

BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Kedudukan dalam struktur organisasi PT Dayalima Abisatya berada pada Divisi *IT Department*, yaitu unit yang memiliki peran sentral dalam mendukung operasional perusahaan melalui pengembangan sistem, integrasi teknologi, serta pemeliharaan infrastruktur digital. *IT Department* merupakan bagian penting dari proses digitalisasi internal yang menjadi fokus strategis perusahaan dalam meningkatkan efisiensi administrasi dan kualitas pelayanan. Melalui divisi ini, berbagai kebutuhan teknologi dari departemen lain dapat difasilitasi secara terstruktur dan terintegrasi, sehingga seluruh proses kerja perusahaan dapat berjalan secara lebih efektif dan terdokumentasi dengan baik.

Secara hierarkis, posisi kerja magang berada di bawah koordinasi langsung *Head of IT Department*, yang bertanggung jawab dalam merancang arah teknis pengembangan sistem, menetapkan prioritas proyek, serta memastikan setiap solusi teknologi yang dikembangkan selaras dengan standar dan kebutuhan perusahaan. *Head of IT Department* memiliki tanggung jawab langsung kepada Direksi, sehingga setiap kegiatan dan proyek yang dijalankan dalam divisi ini memiliki keterkaitan erat dengan kebijakan strategis dan arah pengembangan perusahaan secara keseluruhan. Struktur hierarkis ini menunjukkan adanya alur koordinasi yang jelas antara fungsi teknis dan manajerial dalam mendukung transformasi digital perusahaan.

Dalam pelaksanaan kerja sehari-hari, kegiatan magang dilakukan sebagai bagian dari tim *IT Support DDI* (Daya Dimensi Indonesia), yang merupakan unit operasional di bawah *IT Department*. Tim ini berfungsi sebagai garda terdepan dalam menangani kebutuhan teknologi informasi internal, khususnya yang berkaitan dengan pengembangan sistem dan otomatisasi proses kerja. Tim *IT Support DDI* terdiri dari empat orang karyawan tetap yang memiliki tanggung jawab utama dalam pengelolaan operasional IT, serta dua orang mahasiswa magang

yang dilibatkan secara aktif dalam kegiatan pengembangan sistem internal perusahaan.

Keterlibatan mahasiswa magang dalam tim *IT Support* DDI memberikan kontribusi nyata dalam mendukung kegiatan pengembangan sistem, terutama pada proyek-proyek yang bersifat mendesak dan membutuhkan solusi dalam waktu relatif singkat. Dalam struktur kerja tim, karyawan tetap berperan sebagai pengawas teknis dan penanggung jawab operasional, sementara mahasiswa magang diberikan kepercayaan untuk terlibat langsung dalam proses perancangan dan pengembangan sistem. Pembagian peran ini memungkinkan proses kerja berjalan lebih efisien, sekaligus memberikan ruang pembelajaran yang optimal bagi mahasiswa magang dalam memahami praktik kerja profesional di bidang teknologi informasi.

Untuk memperjelas struktur tim dan kedudukan kegiatan magang dalam pelaksanaan kerja sehari-hari, berikut ditampilkan dokumentasi tim *IT Support* DDI PT Dayalima Abisatya yang terlibat langsung dalam kegiatan operasional dan pengembangan sistem internal perusahaan.



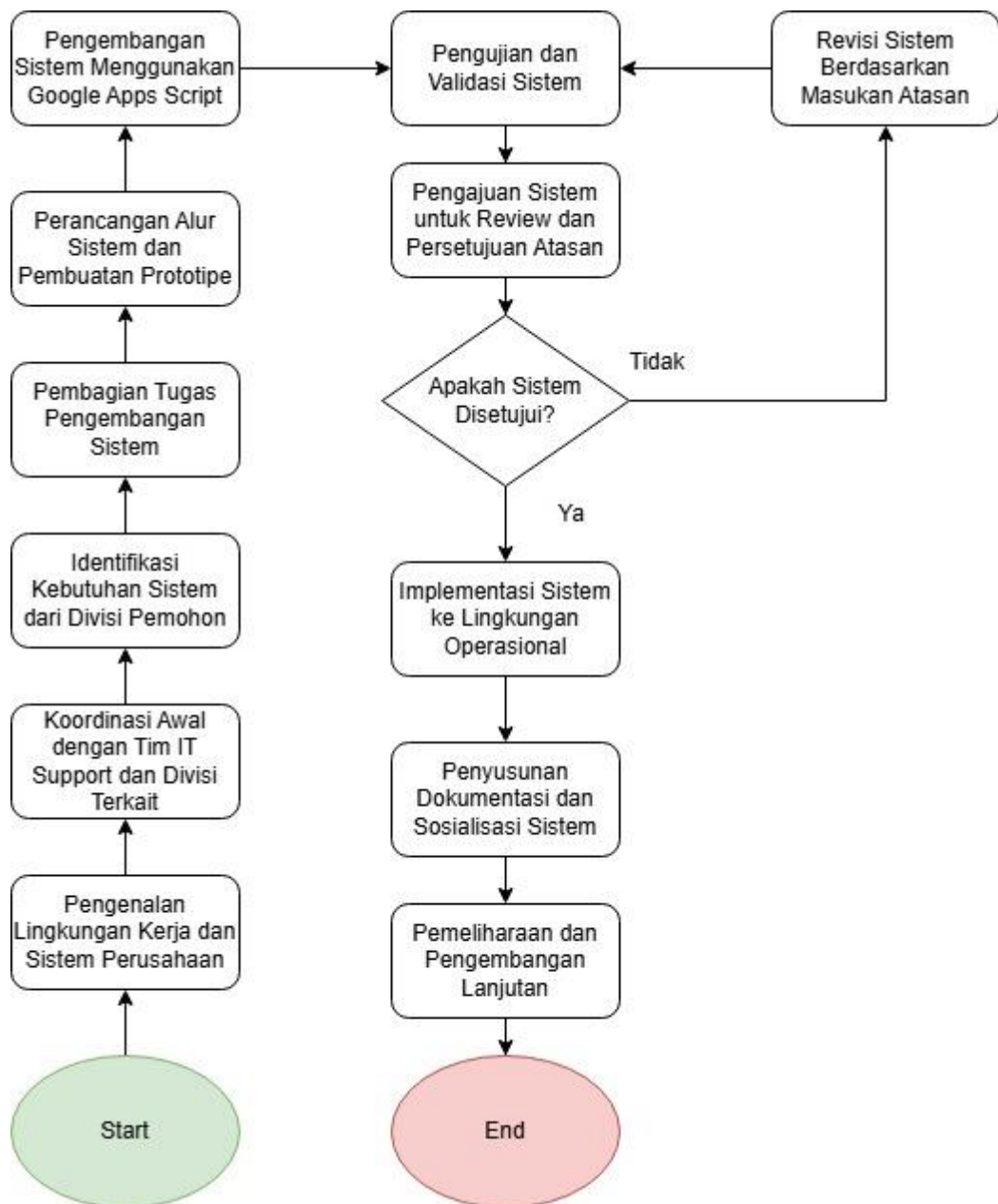
Gambar 3.1 Tim IT Support PT Dayalima Abisatya

Berdasarkan Gambar 3.1, terlihat struktur tim yang terlibat langsung dalam pelaksanaan kerja magang, di mana mahasiswa magang menjadi bagian dari tim *IT Support* DDI dan berkontribusi aktif dalam mendukung pengembangan sistem otomatisasi internal. Kolaborasi antara karyawan tetap dan mahasiswa magang memudahkan terjadinya pertukaran pengetahuan, diskusi teknis, serta koordinasi kerja yang mendukung penyelesaian proyek secara lebih efektif dan terarah.

Kedudukan sebagai *programmer intern* menempatkan peran ini sebagai bagian dari proses pengembangan sistem internal perusahaan, khususnya dalam pembuatan dan otomatisasi alur administrasi berbasis Google Apps Script. Posisi ini mencakup keterlibatan langsung dalam berbagai aktivitas teknis, seperti analisis kebutuhan pengguna, perancangan alur kerja digital, pengembangan kode program, integrasi sistem dengan Google Workspace, hingga pengujian fungsionalitas sistem. Keberadaan posisi ini berkontribusi dalam mempercepat proses digitalisasi internal dan mengurangi ketergantungan perusahaan terhadap proses manual.

Model kerja yang diterapkan dalam tim *IT Support* DDI bersifat responsif terhadap kebutuhan divisi lain. Setiap divisi di lingkungan PT Dayalima Abisatya yang membutuhkan sistem baru atau pengembangan sistem akan menyampaikan permintaan secara langsung kepada tim *IT Support*. Permintaan tersebut dapat berupa kebutuhan sistem *booking*, sistem pencatatan, sistem *approval*, maupun sistem pendukung operasional lainnya. Seluruh permintaan dicatat dan dianalisis untuk memastikan solusi yang dikembangkan sesuai dengan kebutuhan pengguna dan kondisi operasional yang ada.

Untuk menggambarkan secara lebih jelas alur pelaksanaan kerja magang dalam pengembangan sistem, digunakan sebuah *flowchart* yang menunjukkan tahapan kerja sejak koordinasi awal hingga pemeliharaan sistem. *Flowchart* ini merepresentasikan proses kerja yang dijalankan selama magang, termasuk pembagian tugas, proses pengembangan, pengujian, persetujuan atasan, hingga implementasi sistem ke lingkungan operasional perusahaan.



