login. Terdapat tiga buah menu yaitu daftar mata kuliah, bantuan, dan logout. Tombol daftar mata kuliah akan membawa pengguna ke menu mata kuliah yang diambil. Sedangkan menu logout dapat digunakan oleh pengguna jika ingin mengakhiri penggunaan *chatbot*.

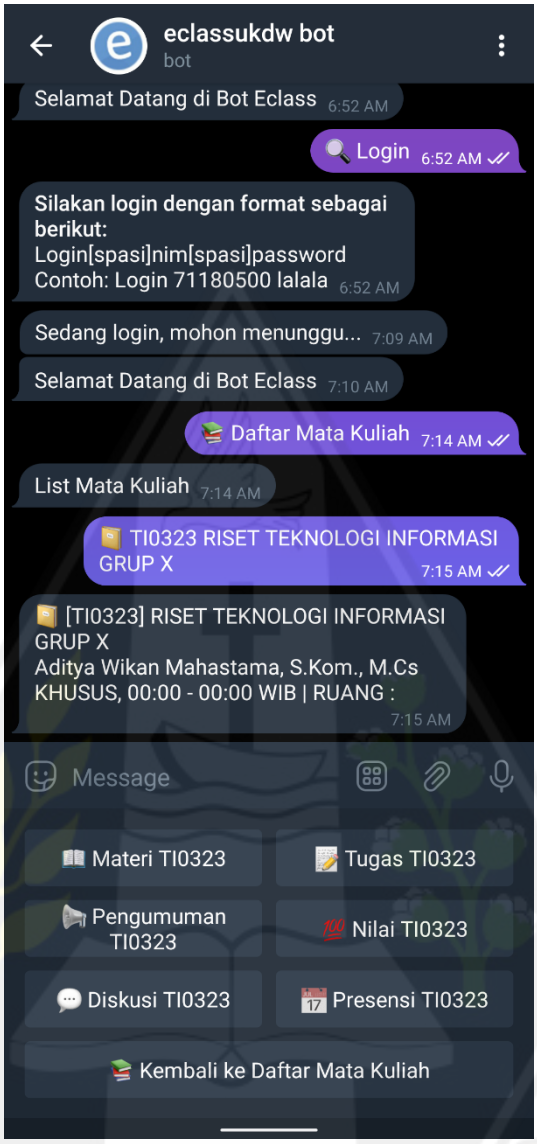
4.2.5 Antarmuka Daftar Mata Kuliah



Gambar 4.18. Antarmuka daftar mata kuliah

Gambar 4.18 adalah tampilan daftar mata kuliah. Tampilan ini berbeda-beda setiap pengguna, tergantung dari mata kuliah yang sedang diambil oleh pengguna. Untuk melihat informasi dari salah satu mata kuliah, pengguna perlu menekan nama mata kuliah yang ada.

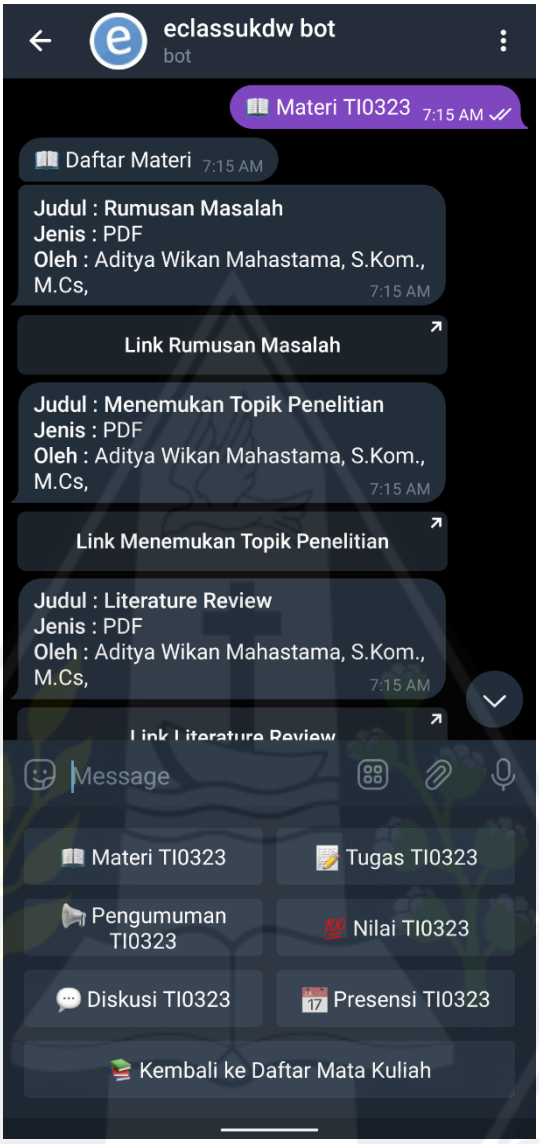
4.2.6 Antarmuka Menu Mata Kuliah



Gambar 4.19. Antarmuka menu mata kuliah

Gambar 4.19 merupakan tampilan menu mata kuliah yang sedang dipilih pengguna. Informasi yang dapat ditampilkan oleh chatbot ini berupa materi, tugas, pengumuman, nilai, diskusi, dan presensi. Untuk melihat mata kuliah lain pengguna perlu menekan menu kembali ke daftar mata kuliah.

