Panduan Lengkap: Chatbot Elecomp Indonesia

> Dokumentasi ini menjelaskan cara membuat dan menggunakan sistem chatbot berbasis **CodeIgniter 4** + **Gemini AI** sesuai dengan kode yang ada.

Daftar Isi

1. Gambaran Umum
2. Struktur & Komponen
3. Persyaratan & Instalasi
4. Struktur Database
5. Konfigurasi
6. Penjelasan Alur Kerja (Flow)
7. Penjelasan Detail Per File
8. Routing
9. Cara Menggunakan (User Flow)
10. Fitur Suara (Voice Mode)
11. Catatan & Saran

1. Gambaran Umum

Chatbot ini adalah aplikasi web yang memungkinkan pengguna (guest) bercakap-cakap dengan AI berbasis **Google Gemini 2.5 Flash**. Chatbot dirancang khusus untuk keperluan **Elecomp Indonesia** dengan pengetahuan dan aturan yang bisa dikonfigurasi via database.

Teknologi yang digunakan:

Komponen	Teknologi
Backend Framework	CodeIgniter 4 (PHP)
AI Model	Google Gemini 2.5 Flash
Gemini PHP Client	`gemini-api-php/client ^1.7`
Markdown Renderer	`league/commonmark ^2.8`
HTTP Client	`symfony/http-client ^8.0`
Text-to-Speech	ResponsiveVoice JS
Speech-to-Text	Web Speech API (Browser)

2. Struktur & Komponen

```
app/
├── Controllers/
```

```
│   └── ApiController.php           # Controller utama chatbot
├── Models/
│   ├── ChatModel.php               # Model riwayat chat
│   ├── PengetahuanChatbotModel.php  # Model knowledge base
│   └── AturanChatbotModel.php      # Model aturan/system prompt
└── Views/
    └── currentChat.php             # Tampilan halaman chat
```

---

3. Persyaratan & Instalasi

3.1 Persyaratan

- PHP >= 8.1
- CodeIgniter 4
- Composer
- MySQL / MariaDB
- API Key Google Gemini (dari Google AI Studio)

3.2 Install Dependencies via Composer

Tambahkan ke `composer.json` lalu jalankan `composer install`:

```
{
    "require": {
        "gemini-api-php/client": "^1.7",
        "league/commonmark": "^2.8",
        "nyholm/psr7": "^1.8",
        "psr/http-client": "^1.0",
        "symfony/http-client": "^8.0"
    }
}
```

```
composer install
```

atau

```
composer require gemini-api-php/client
composer require league/commonmark
composer require nyholm/psr7
composer require psr/http-client
composer require symfony/http-client
```

3.3 Konfigurasi `.env`

Buka file `.env` di root project CodeIgniter, tambahkan:

```
GEMINI_API_KEY=your_gemini_api_key_here
```

API key ini dipanggil di controller menggunakan:

```
$apiKey = env('GEMINI_API_KEY');
```

---

4. Struktur Database

Buat tiga tabel berikut di database kamu:

4.1 Tabel `chatbot_history`

Menyimpan seluruh riwayat percakapan setiap sesi. Kolom `chat` berisi array JSON dari semua pesan.

```
CREATE TABLE chatbot_history (
    id          INT AUTO_INCREMENT PRIMARY KEY,
    uuid        VARCHAR(100) NOT NULL UNIQUE,
    user        VARCHAR(100) NOT NULL,
    chat        LONGTEXT,           -- disimpan dalam format JSON
    kontak      VARCHAR(255) DEFAULT '-',
    created_at  DATETIME,
    updated_at  DATETIME
);
```

Contoh isi kolom `chat` yang tersimpan di database:

```
[
    {"role": "USER", "text": "Halo"},
    {"role": "BOT",  "text": "Selamat datang di Elecomp, ada yang bisa dibantu?"},
    {"role": "USER", "text": "Saya ingin tanya soal produk"},
    {"role": "BOT",  "text": "Tentu! Produk apa yang ingin Anda tanyakan?"}
]
```

4.2 Tabel `pengetahuan_chatbot`

Menyimpan knowledge base yang diinjeksikan ke dalam prompt AI setiap kali bot menjawab.

```
CREATE TABLE pengetahuan_chatbot (
    id               INT AUTO_INCREMENT PRIMARY KEY,
    nama_pengetahuan VARCHAR(255) NOT NULL,
```

```
    isi_pengetahuan  TEXT NOT NULL,
    created_at       DATETIME,
    updated_at       DATETIME,
    deleted_at       DATETIME NULL
);
```

Contoh isi tabel:

id	nama_pengetahuan	isi_pengetahuan
1	Profil Perusahaan	Elecomp Indonesia adalah perusahaan yang bergerak di bidang elektronik...
2	Produk Unggulan	Kami menjual laptop, PC rakitan, aksesoris komputer...
3	Jam Operasional	Kami buka Senin-Jumat pukul 08.00-17.00 WIB

