

**PERANCANGAN DAN IMPLEMENTASI BOT DISCORD UNTUK  
PENGELOLAAN KOMUNITAS MENGGUNAKAN PENDEKATAN  
JAVASCRIPT**



**MOHAMMAD RIZKY**

**SITI NURJANNAH**

**SYIFA KHAERUNNISA**

**Perancangan Perangkat Lunak**

**TEKNOLOGI INFORMASI**

**FAKULTAS TEKNIK**

**UNIVERSITAS TANGERANG RAYA**

**TAHUN 2024/2025**

## DAFTAR ISI

DAFTAR ISI.....	I
PENDAHULUAN.....	1
1.1. Latar Belakang.....	1
1.2. Identifikasi Masalah.....	2
1.3. Rumusan Masalah.....	2
1.4. Tujuan.....	2
PERANCANGAN DAN IMPLEMENTASI.....	3
2.1. Skema Hardware.....	3
2.2. Skema Software.....	4
2.3. Skema Database.....	7
2.4. Skema Actor.....	8
2.5. Coding, Implementasi, & Testing.....	10
PENUTUP.....	13
3.1. Kesimpulan.....	13
3.2. Saran.....	13

# PENDAHULUAN

## 1.1 Latar Belakang

Pada era saat ini teknologi selalu mengalami perkembangan dari waktu ke waktu. Internet merupakan suatu contoh teknologi yang sangat melekat pada kebutuhan manusia saat ini khususnya di Indonesia. Menurut survei menemukan bahwa 132,7 juta dari 256 juta penduduk di Indonesia telah terhubung ke internet (Asosiasi Penyelenggara Jaringan Internet Indonesia, 2016). Secara fisik internet merupakan interkoneksi antar jaringan dan sumber daya informasi dan bahkan internet merupakan dunia dalam bentuk maya karena banyak aspek di kehidupan nyata terdapat pada internet seperti hiburan, bisnis, politik, dan lain sebagainya (Lani Sidharta, 1996).

Bot Discord berbasis JavaScript menawarkan solusi yang efektif untuk mengatasi tantangan tersebut. JavaScript, sebagai salah satu bahasa pemrograman yang kuat dan serbaguna, mendukung pembuatan bot yang dapat diintegrasikan langsung dengan Discord melalui pustaka seperti Discord.js (Harris, 2019). Dengan menggunakan bot, pengelola komunitas dapat mengotomatisasi interaksi dan mengurangi beban kerja manual, sehingga pengalaman pengguna dalam komunitas menjadi lebih baik (Anderson, 2021).

Maka dari itu, perancangan ini bertujuan untuk merancang dan mengimplementasikan bot Discord berbasis JavaScript yang berfokus pada pengelolaan komunitas. Dalam perancangan ini, aspek-aspek teknis seperti pembuatan flowchart, *Use Case*, *Activity Diagram*, dan lain sebagainya akan digunakan untuk merencanakan dan memvisualisasikan alur kerja bot secara lebih terstruktur. Dengan adanya perancangan yang sistematis, bot ini diharapkan dapat memenuhi kebutuhan komunitas dalam hal automasi dan moderasi, serta menjadi referensi bagi pengembang lain yang ingin mengembangkan bot serupa.

## **1.2 Identifikasi Masalah**

Dari latar belakang yang ditulis, pengembang memberikan identifikasi masalah yang akan dijadikan topik utama perancangan yaitu bagaimana merancang dan mengimplementasikan bot Discord berbasis JavaScript yang mampu mengotomatisasi berbagai tugas pengelolaan komunitas, seperti moderasi serta penyediaan layanan interaktif yang menarik. Sistem informasi yang dirancang ini diharapkan dapat mengumpulkan dan menganalisis data interaksi anggota untuk menciptakan lingkungan komunitas yang lebih terstruktur, efisien, dan menarik. Perancangan ini juga diharapkan dapat menjadi acuan bagi pengelola komunitas Discord dalam menciptakan sistem bot yang membantu mempermudah administrasi serta meningkatkan keterlibatan dan kenyamanan anggota.

## **1.3 Rumusan Masalah**

Berdasarkan latar belakang permasalahan yang ada, maka rumusan masalah dalam perancangan ini adalah:

- a. Belum adanya bot Discord yang memiliki fitur-fitur pengelolaan komunitas secara rinci dan berbasis fungsionalitas, baik untuk fungsi moderasi maupun penyediaan layanan interaktif yang menarik bagi anggota.
- b. Bagaimana merancang dan mengimplementasikan bot Discord berbasis JavaScript yang mampu mengotomatisasi tugas-tugas pengelolaan komunitas dan meningkatkan keterlibatan serta kenyamanan anggota di dalam komunitas Discord.