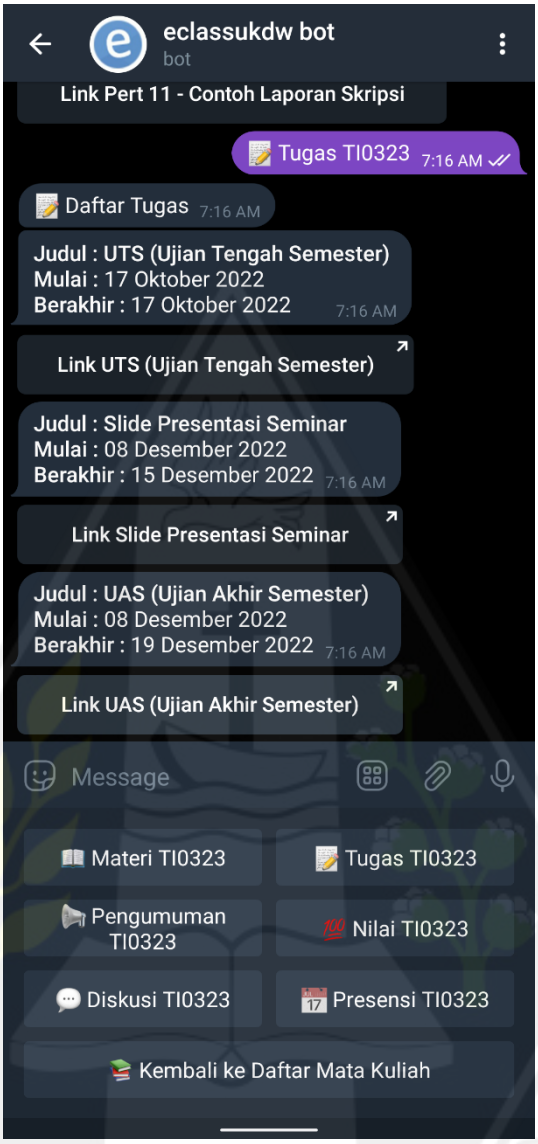
4.2.7 Antarmuka Menu Materi



Gambar 4.20. Antarmuka materi

Gambar 4.20 merupakan contoh balasan dari chatbot ketika pengguna menekan menu materi. Jika mata kuliah yang dipilih mempunyai materi, maka chatbot akan mengirimkan informasi materi untuk semua materi yang ada pada mata kuliah tersebut. Informasi yang dikirimkan chatbot sudah termasuk *link* ke materi tersebut pada website e-class.

4.2.8 Antarmuka Menu Tugas



Gambar 4.21. Antarmuka tugas

Gambar 4.21 merupakan contoh balasan dari chatbot ketika pengguna menekan menu tugas. Jika mata kuliah yang dipilih mempunyai tugas, maka chatbot akan mengirimkan informasi tugas untuk semua tugas yang ada pada mata kuliah tersebut. Informasi yang dikirimkan chatbot sudah termasuk *link* ke tugas tersebut pada website e-class.

4.2.9 Antarmuka Menu Pengumuman

Gambar 4.22. Antarmuka pengumuman

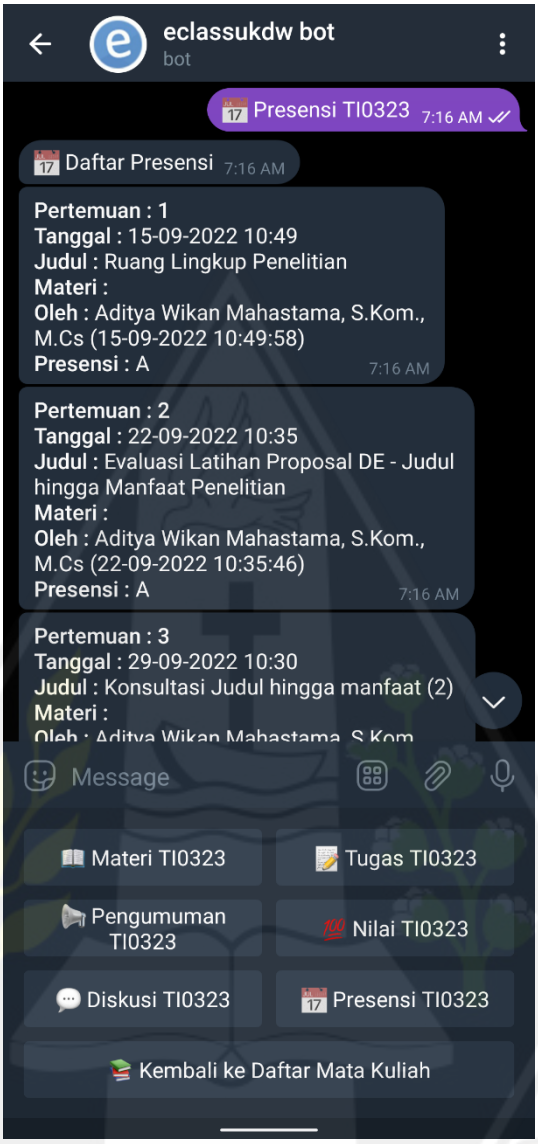
Gambar 4.22 merupakan contoh balasan dari chatbot ketika pengguna menekan menu pengumuman. Jika mata kuliah yang dipilih mempunyai pengumuman, maka chatbot akan mengirimkan informasi pengumuman untuk semua pengumuman yang ada pada mata kuliah tersebut. Informasi yang dikirimkan chatbot sudah termasuk *link* ke pengumuman tersebut pada website eclass.