Gambar 3.2 Flowchart Alur Pelaksanaan Kerja Magang di Divisi IT Department PT Dayalima Abisatya

Berdasarkan Gambar 3.2, alur pelaksanaan kerja magang dimulai dari tahap pengenalan lingkungan kerja dan sistem perusahaan, yang dilanjutkan dengan koordinasi awal bersama Tim IT Support dan divisi terkait. Setelah kebutuhan sistem dari divisi pemohon teridentifikasi, dilakukan pembagian tugas pengembangan sistem untuk memastikan setiap tahapan pengerjaan berjalan secara terstruktur dan efisien.

Tahap berikutnya mencakup perancangan alur sistem dan pembuatan *prototype*, yang kemudian dilanjutkan dengan pengembangan sistem menggunakan Google Apps Script. Sistem yang telah dikembangkan selanjutnya melalui proses pengujian dan validasi untuk memastikan kesesuaian dengan kebutuhan pengguna. Apabila sistem belum disetujui oleh atasan, maka dilakukan proses revisi berdasarkan masukan yang diberikan, sebelum kembali diuji dan diajukan ulang untuk persetujuan.

Setelah sistem memperoleh persetujuan, dilakukan implementasi ke lingkungan operasional perusahaan, disertai dengan penyusunan dokumentasi dan kegiatan sosialisasi kepada pengguna. Tahap akhir dari alur kerja ini adalah pemeliharaan dan pengembangan lanjutan sistem, guna memastikan sistem tetap berjalan dengan baik serta dapat disesuaikan dengan kebutuhan perusahaan di masa mendatang. Alur kerja yang terstruktur ini mendukung pelaksanaan magang yang sistematis, terkontrol, dan selaras dengan tujuan transformasi digital PT Dayalima Abisatya.

Dengan demikian, kegiatan magang yang dilaksanakan dalam divisi *IT Department* melalui tim *IT Support* DDI menjadi bagian integral dari upaya perusahaan dalam membangun sistem kerja yang lebih efisien, adaptif, dan berbasis teknologi digital.

3.2 Tugas dan Uraian Kerja Magang

Selama masa pelaksanaan kegiatan magang di PT Dayalima Abisatya, fokus utama pekerjaan terletak pada proses digitalisasi administrasi perusahaan melalui pengembangan berbagai sistem otomatisasi berbasis Google Apps Script. Seluruh sistem yang dirancang ditujukan untuk menggantikan proses manual yang sebelumnya dilakukan menggunakan email, percakapan langsung, ataupun pencatatan di Excel, sehingga aktivitas administrasi dapat berjalan lebih efisien, terdokumentasi, dan mudah dipantau oleh setiap unit terkait.

Rangkaian tugas dalam kegiatan magang dilakukan secara bertahap, dimulai dari proses identifikasi kebutuhan operasional di masing-masing divisi, perancangan alur proses, penyusunan struktur data, pengembangan script, hingga

tahap pengujian dan implementasi. Seluruh tahapan dilakukan dengan melibatkan komunikasi aktif dengan *Head of IT Department* serta perwakilan *user* yang menjadi pemilik proses (*process owner*) dari masing-masing sistem. Melalui pendekatan ini, setiap sistem yang dibangun dapat memenuhi kebutuhan operasional secara tepat dan sesuai alur kerja nyata perusahaan.

Untuk memberikan gambaran umum mengenai ruang lingkup tugas dan periode pengerjaannya, daftar sistem yang dikembangkan disajikan dalam Tabel 3.1 berikut. Tanggal pelaksanaan akan diisi setelah data final diperoleh dari catatan proyek perusahaan.

Tabel 3.1 Tugas dan Uraian Kerja Magang

No.	Tugas	Realisasi	
		Tanggal Mulai	Tanggal Selesai
1	Sistem <i>Booking</i> Kendaraan	18 July 2025	7 Agustus 2025
2	Sistem <i>Ticketing IT Support</i>	8 Agustus 2025	19 September 2025
3	Sistem <i>Approval Project</i> PM	11 September 2025	28 Oktober 2025
4	<i>Update & Maintenance</i>	28 Oktober 2025	10 November 2025

3.2.1 Sistem Booking Kendaraan

Sistem *booking kendaraan* dikembangkan berdasarkan kebutuhan yang diajukan oleh tim *General Affairs* (GA), yang bertanggung jawab mengelola kendaraan operasional perusahaan dan koordinasi dengan pengemudi (*driver*). Sebelum adanya sistem ini, proses peminjaman kendaraan dilakukan melalui chat manual, sehingga sulit untuk melacak status kendaraan, ketersediaan *driver*, serta tujuan penggunaan kendaraan secara terdokumentasi.

Sistem ini memudahkan seluruh karyawan untuk mengajukan permintaan peminjaman kendaraan melalui Google Form dengan format yang seragam. Data yang masuk kemudian diproses oleh Google Apps Script untuk mengirimkan

notifikasi otomatis kepada GA dan menyimpan *record* ke dalam Google Sheets. Dengan sistem ini, seluruh riwayat peminjaman kendaraan menjadi lebih jelas, dapat diaudit, dan mempermudah GA dalam mengatur jadwal *driver* serta ketersediaan unit kendaraan.

Sistem ini mencakup beberapa proses utama, seperti validasi input, penjadwalan otomatis, pengiriman notifikasi email, serta pembaruan status kendaraan berdasarkan aktivitas peminjaman. Selain itu, sistem ini mendukung proses peninjauan permintaan sehingga GA dapat mengetahui apakah suatu kendaraan atau *driver* sedang dalam penggunaan atau tersedia.



```
// === KONFIGURASI ===  
const SPREADSHEET_ID = '1LJUNPawmUk_5a5YtMkAta3a2aWmYvWp1_5tX_4mU';  
const EMAIL_GA = 'jantzen.fernandes@umsida.ac.id';
```

Gambar 3.3 Screenshot Google Script Sistem Booking Kendaraan 1

Pada Gambar 3.3 dapat dilihat bahwa pengembangan sistem *booking* kendaraan diawali dengan penyusunan konfigurasi dasar yang menjadi fondasi dari keseluruhan sistem. Tahapan ini dilakukan untuk memastikan bahwa sistem memiliki acuan yang jelas terkait sumber penyimpanan data serta pihak yang bertanggung jawab dalam pengelolaan permintaan peminjaman kendaraan.

Pada tahap konfigurasi ini, sistem diarahkan untuk menggunakan Google Sheets sebagai basis data utama dalam mencatat seluruh permintaan *booking* kendaraan yang masuk melalui Google Form. Penetapan Spreadsheet secara spesifik bertujuan agar seluruh data peminjaman dapat tersimpan secara terpusat, terdokumentasi dengan baik, dan mudah diakses oleh pihak terkait. Dengan pendekatan ini, proses pencatatan tidak lagi bergantung pada input manual, sehingga potensi kesalahan pencatatan dapat diminimalkan.

Selain pengaturan media penyimpanan data, konfigurasi awal juga mencakup penentuan alamat email pihak *General Affairs* sebagai penerima notifikasi sistem. Pengaturan ini dirancang agar setiap permintaan *booking* kendaraan yang masuk dapat langsung diketahui oleh pihak yang berwenang tanpa perlu melakukan

pengecekan berkala pada Spreadsheet. Dengan demikian, alur komunikasi antara pemohon dan pengelola kendaraan dapat berjalan lebih cepat dan efisien.

Tahapan konfigurasi awal ini menjadi langkah penting dalam pengembangan sistem karena memastikan seluruh komponen utama, seperti Google Form, Google Sheets, dan layanan email, dapat saling terhubung sebelum sistem memasuki tahap pemrosesan data yang lebih kompleks. Fondasi yang kuat pada tahap ini mendukung keberlangsungan sistem pada tahap-tahap berikutnya.

```
// === UTAMA: Dipanggil saat form disubmit ===
function onFormSubmit(e) {
  const sheet = SpreadsheetApp.openById(SPREADSHEET_ID).getActiveSheet();
  const data = e.values;

  // Ambil data dari form
  const timestamp = data[0];           // Timestamp
  const nama = data[1];                // Nama
  const email = data[2];               // Email Karyawan
  const telepon = data[3];             // Nomor Telepon
  const divisi = data[4];              // Divisi/Departemen
  const tanggal = data[5];            // Tanggal Penggunaan Driver
  const jamMulai = data[6];            // Jam Mulai
  const jamSelesai = data[7];          // Jam Selesai
  const lokasi = data[8];              // Lokasi Tujuan
  const cofipof = data[9];             // COF/IPOF
  const keperluan = data[10];          // Keperluan
  const penumpangInfo = data[11];     // Total & Nama Penumpang
```

Gambar 3.4 Screenshot Google Script Sistem Booking Kendaraan 2

Pada Gambar 3.4 dapat dilihat bahwa sistem mulai memasuki tahap pemrosesan data peminjaman kendaraan yang dikirimkan oleh pengguna melalui Google Form. Pada tahap ini, sistem dirancang untuk merespons setiap pengiriman formulir secara otomatis, sehingga proses pencatatan dapat dilakukan tanpa keterlibatan manual dari pihak pengelola.

Data yang dikirimkan oleh pengguna kemudian dibaca dan dipisahkan ke dalam beberapa variabel sesuai dengan jenis informasi yang dikirimkan. Informasi tersebut meliputi identitas pemohon, kontak yang dapat dihubungi, divisi atau departemen, waktu penggunaan kendaraan, lokasi tujuan, jenis kendaraan, serta keperluan peminjaman. Pemisahan data ini dilakukan agar setiap informasi dapat dikelola secara terstruktur dan digunakan secara tepat pada proses selanjutnya.