## **1.4 Tujuan**

Tujuan yang ingin dicapai dalam perancangan ini adalah untuk meningkatkan efisiensi pengelolaan komunitas Discord melalui bot berbasis JavaScript yang mampu mengotomatisasi tugas-tugas rutin, seperti moderasi dan penyediaan layanan interaktif. Dengan adanya bot yang terintegrasi dalam satu sistem, diharapkan proses administrasi menjadi lebih cepat dan akurat, sehingga menciptakan lingkungan

## PERANCANGAN DAN IMPLEMENTASI

### 2.1. Skema Hardware

Hardware yang digunakan dalam perancangan bot Discord akan bergantung pada kebutuhan aplikasi bot dan skala penggunaannya. Berikut adalah poin-poin rinci kebutuhan hardware yang kami gunakan selama perancangan bot/aplikasi:

- Hosting

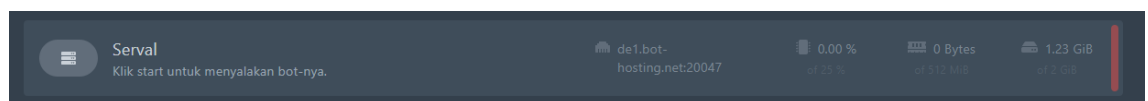
Disini kami menggunakan shared hosting alternatif yang gratisan untuk menjalankan bot-nya secara efisien dan mudah menggunakan <https://bot-hosting.net/> , dimana sebagian besar kebutuhan hardware fisik dialihkan ke penyedia layanan hosting. Alasan kami memilih hosting ini selain gratis juga memiliki vCPU yang memadai untuk menjalankan Node.js dan menangani permintaan dari Discord API secara efisien. Hositng ini juga menawarkan bandwidth yang cukup untuk komunikasi *real-time* antara bot dan Discord API serta menjamin uptime yang tinggi sehingga bot dapat berjalan tanpa gangguan. Untuk pemilihan pakatnya sendiri kami menggunakan paket yang standar dimana kami mendapatkan penggunaan vCPU sebanyak 25%, RAM sebanyak 512 MB, dan SSD sebanyak 2 GB.

- Perangkat Pengembangan

Laptop atau PC dengan spesifikasi standar (CPU minimal dual-core, RAM 4 GB, penyimpanan HDD/SSD) untuk pengembangan dan pengujian bot sebelum deployment.

- Penyimpanan

Minimal 5 GB untuk menyimpan kode bot, library, database lokal (jika diperlukan), dan file pendukung lainnya.



Gambar diatas merupakan manajemen dan monitoring dari cloud hosting yang digunakan.

## 2.2. Skema Software

Software memainkan peran penting dalam pengembangan dan operasional bot Discord. Berikut adalah rincian kebutuhan software yang biasa kami gunakan selama pengembangan:

- Sistem Operasi

Kami menggunakan OS Windows selama pengembangan karena sesuai dengan preferensi kami dan jenis hosting yang akan kami gunakan.

- Runtime Environment

Kami menggunakan Node.js sebagai runtime environment untuk menjalankan bot berbasis *JavaScript*. Versi yang kami gunakan adalah versi LTS terbaru untuk stabilitas.

- Library dan Framework

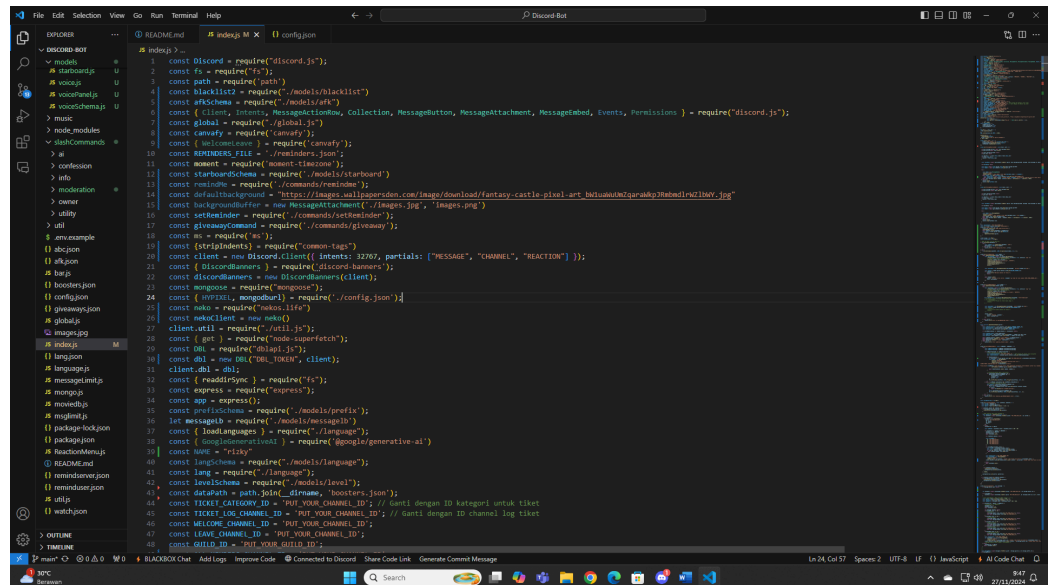
- Discord.js: Library utama yang kami gunakan untuk berinteraksi dengan Discord API, memungkinkan bot untuk menerima perintah, mengirim pesan, dan memodifikasi channel atau role.
- Express.js: Untuk membuat API tambahan jika bot memerlukan antarmuka web atau endpoint HTTP.