4.2.10 Antarmuka Menu Nilai

Gambar 4.23. Antarmuka nilai

Gambar 4.23 merupakan contoh balasan dari chatbot ketika pengguna menekan menu nilai. Dalam contoh ini, chatbot mengirimkan pesan balasan berupa belum ada nilai karena pada mata kuliah yang dipilih memang belum memiliki nilai yang sudah dipublikasikan.

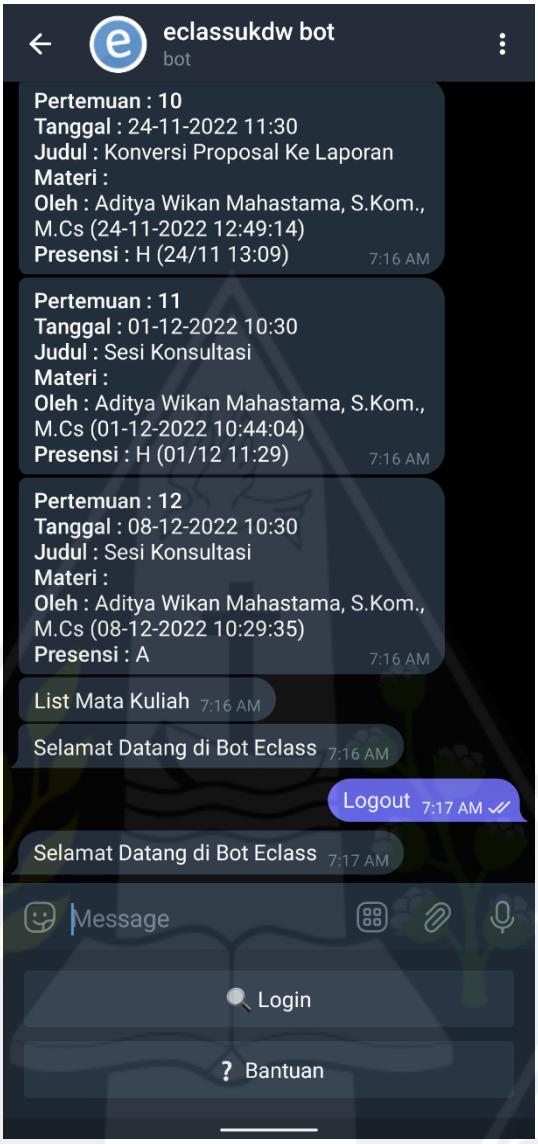
4.2.11 Antarmuka Menu Presensi



Gambar 4.24. Antarmuka presensi

Gambar 4.24 merupakan contoh balasan dari chatbot ketika pengguna menekan menu presensi. Jika mata kuliah yang dipilih mempunyai presensi, maka chatbot akan mengirimkan informasi presensi untuk semua pertemuan yang ada pada mata kuliah tersebut.