Pendekatan pemrosesan data yang terstruktur ini membantu memastikan bahwa seluruh informasi penting tersedia secara lengkap sejak awal. Dengan data yang tersimpan secara rapi, pihak General Affairs dapat memahami konteks permintaan peminjaman kendaraan tanpa perlu melakukan klarifikasi tambahan kepada pemohon. Hal ini mendukung efisiensi kerja serta mengurangi potensi miskomunikasi.

Tahapan ini menunjukkan bagaimana sistem *booking* kendaraan dikembangkan tidak hanya untuk mencatat data, tetapi juga untuk menyiapkan informasi yang siap digunakan dalam proses pengelolaan kendaraan dan penjadwalan *driver*.

```
// Format isi email
const subject = `Permintaan Booking Kendaraan - ${nama}`;
const body =
  `Yth. GA Team,\n\n` +
  `Ada permintaan baru untuk booking kendaraan dengan detail sebagai berikut:\n\n` +
  `👤 Nama: ${nama}\n` +
  `✉ Email: ${email}\n` +
  `☎ Nomor Telepon: ${telepon}\n` +
  `🏢 Divisi/Departemen: ${divisi}\n` +
  `📅 Tanggal Penggunaan: ${tanggal}\n` +
  `🕒 Waktu: ${jamMulai} - ${jamSelesai}\n` +
  `📍 Lokasi Tujuan: ${lokasi}\n` +
  `🆔 COF/IPOF: ${cofipof}\n` +
  `🚗 Keperluan: ${keperluan}\n` +
  `👤 Penumpang: ${penumpangInfo}\n\n` +
  `Mohon ditindaklanjuti dan diassign driver serta kendaraan secara manual.\n\n` +
  `Terima kasih.\n-`;

// Kirim email notifikasi
MailApp.sendEmail({
  to: EMAIL_GA,
  subject: subject,
  body: body
});
```

Gambar 3.5 Screenshot Google Script Sistem Booking Kendaraan 3

Pada Gambar 3.5 dapat dilihat bahwa sistem *booking* kendaraan dikembangkan untuk membentuk dan mengirimkan notifikasi email secara otomatis kepada tim *General Affairs* (GA) setelah pengguna berhasil mengajukan permintaan peminjaman kendaraan melalui sistem. Fitur ini dirancang sebagai bagian penting dari alur kerja sistem untuk memastikan bahwa setiap permintaan

booking dapat segera diketahui oleh pihak yang bertanggung jawab tanpa harus melakukan pengecekan manual pada data Spreadsheet.

Pada tahap ini, sistem menyusun subjek email yang berfungsi sebagai penanda utama isi pesan. Subjek email dibuat sedemikian rupa sehingga mencerminkan adanya permintaan *booking* kendaraan baru, serta dilengkapi dengan identitas nama pemohon. Penyertaan nama pemohon pada subjek email bertujuan untuk memudahkan tim GA dalam mengidentifikasi dan membedakan setiap permintaan yang masuk, terutama ketika terdapat beberapa permintaan peminjaman kendaraan dalam waktu yang berdekatan.

Selanjutnya, sistem menyusun isi email secara terstruktur melalui pembentukan badan email yang memuat informasi lengkap terkait permintaan peminjaman kendaraan. Isi email diawali dengan sapaan kepada tim GA, kemudian dilanjutkan dengan rincian data yang dikirimkan oleh pengguna melalui Google Form. Informasi tersebut mencakup identitas pemohon, alamat email, nomor telepon, divisi atau departemen, tanggal penggunaan kendaraan, waktu mulai dan selesai penggunaan, lokasi tujuan, jenis kendaraan yang dipilih, serta keperluan peminjaman kendaraan.

Penyajian informasi secara lengkap dan terperinci dalam isi email bertujuan agar tim GA dapat langsung memahami konteks permintaan peminjaman tanpa perlu membuka atau menelusuri data pada Spreadsheet. Dengan demikian, proses tindak lanjut seperti pengecekan ketersediaan kendaraan, penyesuaian jadwal, serta penugasan driver dapat dilakukan dengan lebih cepat dan efisien. Pada bagian akhir email juga disertakan pesan yang menegaskan perlunya tindak lanjut dari tim GA, sehingga alur koordinasi tetap berjalan sesuai dengan prosedur yang berlaku.

Setelah seluruh format email berhasil disusun, sistem secara otomatis menjalankan proses pengiriman email ke alamat email tim General Affairs yang telah ditentukan pada tahap konfigurasi awal sistem. Pengiriman email ini dilakukan setiap kali terdapat pengajuan *booking* kendaraan baru, sehingga memastikan bahwa seluruh permintaan dapat ditindaklanjuti secara tepat waktu dan tidak terlewat. Implementasi fitur pengiriman email notifikasi pada Gambar 3.5 menghasilkan sistem yang responsif dan mendukung komunikasi *real-time* antara

pengguna dan pengelola kendaraan. Setiap permintaan *booking* kendaraan dapat langsung diteruskan kepada tim *General Affairs* melalui email, sehingga meningkatkan kecepatan respon, keteraturan proses administrasi, serta transparansi dalam pengelolaan peminjaman kendaraan di lingkungan PT Dayalima Abisatya.

3.2.2 Sistem Ticketing IT Support

Sistem *Ticketing IT Support* dikembangkan berdasarkan permintaan dari tim *IT Support*, yang membutuhkan platform terpusat untuk menangani permintaan bantuan teknis dari seluruh divisi perusahaan. Sebelum sistem ini dibuat, karyawan yang membutuhkan bantuan IT harus datang langsung ke area kerja Tim IT atau menghubungi mereka melalui pesan pribadi, yang sering menimbulkan antrean, miskomunikasi, dan data yang tidak tercatat.

Sistem *Ticketing IT Support* dirancang sebagai solusi untuk mencatat seluruh permintaan bantuan IT secara otomatis melalui Google Form, yang kemudian diproses oleh Google Apps Script untuk menghasilkan tiket digital dengan nomor unik. Setiap tiket yang masuk disimpan ke Google Sheets dan dikategorikan berdasarkan jenis kendala, seperti jaringan, akses email, perangkat komputer, dan aplikasi internal. Hal ini mempermudah Tim IT Support dalam memprioritaskan laporan, memantau status pengerjaan, serta menutup ticket setelah masalah terselesaikan.

Selain itu, sistem ini dilengkapi dengan notifikasi otomatis yang mengirimkan email ke pelapor dan tim IT setiap kali tiket dibuat, diperbarui, atau ditutup. Dengan adanya sistem ini, alur komunikasi menjadi lebih jelas, terstruktur, dan seluruh histori penanganan masalah dapat digunakan sebagai acuan evaluasi kinerja *IT Support*.

```
*****
* KONFIGURASI
*****/
const SPREADSHEET_ID = '1Rn8S2E6wW3Dre0U7Ad293Qs0mW6a-0zRTTnv38g0A'; // ganti dengan ID spreadsheet tiket
const EMAIL_IT      = 'it_helpdesk@dayalima.id'; // email tujuan IT Helpdesk
```

Gambar 3.6 Screenshot Code Sistem Ticketing IT Support 1

Pada Gambar 3.6 dapat dilihat bahwa *screenshot* kode yang ditampilkan merupakan bagian konfigurasi awal pada Sistem *Ticketing IT Support* yang dikembangkan menggunakan Google Apps Script. Konfigurasi awal ini disusun sebagai langkah pertama dalam pengembangan sistem untuk menetapkan komponen-komponen dasar yang diperlukan sebelum seluruh proses pengelolaan tiket bantuan dijalankan. Melalui tahap ini, sistem dipersiapkan agar dapat berjalan secara terintegrasi dan selaras dengan kebutuhan operasional perusahaan.

Variabel `SPREADSHEET_ID` digunakan untuk menentukan identitas Google Sheets yang berfungsi sebagai basis data utama dalam sistem *ticketing*. Spreadsheet ini digunakan sebagai media penyimpanan seluruh data tiket bantuan yang dikirimkan oleh karyawan melalui Google Form. Dengan penentuan Spreadsheet secara spesifik, setiap tiket yang masuk dapat tercatat secara otomatis, tersusun dengan rapi, dan terdokumentasi dengan baik tanpa memerlukan pencatatan manual oleh tim *IT Support*.

Selain pengaturan media penyimpanan data, variabel `EMAIL_IT` digunakan untuk menyimpan alamat email tujuan tim *IT Helpdesk*. Alamat email ini berperan sebagai penerima notifikasi otomatis setiap kali terdapat tiket bantuan baru yang masuk ke dalam sistem. Mekanisme ini dirancang untuk memastikan bahwa tim *IT Support* dapat segera mengetahui adanya permintaan bantuan yang masuk tanpa perlu melakukan pengecekan berkala terhadap Spreadsheet.

Konfigurasi awal sistem ini juga dirancang untuk menghubungkan beberapa layanan dalam ekosistem Google Workspace, yaitu Google Form sebagai media input tiket, Google Sheets sebagai media penyimpanan data, dan layanan email sebagai sarana notifikasi. Integrasi antar layanan tersebut memungkinkan alur penanganan tiket berjalan secara otomatis dan terpusat, sehingga dapat meminimalkan risiko miskomunikasi serta keterlambatan dalam merespons permintaan bantuan.