- Database

Kami menggunakan MongoDB karena ini merupakan pilihan populer untuk database *NoSQL* yang cocok untuk menyimpan data leveling, role management, atau konfigurasi server.

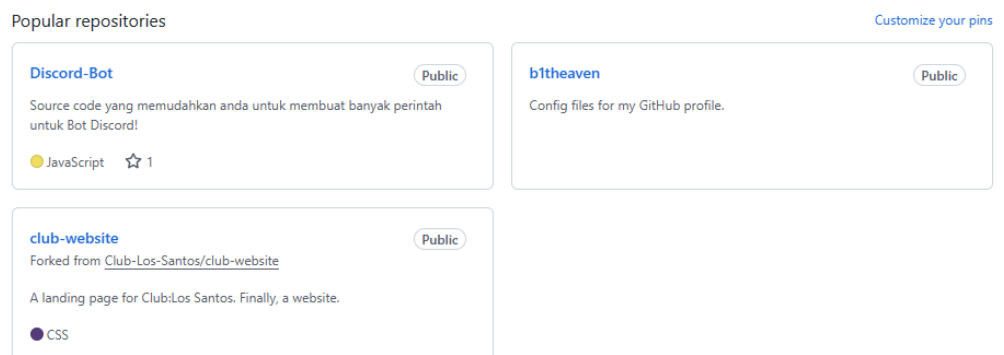
- Alat Pengembangan

- Code Editor: Kami menggunakan VSCode selama proses pengembangan karena VSCode merupakan editor teks yang ringan, cepat, dan fleksibel untuk pengembangan Node.js. VSCode juga mendukung banyak ekstensi yang kami butuhkan selama perancangan dan editor teks ini memang populer saat-saat ini yang paling sering digunakan.



Gambar diatas merupakan tampilan dari Visual Studio Code.

- Version Control System: Untuk version control dan kolaborasi pengembangan kami menggunakan GitHub untuk membuat eksprimen tanpa memengaruhi kode utama, dan juga fitur *pull requests* untuk memeriksa kode sebelum penggabungan.



Gambar diatas merupakan tampilan repositori pada aplikasi GitHub.

## • Deployment Tools

Pada bagian ini, kami menggunakan PM2 dimana PM2 sendiri merupakan proses manager untuk Node.js, yang menjaga agar bot tetap berjalan meskipun terjadi error. Selain itu kami juga menggunakan Docker untuk *containerization*, sehingga bot dapat dijalankan di berbagai lingkungan tanpa masalah kompatibilitas.

- Security Tools

Untuk mengelola dan melindungi kredensial seperti token bot atau kunci API, serta keamanan tambahan untuk melindungi dari serangan eksternal kami menggunakan dotenv karena pilihan ini adalah yang paling mudah dan simpel namun sangat worth it.

PLAINTEXT

## Environment Variables

The .env file is for storing secrets for your app, like an API key. Any project member can see the contents in the same way that you can, and everyone else can just see the variable names. [Learn How to Use Environment Variables](#)

Comments are visible in remixes. Make sure there are no secrets in your comments!

# Scrubbed by Glitch 2021-10-27T10:23:15+0000

Variable Name	Value
TOPGGTOKEN	eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp

# Scrubbed by Glitch 2024-06-15T12:25:51+0000

Variable Name	Value
MOVIE_KEY	438d8c8767d77cfdacc9e93136b

Variable Name	Value
DATABASE	mongodb+srv://ateez:ateez123@

Variable Name	Value
CHATBOT_APIKEY	AlzaSyASYd3G2WYyqYSflmL7b

Variable Name	Value
YOUTUBE_API_KEY	AlzaSyB6eeQJulQr8vqj_WFHne

Variable Name	Value
TOKEN	MTI3MDM2MzQxNTE5NzUxOTg

Add a variable

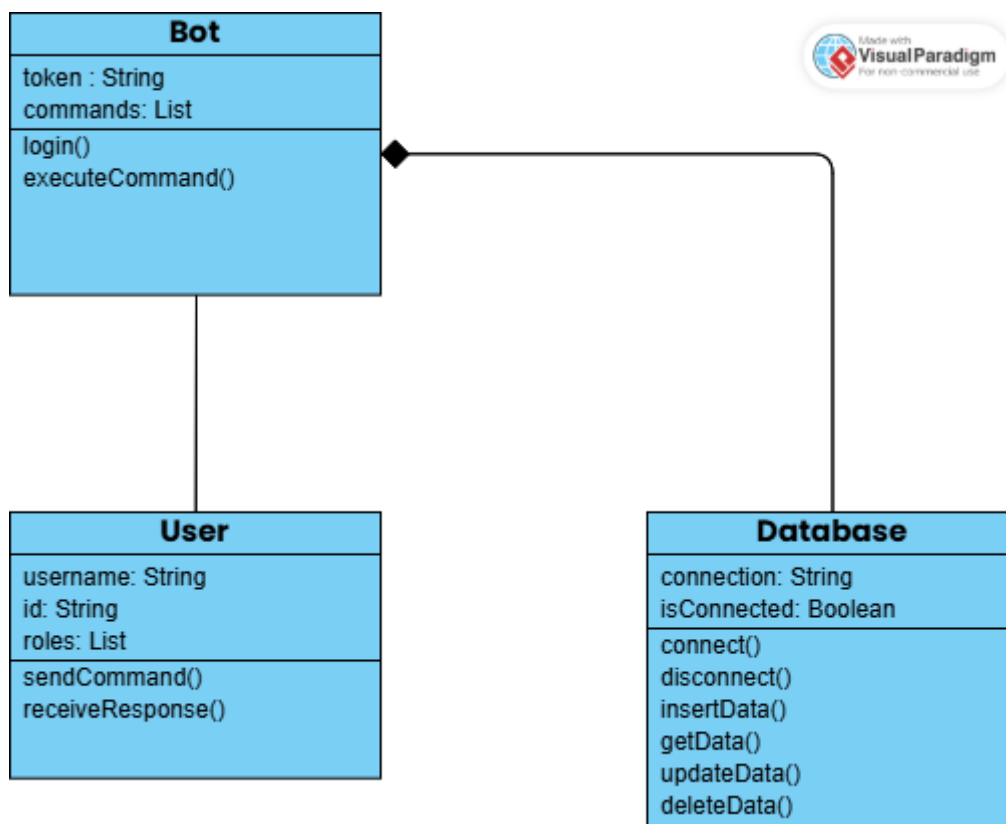
Gambar diatas merupakan contoh tampilan isi dotenv.