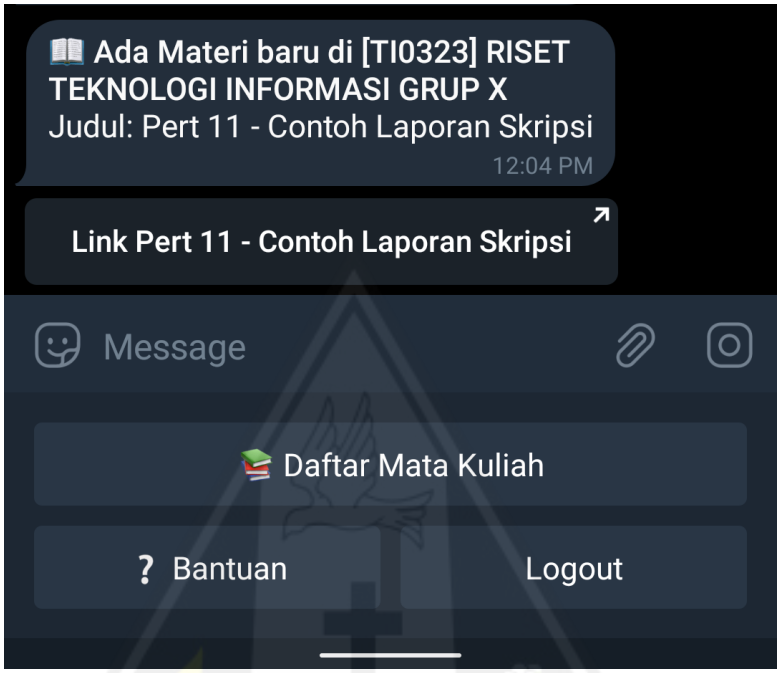
4.2.12 Antarmuka Saat Logout



Gambar 4.25. Antarmuka saat *logout*

Gambar 4.25 merupakan contoh balasan dari chatbot ketika pengguna menekan menu Logout. *Chatbot* akan menghapus semua data yang sudah disimpan, dan kemudian membalas dengan pesan selamat datang di bot eclass.

4.2.13 Antarmuka Notifikasi



Gambar 4.26. Antarmuka notifikasi

Gambar 4.26 merupakan contoh notifikasi yang akan dikirimkan oleh chatbot ketika mendeteksi adanya informasi baru pada website eclass.

4.3 Pembahasan

Pengujian sistem yang dilakukan dalam penelitian ini menggunakan pengujian *User Acceptance Testing (UAT)* dengan metode *Alpha/Beta* Testing.

4.3.1 Pengujian *Alpha Testing*

Pada tahap *Alpha Testing* terdapat 29 skenario yang diuji oleh tiga responden dari program studi yang berbeda-beda. Skenario tersebut mencakup 11 menu yang ada pada chatbot. Data hasil pengujian *Alpha Testing* dapat dilihat pada Tabel 4.1.

Tabel 4.1. Data Hasil Pengujian Alpha Testing

No	Skenario	Hasil yang diharapkan	Ketercapaian	
			Ya	Tidak
1	Melakukan *login* dengan username	Muncul pesan sedang login mohon menunggu diikuti dengan	3	0

No	Skenario	Hasil yang diharapkan	Ketercapaian	
			Ya	Tidak
	atau password benar	pesan Selamat Datang di Bot Eclass		
2	Melakukan *login* dengan username atau password salah	Muncul pesan Login gagal, username atau password salah. Silakan coba lagi	3	0
3	Melakukan login, saat sudah *login*	Muncul pesan silakan *logout* terlebih dahulu	3	0
4	Melihat menu daftar mata kuliah	Muncul pesan list mata kuliah beserta menu untuk setiap mata kuliah yang ada	3	0
5	Melihat menu daftar mata kuliah, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu	3	0
6	Melihat menu info mata kuliah	Muncul pesan informasi berupa kode, nama, grup, dosen pengampu, jadwal, dan ruang beserta menu untuk mata kuliah tersebut	3	0
7	Melihat menu info mata kuliah, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu	3	0
8	Melihat menu info mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan	2	1
9	Melihat menu materi	Jika ada materi, muncul pesan berupa judul, jenis, pembuat, dan link dari setiap materi yang ada	3	0

No	Skenario	Hasil yang diharapkan	Ketercapaian	
			Ya	Tidak
		Jika tidak ada materi, muncul pesan belum ada materi		
10	Melihat menu materi, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu	3	0
11	Melihat menu materi mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan	3	0
12	Melihat menu pengumuman	Jika ada pengumuman, muncul pesan berupa judul, tanggal mulai, tanggal berakhir dan link dari setiap pengumuman yang ada	3	0
		Jika tidak ada pengumuman, muncul pesan belum ada pengumuman		
13	Melihat menu pengumuman, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu	3	0
14	Melihat menu pengumuman mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan	3	0
15	Melihat menu tugas	Jika ada tugas, muncul pesan berupa judul, tanggal mulai, tanggal berakhir dan link dari setiap tugas yang ada	3	0
		Jika tidak ada tugas, muncul pesan belum ada tugas		

No	Skenario	Hasil yang diharapkan	Ketercapaian	
			Ya	Tidak
16	Melihat menu tugas, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu	3	0
17	Melihat menu tugas mata kuliah yang tidak diambil	Muncul pesan mata kuliah tidak ditemukan	3	0
18	Melihat menu nilai	Jika ada nilai, muncul pesan informasi nilai sementara dan nilai maksimal dan pesan berupa judul, bobot, nilai, rata-rata, skala dan link dari setiap nilai yang ada	3	0
		Jika tidak ada nilai, muncul pesan belum ada nilai		
19	Melihat menu nilai, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu	3	0
20	Melihat menu nilai mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan	3	0
21	Melihat menu diskusi	Jika ada diskusi, muncul pesan berupa judul, tanggal, pembuat dan link dari setiap diskusi yang ada	2	1
		Jika tidak ada diskusi, muncul pesan belum ada diskusi		
22	Melihat menu diskusi, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu	3	0
23	Melihat menu diskusi mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan	3	0