4.3 Tabel `aturan_chatbot`

Menyimpan aturan/instruksi sistem untuk AI. **Baris pertama (id = 1)** adalah system instruction utama. Baris selanjutnya adalah aturan tambahan.

```
CREATE TABLE aturan_chatbot (
    id     INT AUTO_INCREMENT PRIMARY KEY,
    aturan TEXT NOT NULL
);
```

Contoh isi tabel:

id	aturan
1	Kamu adalah asisten virtual Elecomp Indonesia. Jawab dengan ramah dan profesional dalam Bahasa Indonesia.
2	Jangan menjawab pertanyaan di luar topik Elecomp Indonesia
3	Selalu tawarkan bantuan lebih lanjut di akhir jawaban
4	Jika pengguna meminta dihubungi, minta kontak (email/nomor HP) mereka

5. Konfigurasi

5.1 Routing (`app/Config/Routes.php`)

```
// Halaman awal chatbot — membuat sesi baru
$routes->get('/chatbot/', [ApiController::class, "newChat"]);

// Halaman percakapan berdasarkan UUID sesi
$routes->get('/chatbot/c/(:segment)', [ApiController::class, "index/$1"]);
```

```
// API endpoint
$routes->group('api/v1', function () use ($routes) {
    $routes->post('send', [ApiController::class, 'sendAfterPrompt']);
    $routes->post('send/(:segment)/user', [ApiController::class,
'sendAfterPrompt/$1']);
    $routes->get('send/(:segment)/bot', [ApiController::class, 'send/$1']);
});

// Mengakhiri dan menghapus sesi chat
$routes->get('/chatbot/c/(:segment)/stop', [ApiController::class, "stoped/$1"]);
```

---

6. Penjelasan Alur Kerja (Flow)

```
Pengguna akses /chatbot/
        │
        ▼
[newChat()] — Buat sesi baru (UUID acak), simpan ke DB, redirect ke
/chatbot/c/{uuid}
        │
        ▼
[index($id)] — Tampilkan halaman chat, load riwayat dari DB
        │
        ▼
Pengguna ketik pesan & submit form  →  POST /api/v1/send/{uuid}/user
        │
        ▼
[sendAfterPrompt($id)] — Simpan pesan USER ke DB
        │   set flashdata 'ai' = 'onMessage'
        │   redirect ke /chatbot/c/{uuid}
        ▼
[index($id)] — Halaman chat reload, triggerAI aktif
        │   JS otomatis redirect ke /api/v1/send/{uuid}/bot (setelah 50ms)
        ▼
[send($id)] — Bangun prompt lengkap, panggil Gemini API, simpan respons BOT ke DB
        │   set flashdata 'speak' = 'ongoing'
        │   redirect ke /chatbot/c/{uuid}
        ▼
[index($id)] — Tampilkan pesan BOT, jika voice mode aktif: TTS berbicara
```

Mengapa dua redirect? Sistem memisahkan proses "simpan pesan user" dan "proses AI" agar:

1. Browser tidak double-submit saat reload
2. Bisa menampilkan animasi "Sedang berpikir..." sementara AI memproses
3. Proses AI di-trigger dari JavaScript secara non-blocking

---

7. Penjelasan Detail Per File

7.1 ApiController.php

Constructor — Load Knowledge & Aturan dari DB

```
namespace App\Controllers;

use App\Controllers\BaseController;
use App\Models\AturanChatbotModel;
use App\Models\ChatModel;
use App\Models\PengetahuanChatbotModel;
use GeminiAPI\Client;
use GeminiAPI\Enums\HarmBlockThreshold;
use GeminiAPI\Enums\HarmCategory;
use GeminiAPI\GenerationConfig;
use GeminiAPI\Resources\ModelName;
use GeminiAPI\Resources\Parts\TextPart;
use GeminiAPI\SafetySetting;

class ApiController extends BaseController
{
    private $aturan;
    private $knowladge;

    public function __construct()
    {
        session(); // Inisialisasi session CI4

        // Load semua knowledge base dari DB saat controller dipanggil
        $prompt = new PengetahuanChatbotModel();
        $this->knowladge = $prompt->select(['nama_pengetahuan',
'isi_pengetahuan'])->findAll();

        // Simpan instance model aturan (digunakan di method send())
        $prompt = new AturanChatbotModel();
        $this->aturan = $prompt;
    }
}
```