6



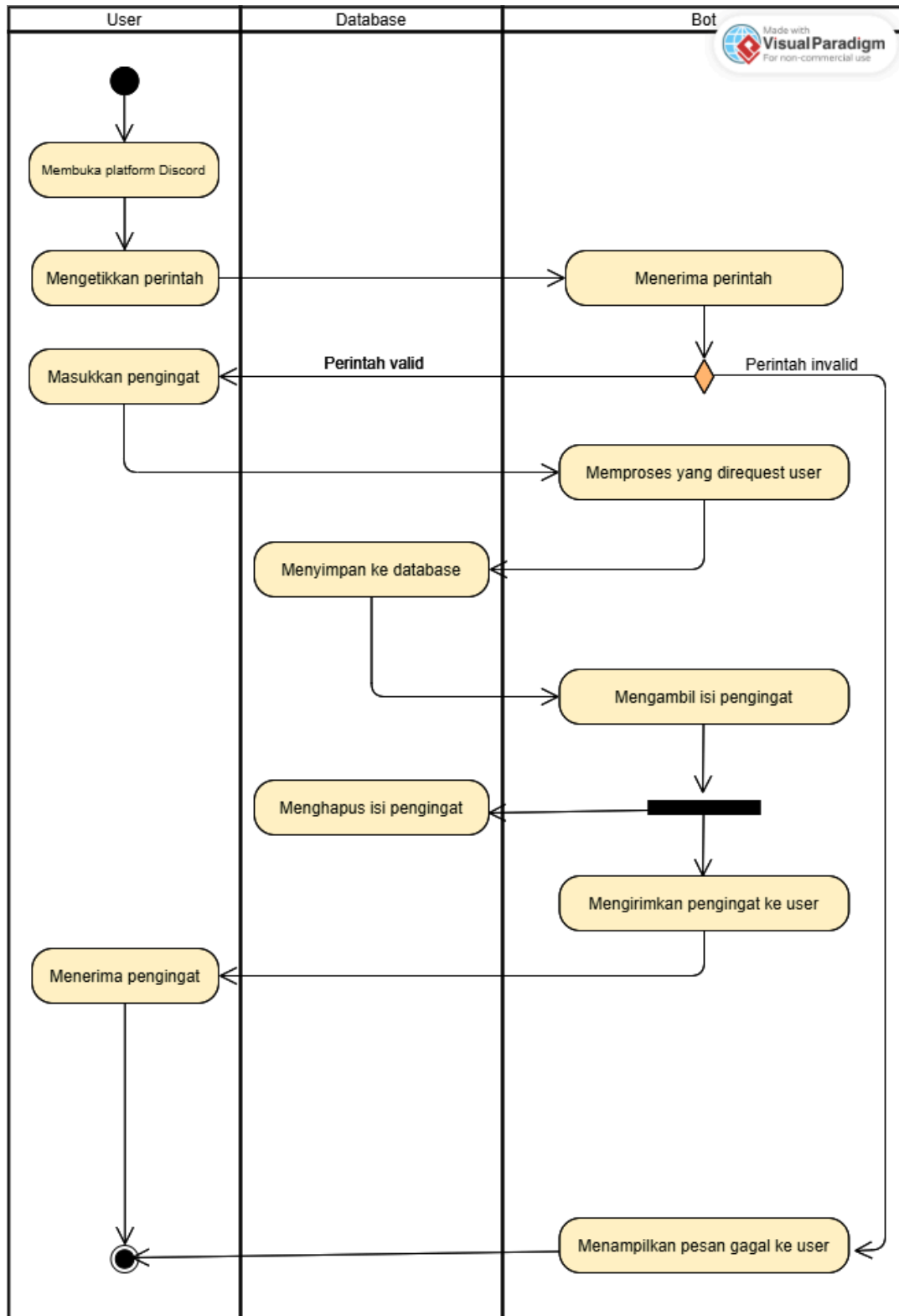
### 2.3. Skema Database

Pada skema ini, bot atau aplikasi yang kami buat menerima perintah dari user melalui metode `sendCommand` dan user menerima respon melalui `receiveResponse`. Sedangkan bot bergantung pada database untuk berbagai operasi seperti menyimpan log aktivitas, mengambil data pengguna, atau memperbarui data server. Database juga menyimpan atribut-atribut yang dimiliki user yang kalian bisa lihat sendiri dibawah ini.

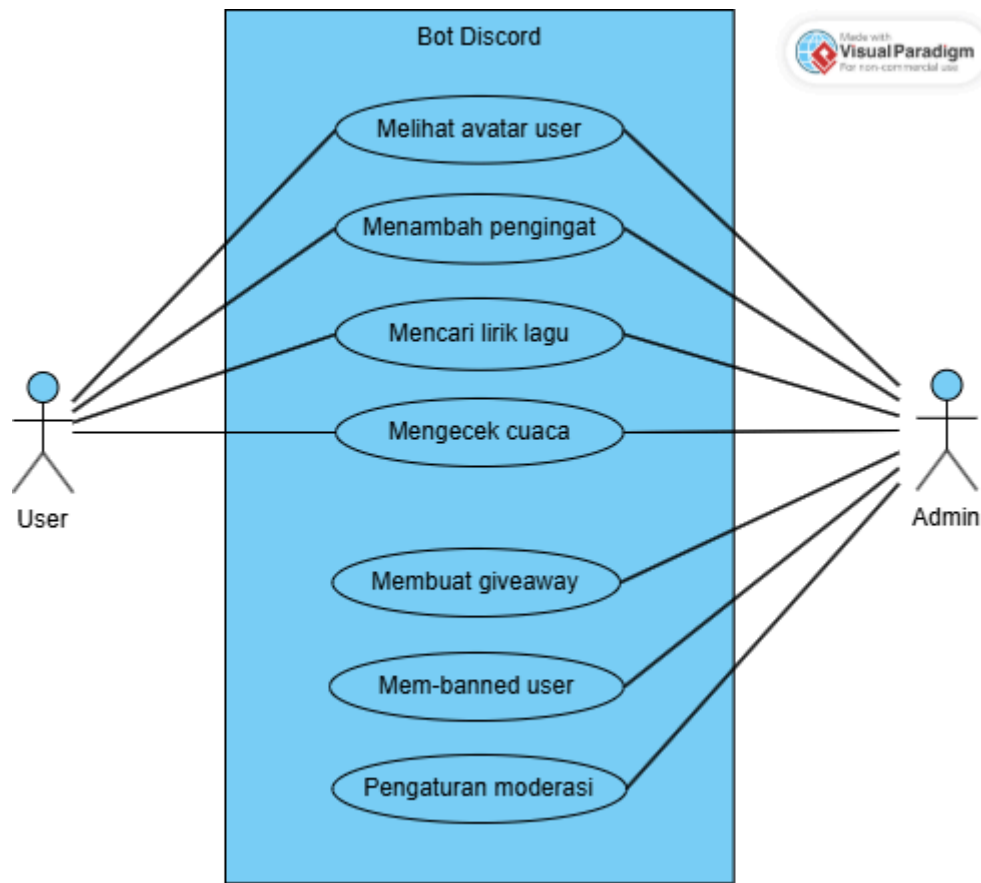


## 2.4. Skema Actor

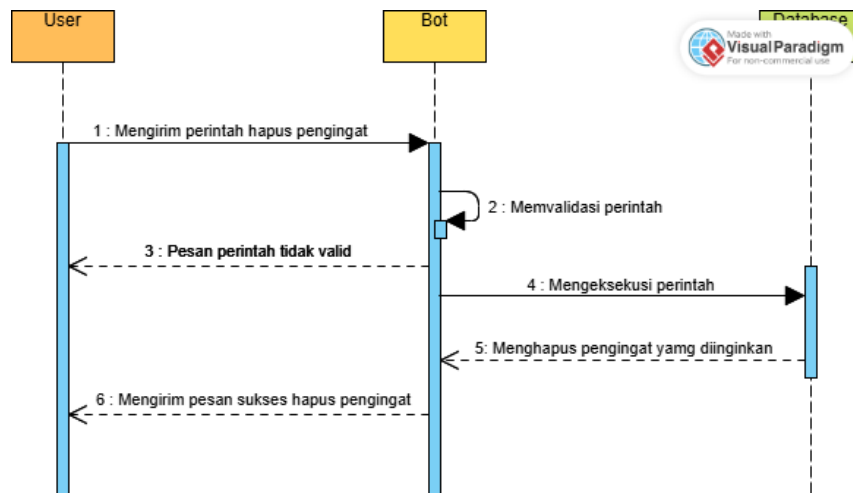
Berikut adalah activity diagram dari penggunaan bot untuk membuat pengingat sebagai contoh:



Untuk use case berikut adalah gambar sederhana mengenai relasi dan fungsi antara admin, user, dan chatbot:

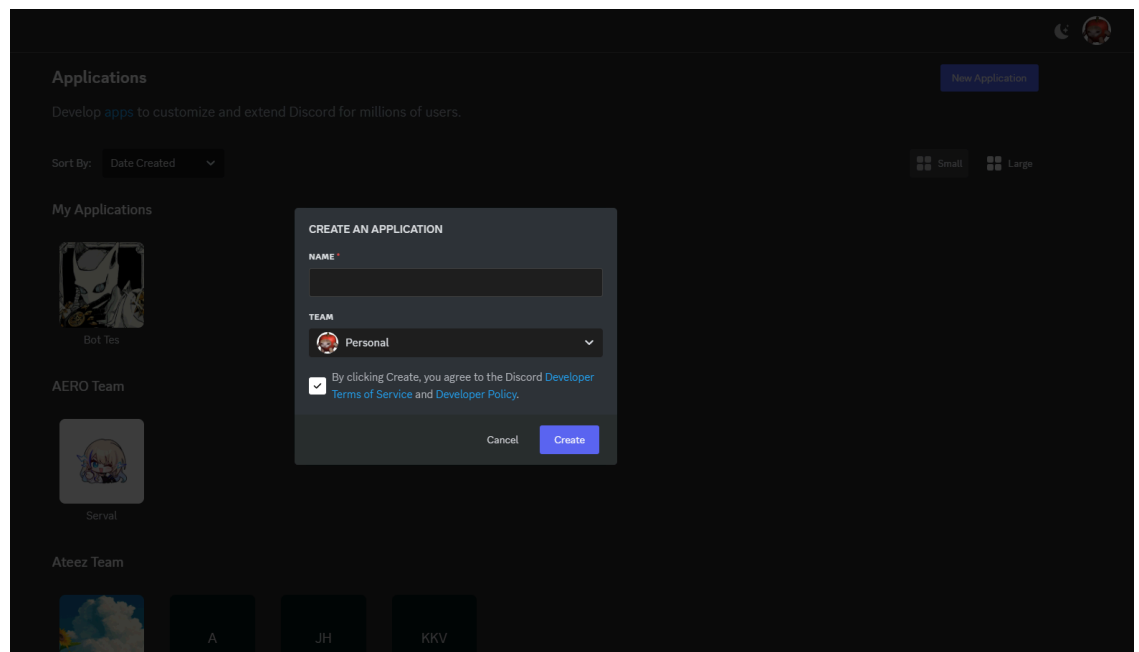


Dan terakhir, masih dalam penerapan buat/hapus pengingat pada chatbot namun kali ini berbentuk sequence diagram adalah sebagai berikut:



## 2.5. Coding, Implementasi, dan Testing

Pertama-tama, pengembang harus memiliki akun Discord dan masuk ke platform, lalu pengembang bisa langsung pergi ke <https://discord.com/developers/applications> . Setelah itu pengembang bisa membuat bot Discord-nya terlebih dahulu dengan menekan tombol “New Application” lalu beri nama dan Create. Pastikan pengembang sudah membaca dan meyetujui *Developer ToS and Policy*.



Gambar diatas merupakan tampilan saat pertama kali ingin membuat bot di Discord.

Setelah itu pengembang bisa mengklik aplikasi yang sudah dibuat di “My Applications” lalu pengembang bisa mengisi deskripsi serta tag-nya. Jika sudah, pengembang bisa pergi ke Settings > Bot untuk mengatur foto profil dan banner bot serta tokennya untuk bisa dikoneksikan kedalam program yang akan dibuat.