Struktur konfigurasi yang sederhana namun terpusat memudahkan proses pemeliharaan dan pengembangan sistem pada tahap selanjutnya. Perubahan

terhadap Spreadsheet tujuan atau alamat email penerima notifikasi dapat dilakukan dengan cepat tanpa perlu mengubah keseluruhan logika program. Dengan pendekatan ini, sistem *ticketing* menjadi lebih fleksibel dan mudah disesuaikan dengan kebutuhan operasional perusahaan yang dapat berkembang dari waktu ke waktu.

```

✓ function _nv(nv, key) {
  |   return (nv[key] && nv[key][0]) ? String(nv[key][0]).trim() : '';
  | }

✓ /**
  * Cari atau buat header kolom di sheet.
  * @param {GoogleAppsScript.Spreadsheet.Sheet} sheet
  * @param {string} headerName
  * @return {number} index kolom (1-based)
  */
✓ function _getOrCreateHeaderCol(sheet, headerName) {
  |   const lastCol = sheet.getLastColumn();
  |   const headers = sheet.getRange(1, 1, 1, Math.max(lastCol, 1)).getValues()[0];
  |   let idx = headers.indexOf(headerName);
  |   if (idx === -1) {
  |     |   idx = headers.filter(h => h !== '').length;
  |     |   sheet.getRange(1, idx + 1).setValue(headerName);
  |   }
  |   return idx + 1;
  | }

```

Gambar 3.7 Screenshot Code Sistem Ticketing IT Support 2

Pada Gambar 3.7 dapat dilihat bahwa pada tahap ini sistem *Ticketing IT Support* dilengkapi dengan sejumlah fungsi pendukung yang dirancang untuk membantu proses pengolahan data tiket serta pengelolaan struktur kolom pada Google Sheets. Fungsi-fungsi ini dikembangkan sebagai bagian dari upaya untuk menjaga kestabilan sistem dan memastikan bahwa data tiket yang diproses memiliki format yang konsisten serta mudah dikelola.

Salah satu fungsi pendukung yang ditampilkan adalah fungsi `_nv(nv, key)`, yang digunakan untuk mengambil nilai tertentu dari data input yang diterima sistem melalui Google Form. Fungsi ini berperan dalam melakukan validasi awal terhadap data dengan cara memeriksa ketersediaan dan isi nilai berdasarkan kunci yang ditentukan. Apabila data tersedia, nilai tersebut dikonversi ke dalam bentuk string dan dibersihkan dari spasi berlebih. Sebaliknya, apabila data tidak tersedia, fungsi ini akan mengembalikan nilai kosong. Pendekatan ini diterapkan untuk mengurangi

risiko kesalahan pemrosesan data akibat input yang tidak lengkap atau tidak terdefinisi.

Selain itu, pada Gambar 3.7 juga ditampilkan fungsi `getOrCreateHeaderCol(sheet, headerName)` yang dikembangkan untuk mengelola struktur kolom pada Google Sheets secara dinamis. Fungsi ini berfungsi untuk memastikan bahwa setiap data tiket disimpan pada kolom yang sesuai, meskipun struktur Spreadsheet mengalami perubahan atau penambahan atribut data.

Proses kerja fungsi ini dimulai dengan membaca seluruh *header* pada baris pertama Spreadsheet dan membandingkannya dengan nama *header* yang dibutuhkan oleh sistem. Apabila *header* yang dicari sudah tersedia, fungsi akan mengembalikan posisi kolom tersebut. Namun, jika *header* belum ditemukan, sistem secara otomatis akan menambahkan kolom baru pada bagian akhir Spreadsheet dengan nama *header* yang sesuai. Mekanisme ini memungkinkan sistem *ticketing* untuk beradaptasi terhadap kebutuhan data tanpa memerlukan pengaturan manual.

Penggunaan fungsi pengelolaan *header* secara dinamis memiliki peran penting dalam pengembangan sistem *Ticketing IT Support*. Dengan pendekatan ini, sistem dapat dengan mudah menambahkan atribut atau informasi tiket baru apabila dibutuhkan di kemudian hari, tanpa harus melakukan perubahan struktur Spreadsheet secara manual setiap kali terjadi penyesuaian.

Keberadaan fungsi-fungsi pendukung seperti yang ditampilkan pada Gambar 3.7 membantu memastikan bahwa proses pencatatan tiket berjalan secara konsisten dan minim kesalahan. Data yang diterima dari Google Form dapat diproses dan disimpan dalam format yang rapi, terstruktur, serta siap digunakan pada tahapan berikutnya, seperti pengelompokan tiket, pembaruan status, dan pengiriman notifikasi kepada pihak terkait.

```

function onFormSubmit(e) {
  const sheet = e.range.getSheet();
  const rowIdx = e.range.getRow();
  const values = e.values;
  const nv = e.namedValues || {};

  // ===== 1. BACA DATA DARI FORM =====
  const timestampStr = values[0] || '';
  const nama = _nv(nv, 'Nama Karyawan'); // col 1
  const email = _nv(nv, 'Email Karyawan'); // col 2
  const telepon = _nv(nv, 'Nomor Telepon '); // col 3
  const departemen = _nv(nv, 'Departemen'); // col 4
  const kendala = _nv(nv, 'Kendala'); // col 5

  // field yang berbeda tergantung tipe kendala
  let jenis = '', detail = '', kapan = '';
  if (kendala === 'Problem') {
    jenis = _nv(nv, 'Jenis Kendala'); // col 6
    detail = _nv(nv, 'Detail Kendala'); // col 7
    kapan = _nv(nv, 'Kapan Kendala Terjadi?'); // col 8
  } else if (kendala === 'Request') {
    jenis = _nv(nv, 'Jenis Request'); // col 9
    detail = _nv(nv, 'Detail Request'); // col 10
    kapan = _nv(nv, 'Kapan Request Dibutuhkan? '); // col 11
  }
}

```

Gambar 3.8 Screenshot Code Sistem Ticketing IT Support 3

Pada Gambar 3.8 dapat dilihat bahwa potongan kode yang ditampilkan merupakan bagian inti dari sistem *Ticketing IT Support*, khususnya pada tahap penerimaan dan pengolahan awal data tiket yang dikirimkan oleh pengguna melalui Google Form. Pada tahap ini, sistem dirancang untuk merespons setiap pengiriman formulir secara otomatis melalui fungsi `onFormSubmit(e)`, sehingga seluruh proses pencatatan tiket dapat dilakukan tanpa memerlukan intervensi manual dari tim *IT Support*.

Pada bagian awal fungsi, sistem mengambil referensi terhadap Spreadsheet yang digunakan sebagai basis data serta baris data terbaru yang ditambahkan melalui objek `e.range`. Data hasil pengisian formulir kemudian disimpan ke dalam beberapa variabel, seperti `values` dan `namedValues`. Penggunaan kedua pendekatan ini memungkinkan sistem untuk membaca jawaban pengguna baik berdasarkan

urutan pengisian maupun berdasarkan nama pertanyaan pada Google Form, sehingga proses pengambilan data menjadi lebih fleksibel dan terstruktur.

Selanjutnya, sistem mulai melakukan pembacaan data utama yang dibutuhkan dalam proses *ticketing*, seperti waktu pengisian formulir (timestamp), nama karyawan, alamat email, nomor telepon, departemen, serta jenis kendala yang dilaporkan. Data-data tersebut diambil menggunakan fungsi pendukung `_nv()` yang telah dijelaskan pada tahap sebelumnya. Penggunaan fungsi ini membantu memastikan bahwa setiap nilai yang diproses telah melalui validasi dasar dan tidak mengandung nilai kosong yang berpotensi menyebabkan kesalahan dalam pencatatan data.

Pada tahap berikutnya, sistem menerapkan logika percabangan menggunakan kondisi `if` dan `else if` berdasarkan jenis kendala yang dipilih oleh pengguna. Apabila pengguna memilih jenis kendala berupa *Problem*, sistem akan mengambil informasi tambahan yang relevan, seperti kategori kendala, detail permasalahan, serta waktu terjadinya kendala. Sebaliknya, *apabila* pengguna memilih jenis kendala berupa *Request*, sistem akan mengambil data yang berbeda, seperti jenis permintaan, deskripsi kebutuhan, serta waktu kebutuhan permintaan tersebut.

Penerapan logika percabangan ini dilakukan untuk menyesuaikan struktur data tiket dengan karakteristik laporan yang masuk. Dengan pendekatan ini, sistem tidak memaksakan satu format data yang sama untuk seluruh jenis tiket, melainkan mencatat informasi yang benar-benar relevan sesuai dengan konteks permasalahan atau permintaan yang diajukan oleh pengguna.

Pendekatan klasifikasi data seperti yang ditampilkan pada Gambar 3.8 membantu tim *IT Support* dalam memahami konteks setiap tiket sejak tahap awal pencatatan. Data yang tersimpan pada Spreadsheet menjadi lebih informatif, terstruktur, dan mudah dibedakan berdasarkan jenis kendala. Hal ini memudahkan proses analisis awal, penentuan prioritas penanganan, serta pelaksanaan tindak lanjut pada tahap penanganan tiket berikutnya.

```
// ===== 2. GENERATE EVENT ID UNIK =====
let dt = new Date(timestampStr);
if (isNaN(dt)) dt = new Date();
const yyyy = dt.getFullYear();
const mm    = String(dt.getMonth() + 1).padStart(2, '0');
const dd    = String(dt.getDate()).padStart(2, '0');
```

Gambar 3.9 Screenshot Code Sistem Ticketing IT Support 4

Pada Gambar 3.9 dapat dilihat bahwa pada tahap ini sistem *Ticketing IT Support* dikembangkan untuk membentuk dasar identitas waktu yang akan digunakan dalam proses pembuatan tiket. Tahapan ini menjadi bagian penting dalam pengembangan sistem karena setiap permintaan bantuan IT yang masuk perlu memiliki penanda waktu yang jelas dan konsisten agar dapat dikelola dengan baik oleh tim *IT Support*.