Method `newChat()` — Membuat Sesi Baru

Dipanggil saat pengguna pertama kali mengakses `/chatbot/`.

```
public function newChat()
{
    $chat = new ChatModel();

    // Pesan awal USER dibuat otomatis oleh sistem
    $data = [];
    $data[] = [
```

```
        'role' => 'USER',
        'text' => 'Halo'
    ];

    $jsonChat = json_encode($data, JSON_UNESCAPED_UNICODE);

    // Generate UUID v4 secara manual menggunakan random_int()
    $uuid = sprintf(
        '%04x%04x-%04x-%04x-%04x-%04x%04x%04x',
        random_int(0, 0xffff),
        random_int(0, 0xffff),
        random_int(0, 0xffff),
        random_int(0, 0x0fff) | 0x4000,
        random_int(0, 0x3fff) | 0x8000,
        random_int(0, 0xffff),
        random_int(0, 0xffff),
        random_int(0, 0xffff)
    );

    // Generate ID guest acak (format: guest-xxxx-xxxx)
    $user = sprintf(
        'guest-%04x%04x-%04x',
        random_int(0, 0xffff),
        random_int(0, 0x0fff) | 0x4000,
        random_int(0, 0x3fff) | 0x8000,
    );

    // Simpan sesi baru ke DB
    $chat->insert(['uuid' => $uuid, 'user' => $user, 'chat' => $jsonChat, 'kontak'
=> '-']);

    // Set flashdata agar halaman chat otomatis trigger bot saat redirect
    session()->setFlashdata('ai', 'onMessage');

    // Redirect menggunakan JS replace (bukan redirect() CI4 biasa)
    // Tujuan: menghindari entri history browser agar tombol back tidak kembali ke
sini
    return $this->response
        ->setHeader('Content-Type', 'text/html')
        ->setBody('<script>window.location.replace("/chatbot/c/' . $uuid . '");
</script>');
}
```

Method `index($id)` — Tampilkan Halaman Chat

```
public function index($id)
{
    // Cegah browser menyimpan cache halaman ini
    // Penting agar tombol back selalu mengambil data terbaru dari server
    $this->response->setHeader('Cache-Control', 'no-cache, no-store, must-
```

```
revalidate');
    $this->response->setHeader('Pragma', 'no-cache');
    $this->response->setHeader('Expires', '0');

    $chat = new ChatModel();
    $chatData = $chat
        ->select('uuid, chat')
        ->where('uuid', $id)
        ->first();

    // Jika sesi tidak ditemukan atau chat kosong, tampilkan array kosong
    if (!$chatData || empty($chatData['chat'])) {
        $messages = [];
    } else {
        $messages = json_decode($chatData['chat'], true);
        $messages = is_array($messages) ? $messages : [];
    }

    return view('currentChat', [
        'result'    => $messages,
        'id'        => $chatData['uuid'],
        'triggerAI' => session()->getFlashdata('ai'),    // trigger animasi
loading + fetch bot
        'speak'     => session()->getFlashdata('speak')  // trigger TTS setelah
bot menjawab
    ]);
}
```

Method `sendAfterPrompt($id)` — Simpan Pesan User ke DB

Dipanggil via POST dari form chat di view.

```
public function sendAfterPrompt($id = null)
{
    $prompt = trim($this->request->getPost('prompt'));
    $kontak = trim($this->request->getPost('kontak'));

    // Jika pesan kosong, abaikan
    if ($prompt === '') {
        return redirect()->back();
    }

    $chat = new ChatModel();

    if ($id !== null) {
        $chatData = $chat->where("uuid", $id)->first();
        $data = json_decode($chatData['chat'], true);
    } else {
        $data = [];
    }
```

```
    $data = is_array($data) ? $data : [];

    // Batasi riwayat hanya 10 pesan terakhir untuk hemat storage
    $data = array_slice($data, -10);

    // Tambahkan pesan baru dari user
    $data[] = [
        'role' => 'USER',
        'text' => $prompt
    ];

    $jsonChat = json_encode($data, JSON_UNESCAPED_UNICODE);

    if ($id !== null && isset($chatData)) {
        // Simpan pesan + update kontak jika user memasukkan email/HP
        $chat->update($chatData['id'], ['chat' => $jsonChat, 'kontak' =>
$kontak]);
        $chatId = $chatData['uuid'];
    }

    // Set trigger agar halaman chat otomatis memanggil endpoint bot
    session()->setFlashdata('ai', 'onMessage');

    return $this->response
        ->setHeader('Content-Type', 'text/html')
        ->setBody('<script>window.location.replace("/chatbot/c/' . $chatId . '");
</script>');
}
```

Method `send($id)` — Proses AI & Simpan Respons Bot

Ini method terpenting. Diakses via GET oleh JavaScript setelah halaman chat load.