No	Skenario	Hasil yang diharapkan	Ketercapaian	
			Ya	Tidak
24	Melihat menu presensi	Jika ada presensi, muncul pesan berupa judul, tanggal, materi, pembuat dan status kehadiran dari setiap presensi yang ada	3	0
		Jika tidak ada presensi, muncul pesan belum ada presensi		
25	Melihat menu presensi, saat pengguna belum login	Muncul pesan silakan login terlebih dahulu	3	0
26	Melihat menu presensi mata kuliah yang tidak diambil mahasiswa	Muncul pesan mata kuliah tidak ditemukan	3	0
27	Terdapat data baru di eclass	Muncul pesan informasi data yang baru	3	0
28	Melakukan *Logout*	Muncul pesan Selamat Datang di Bot Eclass	3	0
29	Melakukan *logout*, saat pengguna belum *login*	Muncul pesan silakan *login* terlebih dahulu	3	0
	Jumlah Nilai		85	2
	Persentase		97,70%	2,30%
	Kategori		Sangat Layak	

Setiap skenario diujikan pada responden memiliki nilai minimum 0 dan nilai maksimum 1. Jika hasil yang diharapkan dari skenario tercapai maka responden akan memberikan nilai 1, sedangkan jika tidak tercapai maka akan memberikan nilai 0. Nilai ini kemudian dijumlahkan dari ketiga responden untuk dihitung jumlah nilai secara keseluruhan. Data pada tabel 4.1 menunjukan bahwa jumlah nilai yang diperoleh dari pengujian ini adalah 85 dari jumlah nilai maksimum 87.

Terdapat dua skenario yang tidak tercapai dengan sempurna ketika diuji oleh responden. Skenario pertama yaitu skenario melihat menu info mata kuliah yang tidak diambil mahasiswa (skenario nomor 8). Pada skenario ini salah satu

responden mencoba mengakses mata kuliah dengan grup yang berbeda dengan yang diambil. Hasilnya responden tetap dapat melihat info mata kuliah tersebut, seharusnya muncul pesan mata kuliah tidak ditemukan. Skenario kedua yaitu skenario melihat menu diskusi (skenario nomor 21). Pada skenario ini salah satu responden mencoba mengakses menu diskusi dari salah satu mata kuliah. Hasilnya responden mendapat pesan belum ada diskusi. Sedangkan pada website *eclass*, mata kuliah tersebut memiliki beberapa diskusi.

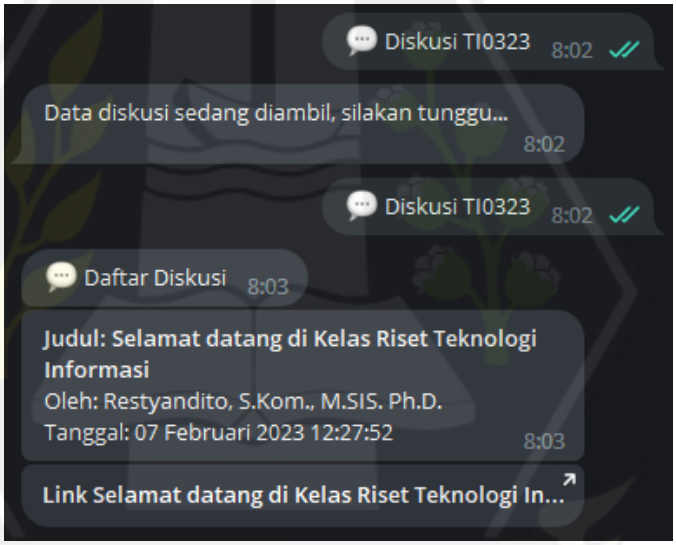
Kesalahan pada skenario nomor 8 terjadi karena *souce code* dari sistem hanya membandingkan kode mata kuliah, seharusnya juga membandingkan grup dari mata kuliah tersebut. Sebagai contoh seorang mahasiswa mengambil mata kuliah A dengan grup 1. Artinya mahasiswa tersebut hanya dapat mengakses mata kuliah A dengan grup 1, meskipun terdapat grup 2, 3, dan seterusnya. Kesalahan ini diperbaiki dengan membandingkan kode serta grup dari mata kuliah yang ingin diakses oleh pengguna. Contoh respon yang benar untuk skenario nomor 8 setelah dilakukan perbaikan seperti pada Gambar 4.27.

Gambar 4.27 Respon *chatbot* untuk skenario 8

Kesalahan pada skenario nomor 21 terjadi pada saat pengguna melakukan proses *login*. Pada proses tersebut *chatbot* mencoba mengambil data dari setiap mata kuliah, termasuk data pada menu diskusi. Namun halaman diskusi dari salah satu mata kuliah tidak kunjung terbuka hingga melewati batas waktu yang sudah

ditentukan. Ketika sudah melewati batas waktu maka *chatbot* akan menyimpan data kosong untuk menu diskusi pada database. Sehingga ketika pengguna ingin melihat menu diskusi, *chatbot* akan menampilkan pesan belum ada diskusi.