Pada implementasi ini, sistem memanfaatkan data waktu (*timestamp*) yang diperoleh dari pengiriman Google Form untuk kemudian diproses menjadi format tanggal yang dapat digunakan oleh Google Apps Script. Proses ini dilakukan agar data waktu yang diterima sistem dapat diolah secara lebih fleksibel dan terhindar dari permasalahan format data yang tidak sesuai.

Selain itu, sistem juga dirancang untuk melakukan validasi terhadap data waktu yang diterima. Apabila data waktu tersebut tidak dapat diproses dengan baik, sistem secara otomatis menggunakan waktu saat ini sebagai alternatif. Mekanisme ini diterapkan untuk menjaga stabilitas sistem dan memastikan bahwa setiap tiket tetap memiliki informasi waktu yang valid meskipun terjadi ketidaksesuaian pada data input.

Setelah data waktu berhasil diproses, sistem memisahkan informasi tanggal ke dalam beberapa komponen, yaitu tahun, bulan, dan hari. Setiap komponen kemudian diformat agar memiliki struktur yang seragam, khususnya pada bagian bulan dan tanggal yang ditampilkan dalam dua digit. Pemformatan ini dilakukan untuk menjaga konsistensi data serta memudahkan proses pengelompokan dan pencarian tiket di kemudian hari.

Pengembangan fitur pembentukan dasar identitas waktu ini mendukung tujuan utama sistem *ticketing*, yaitu menciptakan mekanisme pencatatan tiket yang terstruktur dan mudah ditelusuri. Dengan adanya informasi waktu yang konsisten, tim *IT Support* dapat memantau alur masuknya tiket, mengurutkan prioritas penanganan, serta melakukan evaluasi berdasarkan periode tertentu secara lebih sistematis.

```
// hitung tiket sebelumnya di hari yang sama
const prevCountLen = Math.max(0, rowIdx - 2);
let sameDayPrev = 0;
if (prevCountLen > 0) {
  const prevTimestamps = sheet.getRange(2, 1, prevCountLen, 1).getValues();
  sameDayPrev = prevTimestamps.filter(r => {
    const t = new Date(r[0]);
    return !isNaN(t) &&
      t.getFullYear() === yyyy &&
      (t.getMonth()+1) === (dt.getMonth()+1) &&
      t.getDate() === dt.getDate();
  }).length;
}
const counter = _pad3(sameDayPrev + 1);
const eventId = `TCK-${yyyy}${mm}${dd}-${counter}`;
```

Gambar 3.10 Screenshot Code Sistem Ticketing IT Support 5

Pada Gambar 3.10 dapat dilihat bahwa potongan kode yang ditampilkan merupakan bagian penting dalam sistem *Ticketing IT Support* yang berfungsi untuk membentuk identitas tiket (ticket ID) secara unik dan terstruktur. Mekanisme ini dikembangkan untuk memastikan bahwa setiap permintaan bantuan IT yang masuk memiliki kode identifikasi yang berbeda, mudah dilacak, serta dapat digunakan sebagai acuan dalam proses penanganan dan evaluasi tiket.

Pada tahap awal, sistem menghitung jumlah tiket yang telah tercatat sebelumnya pada tanggal yang sama. Proses ini dilakukan dengan membaca data *timestamp* dari tiket-tiket yang telah tersimpan di Google Sheets, kemudian membandingkannya dengan tanggal pengajuan tiket yang sedang diproses. Dengan pendekatan ini, sistem dapat mengetahui berapa banyak tiket yang sudah masuk pada hari tersebut sebelum tiket terbaru ditambahkan.

Selanjutnya, sistem melakukan proses penyaringan data untuk memastikan bahwa perhitungan hanya dilakukan terhadap tiket yang benar-benar memiliki

tanggal yang sama, baik dari sisi tahun, bulan, maupun hari. Proses validasi ini penting untuk mencegah kesalahan perhitungan akibat data yang tidak sesuai format atau data waktu yang tidak valid.

Hasil dari proses perhitungan tersebut kemudian digunakan untuk menentukan nomor urut tiket pada hari yang bersangkutan. Nomor urut ini akan bertambah secara otomatis setiap kali terdapat tiket baru yang masuk pada tanggal yang sama. Dengan demikian, sistem mampu menjaga konsistensi penomoran tiket tanpa memerlukan pencatatan manual oleh tim *IT Support*.

Setelah nomor urut diperoleh, sistem membentuk ID tiket dengan format yang telah ditentukan, yaitu dengan menggabungkan kode *prefix* tiket, informasi tanggal (tahun, bulan, dan hari), serta nomor urut tiket pada hari tersebut. Format ini dirancang agar ID tiket tidak hanya bersifat unik, tetapi juga informatif, karena secara langsung menunjukkan waktu pengajuan tiket.

Implementasi pembentukan ID tiket seperti yang ditampilkan pada Gambar 3.10 memberikan kemudahan dalam proses pengelolaan tiket *IT Support*. Tim IT dapat dengan cepat mengidentifikasi tiket berdasarkan tanggal masuk, memantau urutan permintaan bantuan, serta melakukan pencarian atau pelacakan tiket secara lebih efisien. Selain itu, struktur ID tiket yang konsisten juga mendukung proses dokumentasi dan pelaporan kinerja *IT Support* dalam jangka panjang.

```
// simpan Event ID di kolom "Event ID"  
const eventCol = _getOrCreateHeaderCol(sheet, 'Event ID');  
sheet.getRange(rowIdx, eventCol).setValue(eventId);
```

Gambar 3.11 Screenshot Code Sistem Ticketing IT Support 6

Pada Gambar 3.11 dapat dilihat bahwa potongan kode yang ditampilkan berfungsi untuk menyimpan identitas tiket (Event ID) yang telah dibentuk sebelumnya ke dalam Google Sheets sebagai basis data utama Sistem *Ticketing IT Support*. Tahapan ini merupakan bagian lanjutan yang memastikan setiap tiket yang masuk tidak hanya memiliki ID unik, tetapi juga tercatat secara permanen dan terstruktur pada media penyimpanan data.

Pada tahap ini, sistem terlebih dahulu menentukan posisi kolom tempat ID tiket akan disimpan dengan memanfaatkan fungsi `getOrCreateHeaderCol`. Fungsi tersebut digunakan untuk mencari keberadaan kolom dengan nama Event ID pada Spreadsheet. Apabila kolom tersebut belum tersedia, sistem secara otomatis akan membuat kolom baru dengan header yang sesuai. Pendekatan ini memungkinkan sistem tetap berjalan dengan baik meskipun struktur Spreadsheet mengalami perubahan atau pengembangan di kemudian hari.

Setelah posisi kolom Event ID diperoleh, sistem kemudian menuliskan nilai ID tiket ke baris yang sesuai dengan data tiket yang baru saja masuk. Penulisan data dilakukan secara otomatis berdasarkan indeks baris yang diperoleh dari proses sebelumnya, sehingga tidak terjadi pergeseran atau kesalahan penempatan data.

Penyimpanan ID tiket ke dalam Spreadsheet memiliki peran penting dalam proses pengelolaan tiket IT Support. Dengan adanya ID yang tercatat secara konsisten, setiap tiket dapat dengan mudah diidentifikasi, ditelusuri, dan dirujuk kembali pada proses penanganan lanjutan, pembaruan status, maupun evaluasi kinerja tim *IT Support*.

Implementasi mekanisme penyimpanan ID tiket seperti yang ditunjukkan pada Gambar 3.11 membantu menciptakan sistem *ticketing* yang lebih tertata dan profesional. Setiap tiket yang masuk memiliki identitas yang jelas dan terdokumentasi dengan baik, sehingga mendukung proses administrasi *IT Support* yang lebih sistematis dan akuntabel di lingkungan PT Dayalima Abisatya.

```
// ===== 3. SIAPKAN EMAIL KE IT =====
let bodyIT =
  `Yth. IT Helpdesk,\n\n` +
  `Terdapat tiket baru yang diajukan melalui sistem Ticketing dengan detail berikut:\n\n` +
  `📄 Event ID: ${eventId}\n` +
  `👤 Nama: ${nama}\n` +
  `✉ Email: ${email}\n` +
  `☎ Nomor Telepon: ${telepon}\n` +
  `🏢 Departemen: ${departemen}\n` +
  `🔍 Tipe Kendala: ${kendala}\n`;
```

Gambar 3.12 Screenshot Code Sistem Ticketing IT Support 7

Pada Gambar 3.12 dapat dilihat bahwa potongan kode yang ditampilkan merupakan bagian dari pengembangan sistem *Ticketing IT Support* yang bertugas menyusun isi email notifikasi untuk tim *IT Helpdesk* ketika terdapat tiket baru yang masuk. Pengembangan bagian ini dilakukan untuk memastikan bahwa setiap permintaan bantuan IT yang diajukan oleh karyawan dapat langsung diketahui oleh tim yang bertanggung jawab tanpa harus melakukan pengecekan manual pada sistem.