Langkah 1 — Load dan siapkan konteks percakapan:

```
$chat = new ChatModel();
$chatData = $chat->where("uuid", $id)->first();
$data = json_decode($chatData['chat'], true);
$data = is_array($data) ? $data : [];

// Salin untuk dijadikan konteks (tanpa pesan terakhir yang akan dijawab)
$context = $data;
if (!empty($context)) {
    array_pop($context); // Hapus pesan user terbaru dari konteks
}

// Ambil hanya 10 pesan terakhir sebagai window konteks
$context = array_slice($context, -10);
```

```
// Format konteks menjadi string teks
$contextText = "";
foreach ($context as $msg) {
    $contextText .= "{$msg['role']}: {$msg['text']}\n";
}

// Ambil pesan user paling baru (yang akan dijawab bot)
$lastChat = $data[count($data) - 1]['role'] == "USER"
    ? $data[count($data) - 1]['text']
    : null;
```

Langkah 2 — Ambil aturan dan pengetahuan dari DB:

```
// System instruction utama (baris id = 1)
$systemInstruction = $this->aturan->select(['aturan'])->first();

// Aturan tambahan (semua baris kecuali id = 1)
$aturanAll = $this->aturan->select(['aturan'])->where('id !=', 1)->findAll();
$aturan = '';
foreach ($aturanAll as $row) {
    $aturan .= '- ' . $row['aturan'] . PHP_EOL;
}

// Susun semua knowledge base menjadi satu teks
$pengetahuan = '';
foreach ($knowladge as $index => $row) {
    $pengetahuan .= ($index + 1) . $row['nama_pengetahuan'] . PHP_EOL .
$row['isi_pengetahuan'];
}
```

Langkah 3 — Susun prompt lengkap (Heredoc):

```
$fullPrompt = <<<PROMPT
[SYSTEM INSTRUCTION]
{$systemInstruction['aturan']}

[KNOWLADGE]
$pengetahuan

ATURAN WAJIB:
$aturan

[CONTEXT]
$contextText

Pertanyaan Baru:
[PROMPT]
$lastChat
PROMPT;
```

Contoh hasil prompt yang dikirim ke Gemini:

```
[SYSTEM INSTRUCTION]
Kamu adalah asisten virtual Elecomp Indonesia. Jawab dengan ramah dan profesional
dalam Bahasa Indonesia.

[KNOWLADGE]
1Profil Perusahaan
Elecomp Indonesia adalah perusahaan yang bergerak di bidang elektronik...
2Produk Unggulan
Kami menjual laptop, PC rakitan, aksesoris komputer...

ATURAN WAJIB:
- Jangan menjawab pertanyaan di luar topik Elecomp Indonesia
- Selalu tawarkan bantuan lebih lanjut di akhir jawaban

[CONTEXT]
BOT: Selamat datang di Elecomp, ada yang bisa dibantu?

Pertanyaan Baru:
[PROMPT]
Apakah kalian jual laptop gaming?
```

Langkah 4 — Konfigurasi dan panggil Gemini API:

```
// Konfigurasi parameter generasi model
$generationConfig = (new GenerationConfig())
    ->withCandidateCount(1)      // Hanya 1 kandidat jawaban
    ->withTemperature(0.5)       // 0 = deterministik, 1 = lebih kreatif
    ->withTopK(40)
    ->withTopP(0.95)
    ->withStopSequences(['STOP']);

try {
    // Gemini hanya dipanggil jika chat sudah lebih dari 2 entry
    // Sesi baru hanya punya 1 pesan "Halo" → jawab dengan teks statis
    if (count($data) > 2) {
        $client = new Client($apiKey);

        $response = $client->generativeModel(ModelName::GEMINI_2_5_FLASH)
            ->withAddedSafetySetting(
                new SafetySetting(
                    HarmCategory::HARM_CATEGORY_HARASSMENT,
                    HarmBlockThreshold::BLOCK_MEDIUM_AND_ABOVE
                )
            )
            ->withAddedSafetySetting(
                new SafetySetting(
                    HarmCategory::HARM_CATEGORY_HATE_SPEECH,
```

```
                    HarmBlockThreshold::BLOCK_LOW_AND_ABOVE
                )
            )
            ->withAddedSafetySetting(
                new SafetySetting(
                    HarmCategory::HARM_CATEGORY_SEXUALLY_EXPLICIT,
                    HarmBlockThreshold::BLOCK_ONLY_HIGH
                )
            )
            ->withAddedSafetySetting(
                new SafetySetting(
                    HarmCategory::HARM_CATEGORY_DANGEROUS_CONTENT,
                    HarmBlockThreshold::BLOCK_LOW_AND_ABOVE
                )
            )
            ->withGenerationConfig($generationConfig)
            ->generateContent(
                new TextPart($fullPrompt)
            );

        $text = $response->text(); // Ambil teks dari respons Gemini
    } else {
        // Pesan sambutan default untuk sesi baru
        $text = "Selamat datang di Elecomp, ada yang bisa dibantu?";
    }
} catch (\Exception $e) {
    // Fallback jika Gemini API error / rate limit
    $text = "Model sedang sibuk, silahkan coba lagi nanti.";
    log_message('debug', 'BOT Error: ' . $e->getMessage());
}
```