Pada tahap pemrograman, kami menggunakan aplikasi Visual Studio Code untuk mengimplementasikan kode dan Node.js sebagai runtime environment. Karena kami menggunakan Discord sebagai platform, maka kami menggunakan Discord.js sebagai modul atau library yang berisi fungsi-fungsi yang digunakan di dalam aplikasi Discord. Untuk menggunakan modul Discord.js, kami harus menginstal modulnya terlebih dahulu dengan

mengetikkan `npm install discord.js` pada terminal di Visual Studio Code.



```
PS C:\Users\WADMIN\Downloads\Discord-Bot> npm i discord.js
npm warn EBADENGINE Unsupported engine {
  npm warn EBADENGINE   package: 'discord.js@9.8.1',
  npm warn EBADENGINE   required: { node: '16.x' },
  npm warn EBADENGINE   current: { node: 'v20.14.0', npm: '10.7.0' }
  npm warn EBADENGINE }

up to date, audited 515 packages in 4s

79 packages are looking for funding
  run 'npm fund' for details

10 vulnerabilities (3 low, 3 moderate, 4 high)

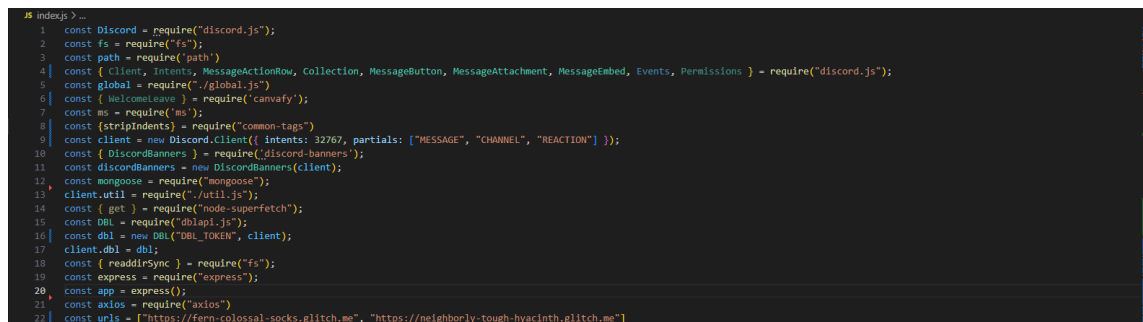
To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run 'npm audit' for details.
PS C:\Users\WADMIN\Downloads\Discord-Bot>
```

Gambar diatas merupakan proses intalasi library Discord.js V13.

Modul yang diinstal secara otomatis akan menghasilkan file package.json dan package-lock.json. Selanjutnya, kami harus mengimpor modul Discord.js ke dalam aplikasi Discord menggunakan fungsi require di file utama. Kami juga membuat satu file .env untuk menyimpan data token dan prefix bot.



```
1 const Discord = require("discord.js");
2 const fs = require("fs");
3 const path = require("path");
4 const { Client, Intents, MessageActionRow, Collection, MessageButton, MessageAttachment, MessageEmbed, Events, Permissions } = require("discord.js");
5 const global = require("./global.js");
6 const { WelcomeLeave } = require("canvafy");
7 const ms = require("ms");
8 const { stripIndents } = require("common-tags");
9 const client = new Discord.Client({ intents: 32767, partials: ["MESSAGE", "CHANNEL", "REACTION"] });
10 const { DiscordBanners } = require("discord-banners");
11 const discordBanners = new DiscordBanners(client);
12 const mongoose = require("mongoose");
13 client.util = require("./util.js");
14 const { get } = require("node-superfetch");
15 const db1 = require("dbapi.js");
16 const db1 = new db1("DB1_TOKEN", client);
17 client.db1 = db1;
18 const { readdirSync } = require("fs");
19 const express = require("express");
20 const app = express();
21 const axios = require("axios");
22 const url1 = ["https://fern-colossal-socks.glitch.me", "https://neighborly-tough-hyacinth.glitch.me"]
```

Gambar diatas merupakan proses impor modul Discord.js ke file utama.

Dalam penulisan kode, kami menggunakan bahasa pemrograman JavaScript karena JavaScript merupakan bahasa yang cukup populer. Menurut kami, JavaScript merupakan salah satu bahasa yang cukup mudah untuk dipelajari. Karena kami membuat chatbot berbasis peraturan yaitu *decision tree-based* yang membuat alur percakapan, kami menggunakan logika *if*. Dimana dalam contoh gambar dibawah ini kami membuat sistem selamat datang secara otomatis kepada setiap anggota yang baru bergabung kedalam komunitas.