Pada tahap ini, sistem mulai membentuk badan email yang berisi informasi utama terkait tiket yang diajukan. Informasi yang dimuat mencakup identitas tiket berupa Event ID, nama pelapor, alamat email, nomor telepon, departemen, serta tipe kendala yang dilaporkan. Penyusunan informasi ini dilakukan secara terstruktur agar tim *IT Helpdesk* dapat dengan cepat memahami konteks tiket yang masuk sejak awal.

Pengembangan fitur ini bertujuan untuk mendukung alur kerja tim *IT Support* agar lebih efisien. Dengan adanya notifikasi email yang memuat informasi dasar tiket, tim IT dapat segera melakukan analisis awal dan menentukan langkah penanganan yang diperlukan tanpa harus membuka data tiket secara manual pada Spreadsheet.

```
if (kendala === 'Problem') {
  bodyIT +=
    '\n- Rincian Problem -\n' +
    '🔍 Jenis Kendala: ${jenis}\n' +
    '📄 Detail Kendala: ${detail}\n' +
    '🕒 Waktu Kejadian: ${kapan}\n';
} else if (kendala === 'Request') {
  bodyIT +=
    '\n- Rincian Request -\n' +
    '🔍 Jenis Request: ${jenis}\n' +
    '📄 Detail Request: ${detail}\n' +
    '🕒 Waktu Dibutuhkan: ${kapan}\n';
}

bodyIT += '\nMohon ditindaklanjuti sesuai prosedur.\n\nTerima kasih.\n- IT Ticketing System';

MailApp.sendEmail({
  to: EMAIL_IT,
  subject: 'Ticket Baru [${eventId}] - ${nama}',
  body: bodyIT
});
```

Gambar 3.13 Screenshot Code Sistem Ticketing IT Support 8

Pada Gambar 3.13 dapat dilihat bahwa sistem *ticketing* dikembangkan dengan menerapkan logika percabangan untuk menyesuaikan isi email notifikasi berdasarkan jenis kendala yang dipilih oleh pengguna. Pada tahap ini, sistem membedakan antara tiket dengan tipe *Problem* dan tiket dengan tipe *Request*, sehingga informasi yang disampaikan kepada tim *IT Helpdesk* menjadi lebih relevan.

Apabila pengguna melaporkan kendala dengan tipe *Problem*, sistem akan menambahkan rincian tambahan ke dalam isi email berupa jenis kendala, detail permasalahan yang dialami, serta waktu terjadinya kendala tersebut. Informasi ini membantu tim *IT Support* dalam memahami karakteristik masalah secara lebih mendalam sebelum melakukan penanganan.

Sebaliknya, apabila jenis kendala yang dipilih adalah *Request*, sistem akan menyusun isi email dengan rincian yang berbeda, seperti jenis permintaan, detail kebutuhan pengguna, serta waktu kebutuhan *request*. Pendekatan ini dikembangkan agar sistem tidak mencatat seluruh tiket dengan format yang sama, melainkan menyesuaikan informasi dengan konteks permintaan yang masuk.

Penerapan logika ini menunjukkan bahwa sistem *ticketing* dirancang untuk fleksibel dan adaptif terhadap berbagai jenis permintaan bantuan IT yang terjadi di lingkungan perusahaan.

```
// ===== 4. SIAPKAN EMAIL BALASAN KE USER =====
let bodyUser =
  `Yth. ${nama},\n\n` +
  `Terima kasih telah menghubungi IT Helpdesk Dayalima. Tiket Anda telah tercatat dengan rincian berikut:\n\n` +
  `📄 Event ID: ${eventId}\n` +
  `📄 Tipe Kendala: ${kendala}\n`;

if (kendala === 'Problem') {
  bodyUser +=
    `\n- Rincian Problem -\n` +
    `📄 Jenis Kendala: ${jenis}\n` +
    `📄 Detail Kendala: ${detail}\n` +
    `📄 Waktu Kejadian: ${kapan}\n`;
} else if (kendala === 'Request') {
  bodyUser +=
    `\n- Rincian Request -\n` +
    `📄 Jenis Request: ${jenis}\n` +
    `📄 Detail Request: ${detail}\n` +
    `📄 Waktu Dibutuhkan: ${kapan}\n`;
}
```

Gambar 3.14 Screenshot Code Sistem Ticketing IT Support 9

Gambar 3.14 Screenshot Code Sistem Ticketing IT Support 9

Pada Gambar 3.14 dapat dilihat bahwa potongan kode yang ditampilkan merupakan bagian lanjutan dari sistem *Ticketing IT Support* yang bertugas menjalankan proses pengiriman email notifikasi secara otomatis kepada tim *IT Helpdesk*. Tahapan ini dikembangkan sebagai bentuk implementasi dari alur komunikasi sistem, di mana setiap tiket yang berhasil dicatat perlu segera diinformasikan kepada tim yang bertanggung jawab dalam penanganan kendala atau permintaan bantuan IT.

Setelah seluruh isi email untuk tim *IT Helpdesk* disusun pada tahap sebelumnya, sistem memanfaatkan layanan email yang tersedia pada Google Apps Script untuk mengirimkan notifikasi ke alamat email yang telah ditentukan dalam konfigurasi sistem. Pengiriman email ini dilakukan secara otomatis setiap kali fungsi pemrosesan tiket dijalankan, sehingga tidak diperlukan tindakan manual dari pihak manapun untuk menyampaikan informasi tiket kepada tim *IT Support*.

Email notifikasi yang dikirimkan memuat informasi penting terkait tiket yang masuk, seperti identitas tiket (Event ID), identitas pelapor, departemen, serta jenis kendala yang dilaporkan. Penyampaian informasi ini dirancang agar tim *IT Helpdesk* dapat langsung memahami konteks tiket sejak awal tanpa perlu membuka Spreadsheet atau sistem pencatatan lainnya. Dengan demikian, proses analisis awal terhadap tiket dapat dilakukan dengan lebih cepat dan efisien.

Selain berfungsi sebagai sarana pemberitahuan, pengiriman email notifikasi ini juga berperan sebagai dokumentasi awal proses penanganan tiket. Setiap email yang dikirimkan dapat menjadi bukti bahwa tiket telah diteruskan kepada tim *IT Support* dan siap untuk ditindaklanjuti sesuai dengan prosedur yang berlaku. Hal ini mendukung keteraturan administrasi serta akuntabilitas dalam pengelolaan tiket bantuan IT.

Implementasi pengiriman email otomatis seperti yang ditunjukkan pada Gambar 3.14 memberikan dampak positif terhadap responsivitas tim *IT Helpdesk*. Tim IT tidak perlu menunggu laporan secara lisan atau pesan pribadi dari pengguna,

karena seluruh permintaan bantuan telah tercatat dan dikomunikasikan melalui sistem yang terstruktur. Pendekatan ini membantu mengurangi potensi keterlambatan respon dan memastikan bahwa setiap tiket memperoleh perhatian yang sesuai.

Dengan adanya mekanisme pengiriman email notifikasi ini, sistem *Ticketing IT Support* mampu menjembatani proses pencatatan tiket dengan proses penanganan kendala secara lebih efektif. Alur kerja menjadi lebih jelas, terdokumentasi, dan mudah ditelusuri, sehingga mendukung peningkatan kualitas layanan *IT Support* di lingkungan PT Dayalima Abisatya.

```
bodyUser +=
'\nTiket Anda akan ditinjau dan diproses oleh Tim IT maksimal dalam waktu 3x24 jam.\n' +
'Apabila ada perkembangan lebih lanjut, kami akan menghubungi Anda melalui email atau nomor telepon yang tercantum.\n\n' +
'Hormat kami,\nTim IT Helpdesk Dayalima';

MailApp.sendEmail({
  to: email,
  subject: 'Konfirmasi Tiket Anda [${eventId}]',
  body: bodyUser
});
}
```

Gambar 3.15 Screenshot Code Sistem Ticketing IT Support 10

Pada Gambar 3.15 dapat dilihat bahwa sistem *ticketing* juga dikembangkan untuk mengirimkan email konfirmasi secara otomatis kepada pengguna setelah tiket berhasil tercatat. Pengembangan bagian ini bertujuan untuk memberikan kepastian kepada pengguna bahwa laporan yang diajukan telah diterima dan akan diproses oleh tim *IT Helpdesk*. Email konfirmasi diawali dengan sapaan kepada pengguna, kemudian diikuti dengan informasi bahwa tiket telah berhasil dicatat dalam sistem. Informasi utama yang disampaikan meliputi Event ID tiket dan tipe kendala yang dilaporkan, sehingga pengguna memiliki referensi resmi atas laporan yang telah diajukan.

Selanjutnya, sistem kembali menampilkan rincian kendala atau permintaan sesuai dengan jenis tiket yang dipilih, baik *Problem* maupun *Request*. Pada bagian akhir email, sistem menyampaikan estimasi waktu peninjauan tiket serta informasi bahwa pengguna akan dihubungi kembali apabila terdapat perkembangan lebih lanjut.

Pengiriman email konfirmasi ini mendukung transparansi dan profesionalisme sistem *Ticketing IT Support*, karena pengguna mendapatkan umpan balik langsung setelah mengajukan tiket dan memahami alur penanganan yang akan dilakukan oleh Tim IT Support.