Langkah 5 — Simpan respons dan redirect:

```
// Tambahkan respons bot ke array chat
$data[] = [
    'role' => 'BOT',
    'text' => $text
];

$jsonChat = json_encode($data, JSON_UNESCAPED_UNICODE);

// Update DB dengan respons bot terbaru
$chat->update($chatData['id'], ['chat' => $jsonChat]);
$chatId = $chatData['uuid'];

// Set flashdata untuk trigger TTS di client jika call mode aktif
session()->setFlashdata('speak', 'ongoing');

return $this->response
    ->setHeader('Content-Type', 'text/html')
    ->setBody('<script>window.location.replace("/chatbot/c/' . $chatId . '");
</script>');
```

Method `stoped($id)` — Akhiri dan Hapus Sesi

```
public function stoped($id)
{
    $modelChat = new ChatModel();
    $chatData = $modelChat->where('uuid', $id)->first();

    // Hapus data HANYA jika pengguna tidak pernah meninggalkan kontak
    // Jika kontak ada (bukan "-"), data disimpan untuk keperluan follow-up
    if ($chatData && $chatData['kontak'] === "-") {
        $modelChat->where('uuid', $id)->delete();
    }

    return $this->response
        ->setHeader('Content-Type', 'text/html')
        ->setBody('<script>window.location.replace("/");</script>');
}
```

7.2 Models

`ChatModel.php`

```
<?php

namespace App\Models;

use CodeIgniter\Model;

class ChatModel extends Model
{
    protected $table            = 'chatbot_history';
    protected $primaryKey       = 'id';
    protected $useAutoIncrement = true;
    protected $returnType       = 'array';
    protected $useSoftDeletes   = false;
    protected $protectFields    = true;
    protected $allowedFields    = ["uuid", "user", "chat", "kontak"];
    protected $useTimestamps    = true;
    protected $dateFormat       = 'datetime';
    protected $createdField     = 'created_at';
    protected $updatedField     = 'updated_at';
}
```

`PengetahuanChatbotModel.php`

```php
<?php

namespace App\Models;

use CodeIgniter\Model;

class PengetahuanChatbotModel extends Model
{
    protected $table         = 'pengetahuan_chatbot';
    protected $primaryKey    = 'id';
    protected $returnType    = 'array';
    protected $useSoftDeletes = false;
    protected $protectFields = true;
    protected $allowedFields = ['nama_pengetahuan', 'isi_pengetahuan'];
    protected $useTimestamps = true;
    protected $dateFormat    = 'datetime';
    protected $createdField  = 'created_at';
    protected $updatedField  = 'updated_at';
    protected $deletedField  = 'deleted_at';
}
```

`AturanChatbotModel.php`

```php
<?php

namespace App\Models;

use CodeIgniter\Model;

class AturanChatbotModel extends Model
{
    protected $table         = 'aturan_chatbot';
    protected $primaryKey    = 'id';
    protected $returnType    = 'array';
    protected $protectFields = true;
    protected $allowedFields = ['aturan'];
    protected $useTimestamps = false; // Tabel ini tidak menggunakan timestamps
}
```

7.3 `currentChat.php` (View)

Load Library dan Inisialisasi Converter Markdown

```php
<head>
    <!-- ResponsiveVoice untuk Text-to-Speech -->
    <script src="https://code.responsivevoice.org/responsivevoice.js?
```

```
key=YT8cCBCv"></script>

    <!-- Boxicons untuk ikon UI -->
    <link href='https://cdn.boxicons.com/3.0.6/fonts/basic/boxicons.min.css'
rel='stylesheet'>

    <link rel="stylesheet" href="<?= base_url('assets/css/chatbot.css') ?>">
</head>
```

```
<?php
use League\CommonMark\CommonMarkConverter;

// Converter Markdown → HTML, soft break menjadi <br>
$converter = new CommonMarkConverter([
    'renderer' => [
        'soft_break' => "<br />\n",
    ],
]);
?>
```

Rendering Bubble Chat

```
<div class="chat-container" id="chatBody">
    <?php if (count($result) > 0): ?>
        <?php foreach ($result as $msg): ?>
            <div class="bubble <?= strtolower($msg['role']) ?>">
                <?php if ($msg['role'] === 'BOT'): ?>
                    <!-- Pesan BOT: render sebagai Markdown → HTML (mendukung
bold, list, link, dll) -->
                    <?= $converter->convert($msg['text']) ?>
                <?php else: ?>
                    <!-- Pesan USER: plain text, di-escape dari XSS, newline jadi
<br> -->
                    <?= nl2br(htmlspecialchars($msg['text'])) ?>
                <?php endif; ?>
            </div>
        <?php endforeach; ?>
    <?php endif; ?>
</div>
```