Untuk mengatasi kesalahan pada skenario nomor 21, dibuatlah sebuah *error handling.* Metode yang digunakan adalah dengan melakukan pengecekan respons dari halaman yang diakses. Jika status respons yang diberikan sudah OK (status 200-299), maka data akan disimpan sesuai dengan keadaan sesungguhnya. Namun jika status responsnya tidak sesuai, maka data yang disimpan adalah berupa pesan "Halaman Materi tidak dapat diakses". Pesan ini akan dikirimkan kepada pengguna ketika akan mengakses menu dari salah mata kuliah, tetapi menu tersebut belum dapat diakses oleh *server chatbot* saat proses *scraping.* Contoh responss yang benar untuk skenario nomor 21 setelah dilakukan perbaikan seperti pada Gambar 4.28.



Gambar 4.28 Respon *chatbot* untuk skenario 21

Persentase akhir yang diperoleh dari pengujian *Alpha Testing* adalah 97,70%. Persentase ini dihitung dengan cara jumlah nilai total dibagi jumlah nilai maksimum kemudian dikalikan dengan 100%. Nilai persentase yang sudah diperoleh kemudian dapat diinterpretasikan menggunakan Tabel 4.2.

Tabel 4.2. Pedoman Interpretasi Persentase

Angka (%)	Kategori
0 - 20	Sangat Tidak Layak
21 - 40	Tidak Layak
41 - 60	Cukup
61 - 80	Layak
81 - 100	Sangat Layak

Kategori yang diperoleh dari nilai persentase 97,70% adalah Sangat Layak. Hasil ini dapat diartikan bahwa *chatbot* yang dikembangkan sudah dapat berfungsi sesuai dengan rencana awal. Kedua kesalahan yang terjadi untuk skenario nomor 8 dan skenario nomor 21 juga sudah diperbaiki untuk kemudian diujikan di tahap selanjutnya.

4.3.2 Pengujian *Beta Testing*

Pada tahap *Beta Testing* terdapat 10 skenario yang diuji oleh 30 responden dari program studi yang berbeda-beda. Skenario tersebut mencakup 10 menu yang ada pada *chatbot*. Data hasil pengujian *Beta Testing* dapat dilihat pada hingga.

Tabel 4.3. Hasil Kuesioner Soal Nomor 1

Keterangan	**Skala**	**Responden**	**Nilai**
Sangat Tidak Setuju	1	0	0
Tidak Setuju	2	2	4
Netral	3	7	21
Setuju	4	8	32
Sangat Setuju	5	13	65
Total			122
Nilai Maksimum			150
Persentase			81,33%

Kuesioner soal nomor 1 merupakan sebuah pernyataan tentang chatbot ini mudah digunakan. Tabel 4.3 menunjukkan total nilai yang diperoleh dari responden adalah 122 dari nilai maksimum 150. Nilai 122 ini memiliki persentase 81,33%.

Tabel 4.4. Hasil Kuesioner Soal Nomor 2

Keterangan	Skala	Responden	Nilai
Sangat Tidak Setuju	1	0	0
Tidak Setuju	2	1	2
Netral	3	7	21
Setuju	4	10	40
Sangat Setuju	5	12	60
Total			123
Nilai Maksimum			150
Persentase			82,00%

Kuesioner soal nomor 2 merupakan sebuah pernyataan tentang menu yang ada dalam chatbot ini mudah dipahami. Tabel 4.4 menunjukkan total nilai yang diperoleh dari responden adalah 123 dari nilai maksimum 150. Nilai 123 ini memiliki persentase 82,00%.

Tabel 4.5. Hasil Kuesioner Soal Nomor 3

Keterangan	Skala	Responden	Nilai
Sangat Tidak Setuju	1	1	1
Tidak Setuju	2	1	2
Netral	3	5	15
Setuju	4	11	44
Sangat Setuju	5	12	60
Total			122
Nilai Maksimum			150
Persentase			81,33%

Kuesioner soal nomor 3 merupakan sebuah pernyataan tentang Ikon yang digunakan dalam setiap menu mudah dipahami. Tabel 4.5 menunjukkan total nilai yang diperoleh dari responden adalah 122 dari nilai maksimum 150. Nilai 122 ini memiliki persentase 81,33%.

Tabel 4.6. Hasil Kuesioner Soal Nomor 4

Keterangan	Skala	Responden	Nilai
Sangat Tidak Setuju	1	2	2
Tidak Setuju	2	1	2
Netral	3	6	18

Keterangan	Skala	Responden	Nilai
Setuju	4	10	40
Sangat Setuju	5	11	55
Total			117
Nilai Maksimum			150
Persentase			78,00%

Kuesioner soal nomor 4 merupakan sebuah pernyataan tentang Navigasi dalam chatbot ini mudah. Tabel 4.6 menunjukkan total nilai yang diperoleh dari responden adalah 117 dari nilai maksimum 150. Nilai 117 ini memiliki persentase 78,00%.