3.2.3 Sistem Approval Project PM

Sistem *Approval Project PM* dikembangkan atas permintaan gabungan dari Tim HR dan *Project Management Office* (PMO). Sistem ini dibuat untuk mempercepat proses persetujuan proyek (*approval*) oleh *Project Manager*, terutama untuk unit *Diagnostic* dan *Learning*. Sebelum adanya sistem ini, proses *approval* dilakukan secara manual dengan menyerahkan dokumen fisik atau file terpisah, yang menyebabkan keterlambatan proses, risiko dokumen hilang, serta inkonsistensi data.

Sistem *Approval Project PM* ini memungkinkan team leader untuk mengajukan laporan proyek melalui form digital. Data tersebut diproses menggunakan Google Apps Script, kemudian dikirimkan kepada *Project Manager* dalam bentuk email notifikasi yang berisi informasi proyek beserta tombol persetujuan. Setelah PM melakukan approval, status proyek akan diperbarui secara otomatis pada *database* Google Sheets.

Sistem ini memberikan keuntungan berupa kemudahan *monitoring* proyek, percepatan proses persetujuan, pengurangan penggunaan dokumen fisik, serta pencatatan yang rapi dan terstandarisasi. Sistem ini juga membantu PMO dalam meninjau progres proyek secara berkala karena seluruh data terkumpul dalam sistem terpusat.

Gambar 3.16 Screenshot Code Sistem Approval Projek PM 1

Pada tahap konfigurasi ini, variabel SPREADSHEET_ID digunakan untuk menentukan Google Sheets yang berfungsi sebagai media penyimpanan utama seluruh data pengajuan dan approval proyek PM. Spreadsheet ini digunakan untuk mencatat informasi proyek yang telah dikerjakan oleh tim, sehingga proses pendataan dan peninjauan dapat dilakukan secara terpusat dan terdokumentasi dengan baik.

52

Selain itu, variabel `DEFAULT_PL_EMAIL` disiapkan sebagai alamat email cadangan apabila data divisi atau email *Practice Leader* tidak ditemukan. Pendekatan ini diterapkan untuk menjaga keandalan sistem agar proses *approval* tetap dapat berjalan meskipun terjadi ketidaksesuaian data pada tahap input.

Konfigurasi sistem ini juga mencakup penetapan alamat `WEB_APP_URL`, yang digunakan sebagai penghubung ke Web App Google Apps Script yang telah dideploy. Web App ini berperan sebagai media interaksi dalam proses *approval*, di mana pihak *Practice Leader* dapat melakukan persetujuan proyek secara digital tanpa perlu menandatangani dokumen fisik atau berpindah-pindah *file* secara manual.

Fungsi pendukung `_nv()` yang ditampilkan pada bagian akhir konfigurasi kembali digunakan untuk membantu proses pengambilan dan validasi data input. Penggunaan fungsi ini menunjukkan konsistensi pendekatan pengembangan sistem, di mana setiap data yang diterima dari pengguna dipastikan dalam kondisi aman untuk diproses lebih lanjut.

Secara keseluruhan, konfigurasi awal pada Sistem *Approval Project* PM menunjukkan kesinambungan pola pengembangan dengan proyek-proyek sebelumnya, namun dengan penyesuaian khusus untuk mendukung proses *approval* proyek pada dua divisi yang berbeda. Pendekatan ini memungkinkan sistem untuk tetap sederhana, terstruktur, dan mudah dipelihara, sekaligus mampu memenuhi kebutuhan operasional PM *Diagnostic* dan PM *Learning* dalam proses pendataan serta persetujuan proyek.

```
function onFormSubmit(e) {
  const nv = e.namedValues || {};
  const ss = SpreadsheetApp.openById(SPREADSHEET_ID);
  const sheet = ss.getSheets()[0];
  const lastRow = sheet.getLastRow();

  const projectManager = _nv(nv, 'Project Manager');
  const email = _nv(nv, 'Email');
  const cofNumber = _nv(nv, 'COF/IPOF NUMBER');
  const practiceLeader = _nv(nv, 'Practice Leader');
  const clientName = _nv(nv, 'Clients Name');
  const periodeProject = _nv(nv, 'Periode Project'); // <-- perubahan di sini
  const resName1 = _nv(nv, 'Resource Name 1');
  const resRole1 = _nv(nv, 'Resource Role 1');
  const resName2 = _nv(nv, 'Resource Name 2');
  const resRole2 = _nv(nv, 'Resource Role 2');
  const resName3 = _nv(nv, 'Resource Name 3');
  const resRole3 = _nv(nv, 'Resource Role 3');
  const resName4 = _nv(nv, 'Resource Name 4');
  const resRole4 = _nv(nv, 'Resource Role 4');

  // buat Event ID unik dan simpan di kolom ke-17
  const eventId = Utilities.getUuid();
  sheet.getRange(lastRow, 17).setValue(eventId);
}
```

Gambar 3.17 Screenshot Code Sistem Approval Proyek PM 2

Pada Gambar 3.17 dapat dilihat bahwa potongan kode yang ditampilkan merupakan bagian utama dari proses penerimaan dan pengolahan data pengajuan proyek pada sistem *Approval Project* PM. Fungsi `onFormSubmit(e)` dijalankan secara otomatis setiap kali pengguna mengirimkan formulir pengajuan proyek, sehingga seluruh data proyek dapat dicatat dan diproses tanpa memerlukan input manual tambahan.

Pada tahap awal, sistem mengambil data input yang dikirimkan melalui Google Form menggunakan objek `namedValues`. Pendekatan ini memungkinkan sistem membaca jawaban berdasarkan nama pertanyaan pada formulir, sehingga lebih fleksibel dan mudah disesuaikan apabila terjadi perubahan struktur pertanyaan di kemudian hari. Data tersebut kemudian diproses dan disiapkan untuk disimpan ke dalam Google Sheets sebagai basis data utama sistem.

Sistem *Approval Project* PM dirancang dengan sejumlah *field* atau kolom pertanyaan yang merepresentasikan informasi penting dalam pengajuan proyek. Informasi tersebut mencakup nama *Project Manager* yang bertanggung jawab atas

proyek, alamat email pengaju, nomor referensi COF atau IPOF, nama *Practice Leader*, serta nama klien yang menjadi pihak terkait dalam proyek tersebut. Seluruh data ini berfungsi sebagai identitas utama proyek yang diajukan untuk proses *approval*.

Selain informasi identitas proyek, sistem juga mencatat periode pelaksanaan proyek melalui *field* Periode Project. Informasi ini penting untuk memberikan gambaran durasi proyek kepada pihak *Practice Leader* atau pihak yang berwenang dalam proses persetujuan. Dengan adanya informasi periode proyek, proses evaluasi dan pengambilan keputusan dapat dilakukan dengan lebih terstruktur.

Sistem ini juga mendukung pencatatan data sumber daya (*resources*) yang terlibat dalam proyek. Pada bagian ini, formulir menyediakan beberapa *field* untuk mengisi nama sumber daya serta peran atau role masing-masing dalam proyek. Data tersebut mencakup *Resource Name* dan *Resource Role* yang dapat diisi hingga beberapa entri, sehingga sistem mampu mengakomodasi proyek dengan lebih dari satu anggota tim. Pendekatan ini memungkinkan pendataan tim proyek dilakukan secara lebih rinci dan terdokumentasi dengan baik.

Setiap nilai yang diambil dari Google Form diproses menggunakan fungsi pendukung `_nv()` yang bertugas melakukan validasi dasar terhadap data input. Fungsi ini memastikan bahwa data yang diproses telah dibersihkan dari nilai kosong atau spasi berlebih, sehingga meminimalkan risiko kesalahan pencatatan pada Spreadsheet.

Pada bagian akhir proses, sistem secara otomatis menghasilkan Event ID unik untuk setiap pengajuan proyek menggunakan fungsi pembangkitan UUID. Identitas ini kemudian disimpan pada kolom tertentu di Google Sheets sebagai penanda unik setiap pengajuan. Pemberian Event ID ini bertujuan untuk memudahkan proses pelacakan, pencarian, serta referensi proyek selama tahapan *approval* berlangsung.

Implementasi proses penerimaan dan pencatatan data seperti yang ditampilkan pada Gambar 3.17 menunjukkan bahwa sistem *Approval Project* PM dikembangkan untuk mendukung proses administrasi proyek secara terstruktur dan

digital. Seluruh data pengajuan proyek dapat dicatat secara otomatis, konsisten, dan terdokumentasi dengan baik, sehingga mempermudah proses peninjauan dan persetujuan oleh pihak terkait, khususnya pada divisi *PM Diagnostic* dan *PM Learning*.

```
// Tentukan email Practice Leader dari mapping (fallback jika tidak ditemukan)
let emailPL = PRACTICE_LEADER_EMAILS[practiceLeader];
if (!emailPL) {
  Logger.log('Nama Practice Leader tidak ditemukan: ' + practiceLeader + '. Gunakan email default.');
```

The image shows a screenshot of a code editor with a light blue background. It contains JavaScript code for determining the email of a Practice Leader. The code uses a mapping object PRACTICE_LEADER_EMAILS and a fallback to a default email if the leader is not found. It also constructs URLs for approving or rejecting a project.

```
    emailPL = DEFAULT_PL_EMAIL;
}

// buat link approve/reject
const approveUrl = `${WEB_APP_URL}?action=approve&id=${encodeURIComponent(eventId)}`;
const rejectUrl = `${WEB_APP_URL}?action=reject&id=${encodeURIComponent(eventId)}`;
```

Gambar 3.18 Screenshot Code Sistem Approval Proyek PM 3

Pada Gambar 3.18 dapat dilihat bahwa potongan kode yang ditampilkan merupakan bagian penting dalam sistem *Approval Project PM* yang berfungsi untuk menentukan pihak yang berwenang memberikan persetujuan proyek, serta membentuk tautan digital untuk proses *approve* dan *reject* secara daring. Tahapan ini menjadi inti dari mekanisme approval karena menghubungkan pengajuan proyek dengan pihak pengambil keputusan.