Form Input Chat

```
<form class="chat-input" action="/api/v1/send/<?= $id ?>/user" method="post">
    <textarea
        name="prompt"
        id="prompt"
```

```
            rows="1"
            maxlength="500"
            placeholder="Tulis pesan..."
            autocomplete="off">
        </textarea>

        <!-- Hidden field: otomatis terisi jika user memasukkan kontak valid -->
        <input type="text" name="kontak" id="kontak" value="-" hidden>

        <button class="send-btn" type="submit">Send</button>
    </form>
```

Submit dengan Enter (Desktop) dan Textarea Auto-resize

```
const textarea = document.getElementById('prompt');
const form = textarea.closest('form');
const isMobile = /Android|iPhone|iPad|iPod/i.test(navigator.userAgent);

// Enter untuk kirim di desktop, Shift+Enter untuk baris baru
textarea.addEventListener('keydown', function(e) {
    if (!isMobile && e.key === 'Enter' && !e.shiftKey) {
        e.preventDefault();
        form.submit();
    }
});

// Auto-resize textarea sampai maksimal 160px
const MAX_HEIGHT = 160;
textarea.addEventListener('input', () => {
    textarea.style.height = 'auto';
    const newHeight = Math.min(textarea.scrollHeight, MAX_HEIGHT);
    textarea.style.height = newHeight + 'px';
    textarea.style.overflowY = textarea.scrollHeight > MAX_HEIGHT ? 'auto' :
'hidden';
});
```

Trigger AI Otomatis (Flashdata `triggerAI`)

Setelah halaman load dan flashdata `'ai'` aktif:

```
<?php if (!empty($triggerAI)): ?>
    <script>
        const sendBtn = document.querySelector(".send-btn")
        const modeCall = document.querySelector(".mode-call")
        const chatBody = document.getElementById("chatBody")

        // Nonaktifkan tombol send agar user tidak bisa kirim pesan selagi bot
berpikir
```

```
        sendBtn.disabled = true;

        // Tampilkan bubble animasi loading
        let bubble = document.createElement("div")
        bubble.setAttribute("class", "bubble bot thinking")
        bubble.textContent = "Sedang berpikir..."
        chatBody.appendChild(bubble);

        // Tampilkan teks di area call mode
        let think = document.createElement("p")
        think.setAttribute("class", "think")
        think.textContent = "Sedang berpikir..."
        modeCall.prepend(think);

        // Setelah 50ms, redirect ke endpoint bot (GET) untuk proses AI
        setTimeout(() => {
            window.location.replace("/api/v1/send/<?= $id ?>/bot");
        }, 50);
    </script>
<?php endif; ?>
```

Trigger TTS Setelah Bot Menjawab (Flashdata `speak`)

```
<?php if (!empty($speak)): ?>
    <script>
        call_mode = localStorage.getItem('isCall') == "true";
        setTimeout(() => {
            if (call_mode) {
                speak(); // Jalankan fungsi TTS jika call mode aktif
            }
        }, 100)
    </script>
<?php endif; ?>
```

Deteksi Input Kontak User

Jika pesan bot terakhir mengandung kata **"kontak"** atau **"dihubungi"**, script ini aktif:

```
const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]{2,}$/;
const phoneRegex = /^(?:\+62|62|0)8[1-9][0-9]{7,10}$/;

// Cek apakah pesan bot terakhir meminta kontak
const lastBot =
    lastBubble.classList.contains("bot") &&
    /kontak|dihubungi/i.test(lastBubble.textContent);

if (lastBot) {
    textarea.addEventListener('input', () => {
```

```
        const value = textarea.value.replace(/[\s-]/g, '');

        if (!value) {
            document.getElementById("kontak").value = '';
            return;
        }

        // Jika cocok email atau nomor HP Indonesia → simpan ke hidden field
        const isKontak = emailRegex.test(value) || phoneRegex.test(value);
        document.getElementById("kontak").value = isKontak ? value : '-';
    });
}
```

Scroll Otomatis ke Pesan Terakhir

```
const offset = 40;
const top = lastBubble.offsetTop - container.offsetTop - offset;

container.scrollTo({
    top,
    behavior: 'smooth'
});
```

Buka Semua Link dari Bot di Tab Baru

```
// Link yang dihasilkan dari Markdown respons bot dibuka di tab baru
container.querySelectorAll("a").forEach(a => a.setAttribute("target", "_blank"));
```

Cegah Halaman Stale saat Tombol Back Ditekan

```
window.addEventListener('pageshow', function(event) {
    // Jika halaman dimuat dari cache back/forward browser
    if (event.persisted ||
        (window.performance &&
            performance.getEntriesByType("navigation")[0]?.type ===
'back_forward')) {

        // Reset textarea
        const textarea = document.getElementById('prompt');
        if (textarea) {
            textarea.value = '';
            textarea.style.height = 'auto';
        }

        // Paksa reload untuk mengambil data terbaru dari server
        window.location.reload();
```

```
    }
});
```