Tabel 4.7. Hasil Kuesioner Soal Nomor 5

Keterangan	Skala	Responden	Nilai
Sangat Tidak Setuju	1	1	1
Tidak Setuju	2	1	2
Netral	3	6	18
Setuju	4	9	36
Sangat Setuju	5	13	65
Total			122
Nilai Maksimum			150
Persentase			81,33%

Kuesioner soal nomor 5 merupakan sebuah pernyataan tentang saya tidak membutuhkan bantuan dari orang lain untuk menggunakan chatbot ini. Tabel 4.7 menunjukkan total nilai yang diperoleh dari responden adalah 122 dari nilai maksimum 150. Nilai 122 ini memiliki persentase 81,33%.

Tabel 4.8. Hasil Kuesioner Soal Nomor 6

Keterangan	Skala	Responden	Nilai
Sangat Tidak Setuju	1	0	0
Tidak Setuju	2	0	0
Netral	3	5	15
Setuju	4	12	48
Sangat Setuju	5	13	65
Total			128
Nilai Maksimum			150
Persentase			85,33%

Kuesioner soal nomor 6 merupakan sebuah pernyataan tentang saya merasa menu yang ada dalam chatbot ini sudah berjalan dengan sesuai. Tabel 4.8 menunjukkan total nilai yang diperoleh dari responden adalah 128 dari nilai maksimum 150. Nilai 128 ini memiliki persentase 85,33%.

Tabel 4.9. Hasil Kuesioner Soal Nomor 7

Keterangan	Skala	Responden	Nilai
Sangat Tidak Setuju	1	1	1
Tidak Setuju	2	1	2
Netral	3	4	12
Setuju	4	12	48
Sangat Setuju	5	12	60
Total			123
Nilai Maksimum			150
Persentase			82,00%

Kuesioner soal nomor 7 merupakan sebuah pernyataan tentang saya dapat menemukan apa yang saya cari dengan cepat di chatbot ini. Tabel 4.9 menunjukkan total nilai yang diperoleh dari responden adalah 123 dari nilai maksimum 150. Nilai 122 ini memiliki persentase 82,00%.

Tabel 4.10. Hasil Kuesioner Soal Nomor 8

Keterangan	Skala	Responden	Nilai
Sangat Tidak Setuju	1	1	1
Tidak Setuju	2	2	4
Netral	3	5	15
Setuju	4	13	52
Sangat Setuju	5	9	45
Total			117
Nilai Maksimum			150
Persentase			78,00%

Kuesioner soal nomor 8 merupakan sebuah pernyataan tentang saya merasa respon chatbot sudah tepat dan cepat. Tabel 4.10 menunjukkan total nilai yang diperoleh dari responden adalah 117 dari nilai maksimum 150. Nilai 117 ini memiliki persentase 78,00%.

Tabel 4.11. Hasil Kuesioner Soal Nomor 9

Keterangan	Skala	Responden	Nilai
Sangat Tidak Setuju	1	0	0
Tidak Setuju	2	0	0
Netral	3	3	9
Setuju	4	16	64
Sangat Setuju	5	11	55
Total			128
Nilai Maksimum			150
Persentase			85,33%

Kuesioner soal nomor 9 merupakan sebuah pernyataan tentang saya akan menggunakan chatbot ini lagi. Tabel 4.11 menunjukkan total nilai yang diperoleh dari responden adalah 128 dari nilai maksimum 150. Nilai 128 ini memiliki persentase 85,33%.

Tabel 4.12. Hasil Kuesioner Soal Nomor 10

Keterangan	Skala	Responden	Nilai
Sangat Tidak Setuju	1	0	0
Tidak Setuju	2	1	2
Netral	3	7	21
Setuju	4	15	60
Sangat Setuju	5	7	35
Total			118
Nilai Maksimum			150
Persentase			78,67%

Kuesioner soal nomor 10 merupakan sebuah pernyataan tentang saya mungkin akan merekomendasikan chatbot ini ke teman atau kolega. Tabel 4.12 menunjukkan total nilai yang diperoleh dari responden adalah 118 dari nilai maksimum 150. Nilai 118 ini memiliki persentase 78,67%.