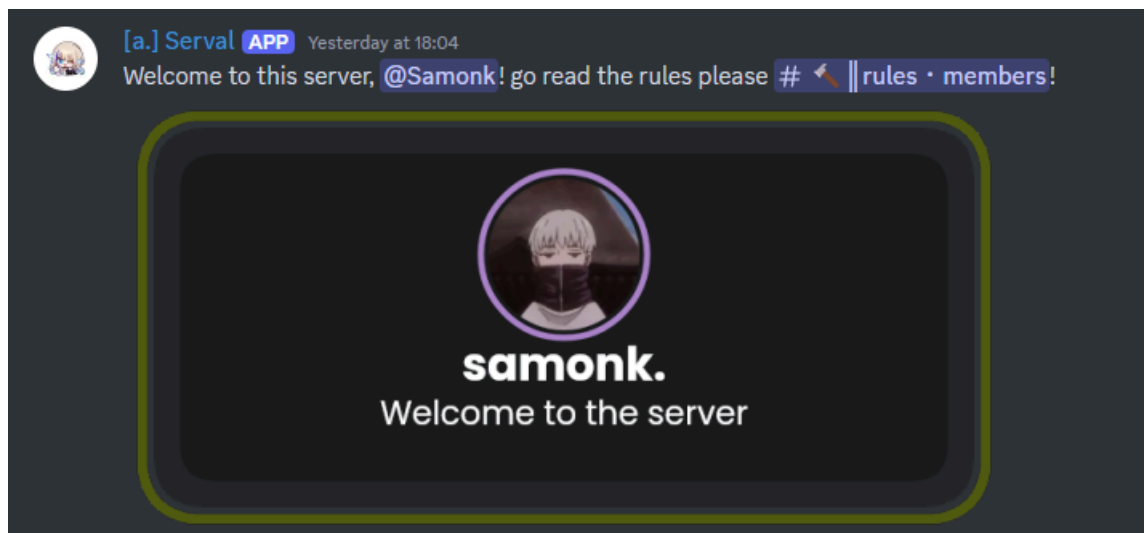
```

184 client.on('guildMemberAdd', async member => {
185   await updateChannelNames(member.guild);
186   const welcomeImage = await new Canvaify.WelcomeLeave()
187     .setAvatar(member.user.displayAvatarURL({ forceStatic: true, extension: "png" }))
188     .setBackground("Image", backgroundBuffer)
189     .setTitle(`${member.user.username}`)
190     .setDescription("Welcome to the server", "#FFFFFF")
191     .setBorder(getRandomColor())
192     .setAvatarBorder(getRandomColor())
193     .setOverlayOpacity(0.3)
194     .build();
195
196   const attachment = new MessageAttachment(welcomeImage, `welcome-${member.id}.png`);
197
198   const channel = member.guild.channels.cache.get(WELCOME_CHANNEL_ID);
199   if (!channel) {
200     console.error("Welcome channel not found.");
201     return;
202   }
203
204   await channel.send({
205     content: `Welcome to this server, ${member}! go read the rules please <@PUT_YOUR_CHANNEL_ID>`,
206     files: [attachment]
207   });
208 }
209
210 })

```

Gambar diatas adalah contoh implementasi pesan selamat datang otomatis.

Dan berikut adalah hasilnya:



Gambar diatas adalah contoh hasil dari implementasi yang sudah dibuat.

Setelah setiap data dan kode diimplementasikan, kami akan menggabungkan modul dan melakukan pengujian seluruh sistem apakah terdapat kesalahan baik dalam sistem kinerja bot atau penulisan percakapan sebelum diluncurkan ke pengguna. Testing masih dilakukan di localhost atau di server Discord dalam channel khusus yang hanya dapat diakses oleh pengembang. Apabila terdapat kesalahan atau kendala dalam sistem kerja chatbot, kami segera memperbaiki masalah tersebut, begitu pula dengan penulisan percakapan.

## PENUTUP

### 3.1. Kesimpulan

Perancangan ini membuktikan bahwa implementasi bot Discord berbasis JavaScript mampu memenuhi kebutuhan pengelolaan komunitas Discord secara efisien. Bot ini berhasil mengotomatisasi tugas-tugas moderasi, pengelolaan event, dan penyediaan layanan interaktif yang membantu menciptakan lingkungan komunitas yang lebih terstruktur dan dinamis. Penggunaan pendekatan berbasis sistem dengan alat bantu seperti Flowchart, *Use Case*, *Activity Diagram*, *Sequence Diagram*, dan lain sebagainya memastikan pengembangan bot dilakukan secara sistematis dan sesuai dengan kebutuhan pengguna.

Hasil pengujian menunjukkan bahwa bot mampu mengurangi beban kerja pengelola komunitas dengan menggantikan proses manual menjadi otomatis, serta memberikan pengalaman yang lebih menarik bagi anggota komunitas. Bot ini juga membuktikan bahwa JavaScript dan Discord.js adalah kombinasi yang efektif untuk membangun sistem otomatisasi berbasis Discord.

Namun, pengembangan ini juga menyadari pentingnya peningkatan performa, terutama dalam lingkungan komunitas dengan jumlah anggota besar. Dengan demikian, hasil perancangan ini diharapkan dapat menjadi dasar bagi pengembangan lebih lanjut untuk menciptakan sistem bot yang lebih optimal dan relevan.

### 3.2. Saran

Untuk pengembangan di masa depan, bot ini dapat ditingkatkan dengan menambahkan fitur-fitur lanjutan seperti integrasi dengan API eksternal, kemampuan interaksi berbasis suara, dan penyesuaian fitur sesuai dengan kebutuhan komunitas yang lebih spesifik. Selain itu, seperti penyediaan dokumentasi, pengembangan keamanan dan pengoptimalan performa bot perlu dilakukan agar tetap stabil meskipun digunakan di komunitas yang sangat besar atau dengan beban kerja yang tinggi.