---

8. Routing

Method	URL	Handler	Fungsi
GET	`/chatbot/`	`newChat()`	Buat sesi baru
GET	`/chatbot/c/{uuid}`	`index($id)`	Tampilkan halaman chat
POST	`/api/v1/send/{uuid}/user`	`sendAfterPrompt($id)`	Simpan pesan user ke DB
GET	`/api/v1/send/{uuid}/bot`	`send($id)`	Proses AI & simpan respons bot
GET	`/chatbot/c/{uuid}/stop`	`stoped($id)`	Akhiri sesi, hapus data jika tidak ada kontak

---

9. Cara Menggunakan (User Flow)

Langkah 1: Mulai Chat

Akses URL:

```
http://yourdomain.com/chatbot/
```

Sistem otomatis membuat UUID sesi baru, menyimpan pesan awal `"Halo"`, dan menampilkan pesan sambutan bot.

Langkah 2: Kirim Pesan

- Ketik pesan di textarea
- **Desktop:** tekan `Enter` untuk kirim (tanpa `Shift`)
- **Mobile:** klik tombol `Send`
- Halaman reload, tampil bubble "Sedang berpikir..."
- Bot otomatis memproses dan membalas

Langkah 3: Mode Suara (Opsional)

Lihat Bagian 10 untuk detail.

Langkah 4: Keluar

Klik tombol ← di navbar. Jika `kontak === '-'` → riwayat chat **dihapus** dari DB. Jika kontak sudah diisi → data **tetap disimpan** untuk follow-up.

---

10. Fitur Suara (Voice Mode)

10.1 Text-to-Speech (TTS) — Bot Berbicara

```
speak = () => {
    if (isListening) return; // Jangan bicara kalau user sedang merekam

    imgBot.classList.add('onSpeak'); // Tambah class animasi visual

    // Ambil teks dari bubble bot terakhir, bersihkan whitespace berlebih
    const text = lastBubble.classList.contains('bot')
        ? lastBubble.textContent.replace(/\n{2,}/g, '\n').trim()
        : '';

    isSpeaking = true;
    responsiveVoice.speak(text, "Indonesian Female", {
        rate: 1.2,   // Kecepatan bicara (1.0 = normal)
        pitch: 1,    // Nada suara
        volume: 1,   // Volume penuh
        onend: () => {
            isSpeaking = false;
            imgBot.classList.remove('onSpeak'); // Hentikan animasi
        }
    });
}
```

10.2 Speech-to-Text (STT) — User Berbicara

```
function checkBrowserSupport() {
    if (!('webkitSpeechRecognition' in window) && !('SpeechRecognition' in
window)) {
        alert('Browser Anda tidak mendukung Speech Recognition. Coba gunakan
Chrome atau Edge.');
        return false;
    }
    return true;
}

function initRecognition() {
    const SpeechRecognition = window.SpeechRecognition ||
window.webkitSpeechRecognition;
    recognition = new SpeechRecognition();

    recognition.continuous     = true;    // Terus dengarkan tanpa berhenti
otomatis
    recognition.interimResults = true;    // Tampilkan hasil sementara (belum
final)
    recognition.lang           = 'id-ID'; // Bahasa Indonesia

    recognition.onstart = function() {
        isListening = true;
        console.log("listening");
```

```
    };

    recognition.onresult = function(event) {
        let finalTranscript = '';

        for (let i = event.resultIndex; i < event.results.length; i++) {
            const transcript = event.results[i][0].transcript;
            if (event.results[i].isFinal) {
                finalTranscript += transcript + ' ';
            }
        }

        // Tambahkan transkripsi akhir ke textarea
        if (finalTranscript) {
            const textarea = document.getElementById('prompt');
            textarea.value += finalTranscript;
            textarea.scrollTop = textarea.scrollHeight;
        }
    };

    recognition.onerror = function(event) {
        console.error('Error:', event.error);
        stopListening();
    };

    recognition.onend = function() {
        stopListening();
    };
}

function startListening() {
    if (!checkBrowserSupport()) return;
    if (isListening) return;
    if (!recognition) initRecognition();

    try {
        recognition.start();
    } catch (error) {
        console.error('Error starting:', error);
        alert('Gagal memulai. Pastikan microphone tersedia dan diizinkan.');
    }
}

function stopListening() {
    if (recognition && isListening) {
        recognition.stop();
        isListening = false;
    }
}
```