Untuk mengetahui *User Acceptance Index* dari chatbot yang dikembangkan maka perlu dihitung menggunakan rumus sebagai berikut:

$$Rata - rata\ total\ nilai = \frac{(total\ nilai_1 + total\ nilai_2 + total\ nilai_3 + \cdots + total\ nilai_n)}{n}$$

$$Rata - rata\ total\ nilai$$
$$= \frac{(122 + 123 + 122 + 117 + 122 + 128 + 123 + 117 + 128 + 118)}{10}$$
$$Rata - rata\ total\ nilai = \frac{1220}{10} = 122$$

$$User\ Acceptance\ Index = \frac{Rata - rata\ total\ nilai}{Nilai\ maksimum} \times 100\%$$
$$User\ Acceptance\ Index = \frac{122}{150} \times 100\%$$
$$User\ Acceptance\ Index = 81{,}33\%$$

Dari perhitungan di atas, didapatkan persentase *User Acceptance Index* oleh responden *Beta Testing* dari chatbot sebesar 81,33%. Hasil ini dapat diinterpretasikan menggunakan bantuan tabel 4.2 sehingga chatbot yang dikembangkan dapat diterima oleh responden dengan sangat layak.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan pada hasil pengujian *User Acceptance Test* dengan metode *Alpha/Beta Testing* terhadap chatbot telegram untuk layanan informasi akademik pada website e-class didapatkan kesimpulan sebagai berikut:

1. Pengujian *UAT* tahap *Alpha Testing* menggunakan metode *blackbox* sudah berhasil dilakukan dengan menggunakan 29 skenario. Sebanyak 27 skenario memiliki nilai persentase kesuksesan sebesar 100% (sangat layak). Sedangkan dua skenario lainnya memiliki nilai persentase kesuksesan 66,66% (layak) untuk skenario melihat menu info mata kuliah yang tidak diambil mahasiswa dan melihat menu diskusi.
2. Pengujian *UAT* tahap *Beta Testing* yang dilakukan dengan menggunakan kuesioner mendapatkan hasil *User Acceptance Index* sebesar 81,33% dan termasuk ke dalam kategori sangat layak.
3. Menu diskusi mata kuliah pada chatbot merupakan satu-satunya menu yang mendapatkan *error*.

5.2 Saran

Dalam penelitian ini tentunya masih ditemukan beberapa kekurangan, saran yang dapat diberikan adalah sebagai berikut:

1. Penyimpanan data login sebaiknya tidak disimpan bersama dengan data mata kuliah pengguna. Misalnya dengan menyimpan data login tersebut dalam memori program.
2. Penambahan menu yang belum tersedia pada *chatbot* seperti menu asisten dan peserta agar informasi pada *chatbot* menjadi lebih lengkap.

3. Pengoptimalan proses *login* agar dapat berlangsung dengan lebih cepat. Misalnya dengan melakukannya secara *asynchronous* untuk setiap mata kuliah.

DAFTAR PUSTAKA

Bots: An introduction for developers. (2022, June 13). Retrieved from Telegram Messenger: https://core.telegram.org/bots

Broucke, S. v., & Baesens, B. (2018). *Practical Web Scraping for Data Science: Best Practices and Examples with Python.* Apress.

Efendy, V., Nugraha, K. A., & Sebastian, D. (2019). Implementasi Chat Room dan Push Notification. *Jurnal Teknik Informatika dan Sistem Informasi*, 5(2). doi:https://doi.org/10.28932/jutisi.v5i2.1763

Hambling, B., & Goathem, P. V. (2013). *User acceptance testing : a step-by-step guide.* BCS.

Khan, R., & Das, A. (2018). *Build Better Chatbots: A Complete Guide to Getting Started with Chatbots.* Apress.

Kondratiuk, D. (2021). *UI Testing with Puppeteer: Implement end-to-end testing and browser automation using JavaScript and Node.js.* Packt Publishing Ltd.

Leonardo, G. C., Herianto, & Irawan, Y. (2020). Pemanfaatan Bot Telegram sebagai Media Informasi Akademik di STMIK Hang Tuah Pekanbaru. *JTIM : Jurnal Teknologi Informasi dan Multimedia*, 351-357. doi:10.35746/jtim.v1i4.59

Long Polling vs. Webhooks | grammY. (2022, June 13). Retrieved from grammY: https://grammy.dev/guide/deployment-types.html

Mitchell, R. (2018). *Web Scraping with Python, 2nd Edition.* O'Reilly Media, Inc.

Rosyana, F. P., Onno, W. P., & RZ., A. A. (2021). *Firebase: Membangun Aplikasi Berbasis Android.* Penerbit Andi.

Sastrawangsa, G. (2017). Pemanfaatan Telegram Bot Untuk Automatisasi Layanan Dan Informasi Mahasiswa Dalam Konsep Smart Campus. *E-Proceedings KNS&I STIKOM Bali*, 772-776. Retrieved from https://knsi.stikom-bali.ac.id/index.php/eproceedings/article/view/138

Wibowo, A. K., & Kurniawan, Y. I. (2019). BOT TELEGRAM SEBAGAI MEDIA ALTERNATIF. *Komputa : Jurnal Ilmiah Komputer Dan Informatika, 8*(1), 1–10. doi:https://doi.org/10.34010/komputa.v8i1.3043

Yahiaoui, H. (2017). *Firebase Cookbook.* Packt.

Yehezkiel. (2019). APLIKASI E-CLASS UNIVERSITAS KRISTEN DUTA WACANA BERBASIS ANDROID MOBILE. *Bachelor thesis, Universitas Kristen Duta Wacana*.