10.3 Interaksi Call Mode (Push-to-Talk)

Status call mode disimpan di `localStorage` agar persisten antar reload halaman:

```
// Aktifkan call mode saat ikon telepon di navbar diklik
callingMe.onclick = () => {
    if (!checkBrowserSupport()) return;
    if (!recognition) initRecognition();

    startListening();
    callMode.style.display = "flex";
    localStorage.setItem('isCall', "true");
}

// Pointerdown pada gambar bot = mulai rekam suara
onHoldCall.addEventListener('pointerdown', function(e) {
    let goals = e.target;

    // Jika TTS sedang berjalan, hentikan dulu
    if (isSpeaking) {
        responsiveVoice.cancel();
        isSpeaking = false;
    }

    if (goals == onHoldCall || goals == onHoldCall.querySelector("img")) {
        startListening(); // Mulai rekam
    } else if (goals == endCall || goals.closest('.end_call') !== null) {
        callMode.style.display = "none";
        localStorage.setItem('isCall', "false"); // Matikan call mode
        imgBot.classList.remove('onSpeak');
    }
});

// Pointerup pada gambar bot = selesai rekam, kirim pesan
onHoldCall.addEventListener('pointerup', (e) => {
    let goals = e.target;
    if (goals == onHoldCall || goals == onHoldCall.querySelector("img")) {
        stopListening();

        // Delay 500ms agar transkripsi terakhir sempat masuk ke textarea
        setTimeout(() => {
            form.submit();
        }, 500);

        localStorage.setItem('isCall', "true");
    }
});
```

---

11. Catatan & Saran

⚠ Hal yang Perlu Diperhatikan (Based on Code)

1. Kondisi `count($data) > 2` di `send()`

Gemini hanya dipanggil jika array chat sudah lebih dari 2 entry. Pada sesi baru, array hanya berisi 1 pesan `USER: "Halo"`, sehingga bot selalu menjawab dengan teks statis:

```
if (count($data) > 2) {
    // Panggil Gemini API
} else {
    $text = "Selamat datang di Elecomp, ada yang bisa dibantu?"; // Statis
}
```

Jika ingin Gemini menjawab dari pesan pertama, ubah kondisinya:

```
if (count($data) >= 1) { // ubah dari > 2 menjadi >= 1
```

---

2. Dua fungsi helper tidak digunakan

Fungsi `findKeywordIndex()` dan `findKeywordTrigger()` ada di controller tapi tidak dipanggil di method manapun. Aman untuk dihapus.

```
// Kedua fungsi ini tidak dipakai di manapun — aman dihapus
function findKeywordIndex(string $text, array $arr): ?int
{
    foreach ($arr as $i => $group) {
        foreach ($group as $word) {
            if (stripos($text, $word) !== false) {
                return $i;
            }
        }
    }
    return null;
}

function findKeywordTrigger(string $text, array $arr): ?int
{
    foreach ($arr as $word) {
        if (stripos($text, $word) !== false) {
            return true;
        }
    }
    return false;
}
```

---

3. Variabel `$index` di `send()` tertimpa oleh foreach

```
// Di-set dari session...
$index = session()->get('q_index');

// ...tapi langsung tertimpa oleh loop ini:
foreach ($knowladge as $index => $row) { // $index di-overwrite di sini
    $pengetahuan .= ($index + 1) . $row['nama_pengetahuan'] . ...;
}
```

Ganti nama variabel loop agar tidak konflik:

```
foreach ($knowladge as $i => $row) {
    $pengetahuan .= ($i + 1) . $row['nama_pengetahuan'] . PHP_EOL .
$row['isi_pengetahuan'];
}
```

---

4. Tidak ada validasi kepemilikan sesi

UUID ada di URL secara publik. Siapapun yang punya URL bisa membaca dan mengirim pesan ke sesi tersebut karena controller tidak memvalidasi kepemilikan.

---

5. ResponsiveVoice memerlukan domain terdaftar di production

Key `YT8cCBCv` di script tag adalah key akun ResponsiveVoice. TTS berfungsi normal di localhost, tapi di production kamu perlu mendaftarkan domain di dashboard ResponsiveVoice agar tidak diblokir.

Saran Pengembangan

1. Ganti redirect berantai dengan AJAX

Pola saat ini: form submit → full page reload → JS fetch ke bot → full page reload lagi. Ini bisa diganti dengan `fetch()` API agar lebih mulus tanpa ketergantungan pada flashdata session.

2. Tambahkan rate limiting

Saat ini tidak ada pembatasan berapa kali pengguna bisa mengirim pesan. Pertimbangkan throttle middleware CI4 untuk endpoint `/api/v1/send/{uuid}/bot`.

3. Pisahkan penyimpanan pesan per baris

Semua pesan tersimpan sebagai satu JSON di satu kolom `LONGTEXT`. Untuk analitik dan pencarian yang lebih baik, pertimbangkan tabel `chat_messages` terpisah dengan kolom `session_id`, `role`, `text`, `created_at`.

4. Gunakan Gemini streaming

Library `gemini-api-php/client` mendukung streaming response. Menggunakannya akan memberikan efek bot "mengetik" secara real-time, meningkatkan UX tanpa harus menunggu seluruh respons selesai diproses.