Pada tahap awal, sistem menentukan alamat email *Practice Leader* berdasarkan pemetaan yang telah didefinisikan pada konfigurasi sistem. Pemetaan ini digunakan untuk menyesuaikan alamat email tujuan sesuai dengan divisi proyek yang diajukan, yaitu *PM Diagnostic* atau *PM Learning*. Dengan pendekatan ini, setiap pengajuan proyek dapat langsung dikirimkan kepada *Practice Leader* yang sesuai dengan bidang atau divisi terkait.

Apabila alamat email *Practice Leader* tidak ditemukan dalam pemetaan yang telah ditentukan, sistem secara otomatis menggunakan alamat email cadangan (default email). Mekanisme fallback ini diterapkan untuk menjaga keandalan sistem agar proses approval tetap dapat berjalan meskipun terjadi ketidaksesuaian data input, seperti kesalahan penulisan nama *Practice Leader* atau perubahan struktur organisasi.

Selanjutnya, sistem membentuk tautan khusus untuk proses persetujuan dan penolakan proyek. Tautan ini dibuat dengan memanfaatkan Web App URL yang telah di *deploy* sebelumnya, kemudian dikombinasikan dengan parameter aksi (*approve* atau *reject*) serta identitas unik proyek (Event ID). Setiap tautan memiliki parameter yang berbeda, sehingga sistem dapat membedakan apakah tindakan yang dilakukan adalah persetujuan atau penolakan proyek.

Penggunaan Event ID pada setiap tautan bertujuan untuk memastikan bahwa keputusan yang diambil oleh *Practice Leader* merujuk pada pengajuan proyek yang tepat. Dengan demikian, sistem dapat menghindari kesalahan persetujuan akibat data yang tertukar atau tidak sesuai.

Implementasi mekanisme penentuan email dan pembuatan tautan *approval* seperti yang ditampilkan pada Gambar 3.18 memungkinkan proses persetujuan proyek dilakukan secara digital dan terintegrasi. *Practice Leader* tidak perlu lagi menandatangani dokumen fisik atau melakukan konfirmasi melalui pesan terpisah, karena seluruh proses *approval* dapat dilakukan langsung melalui tautan yang disediakan oleh sistem.

Secara keseluruhan, pengembangan fitur ini mendukung tujuan utama sistem *Approval Project PM*, yaitu menyederhanakan dan mempercepat proses pendataan serta persetujuan proyek pada divisi PM *Diagnostic* dan PM *Learning*. Alur kerja menjadi lebih efisien, terdokumentasi, dan transparan, sekaligus mengurangi ketergantungan pada proses manual yang sebelumnya digunakan.

UNIVERSITAS
MULTIMEDIA
NUSANTARA

```
// HTML email ke Practice Leader (with buttons)
const htmlBody = `
<p>Yth. ${practiceLeader},</p>
<p>Berikut ada permintaan Approval untuk <b>Project PM Diagnostic</b>.</p>
<p><b>Data Form:</b></p>
<ul>
<li>👤 Project Manager: ${projectManager}</li>
<li>✉ Email: ${email}</li>
<li>📄 COF/IPOF NUMBER: ${cofNumber}</li>
<li>👤 Practice Leader: ${practiceLeader}</li>
<li>👤 Client Name: ${clientName}</li>
<li>📅 Periode Project: ${periodeProject}</li>
<li>👤 Resource 1: ${resName1} (${resRole1})</li>
<li>👤 Resource 2: ${resName2} (${resRole2})</li>
<li>👤 Resource 3: ${resName3} (${resRole3})</li>
<li>👤 Resource 4: ${resName4} (${resRole4})</li>
</ul>
<p>Mohon untuk ditindaklanjuti sesuai dengan prosedur.</p>
<p>
<a href="${approveUrl}" style="background:#28a745;color:white;padding:10px 20px;border-radius:5px;text-decoration:none;">✅ Approve</a>
&nbsp;
<a href="${rejectUrl}" style="background:#dc3545;color:white;padding:10px 20px;border-radius:5px;text-decoration:none;">❌ Reject</a>
</p>
<p>Terima kasih.</p>
`;

MailApp.sendEmail({
  to: emailPL,
  subject: 'Permintaan Approval Project PM Diagnostic',
  htmlBody: htmlBody
});
```

Gambar 3.19 Screenshot Code Sistem Approval Proyek PM 4

Pada Gambar 3.19 dapat dilihat bahwa potongan kode yang ditampilkan merupakan bagian dari sistem *Approval Project PM* yang berfungsi untuk membentuk dan mengirimkan email persetujuan proyek kepada *Practice Leader* menggunakan format HTML. Penggunaan email berbasis HTML dipilih agar informasi proyek dapat disajikan secara lebih terstruktur, mudah dibaca, dan dilengkapi dengan tombol aksi yang interaktif. Pada bagian awal, sistem menyusun isi email dalam variabel `htmlBody`. Email diawali dengan sapaan personal kepada *Practice Leader* sesuai dengan data yang diinputkan pada formulir. Pendekatan ini bertujuan untuk menciptakan komunikasi yang lebih formal dan profesional, sekaligus memastikan bahwa email diterima oleh pihak yang tepat.

Selanjutnya, sistem menampilkan informasi bahwa terdapat permintaan *approval* untuk proyek PM, dalam hal ini ditujukan untuk *Project PM Diagnostic*. Informasi ini ditampilkan secara eksplisit agar *Practice Leader* dapat langsung memahami konteks email tanpa harus membaca keseluruhan isi pesan secara mendetail.

Bagian utama dari email disusun dalam bentuk daftar (*unordered list*) yang memuat detail lengkap proyek yang diajukan. Informasi yang ditampilkan meliputi nama *Project Manager*, alamat email pemohon, nomor COF/IPOF, nama *Practice Leader*, nama klien, periode pelaksanaan proyek, serta daftar sumber daya

(*resource*) yang terlibat beserta peran masing-masing. Penyajian data dalam format daftar bertujuan agar informasi dapat dibaca secara cepat dan sistematis.

Penyertaan data sumber daya proyek, termasuk nama dan peran setiap anggota tim, memberikan gambaran menyeluruh kepada *Practice Leader* mengenai struktur dan kebutuhan proyek yang diajukan. Dengan informasi ini, *Practice Leader* dapat melakukan penilaian awal terkait kelayakan proyek, kecukupan sumber daya, serta kesesuaian periode pelaksanaan sebelum memberikan keputusan.

Pada bagian akhir isi email, sistem menambahkan dua tombol aksi, yaitu tombol *Approve* dan *Reject*, yang masing-masing terhubung dengan tautan khusus yang telah dibentuk sebelumnya. Tombol-tombol ini dirancang menggunakan gaya visual yang berbeda untuk membedakan tindakan persetujuan dan penolakan, sehingga meminimalkan risiko kesalahan klik oleh pengguna. Tombol *Approve* akan mengarahkan *Practice Leader* ke tautan yang menandai proyek sebagai disetujui, sedangkan tombol *Reject* akan menandai proyek sebagai ditolak. Kedua tautan tersebut membawa parameter Event ID proyek, sehingga sistem dapat mengidentifikasi proyek yang bersangkutan secara akurat saat keputusan diambil.

Setelah isi email HTML selesai disusun, sistem menggunakan fungsi `MailApp.sendEmail()` untuk mengirimkan email tersebut ke alamat email *Practice Leader* yang telah ditentukan pada tahap konfigurasi sebelumnya. Pengiriman email dilakukan secara otomatis setiap kali terdapat pengajuan *approval* proyek baru, tanpa memerlukan intervensi manual dari tim *IT Support*.

Implementasi mekanisme email *approval* berbasis HTML seperti yang ditampilkan pada Gambar 3.19 memberikan dampak signifikan terhadap efisiensi proses persetujuan proyek. Proses yang sebelumnya membutuhkan pertukaran dokumen fisik atau konfirmasi berulang melalui pesan pribadi kini dapat dilakukan secara digital, cepat, dan terdokumentasi dengan baik.

Secara keseluruhan, fitur ini mendukung tujuan utama sistem *Approval Project* PM, yaitu menyederhanakan alur persetujuan proyek pada divisi *PM Diagnostic* dan *PM Learning*. Dengan adanya email interaktif yang terintegrasi langsung

dengan sistem, proses approval menjadi lebih transparan, terkontrol, dan selaras dengan upaya digitalisasi operasional di PT Dayalima Abisatya.

```
* WEB APP: HANDLE APPROVE / REJECT
* - Update kolom Status (kolom ke-16)
* - Kirim notifikasi ke Project Manager (format sesuai permintaan)
*****/
function doGet(e) {
  const action = e.parameter.action; // 'approve' atau 'reject'
  const eventId = e.parameter.id;

  if (!action || !eventId) {
    return ContentService.createTextOutput('Missing parameters');
  }

  const ss = SpreadsheetApp.openById(SPREADSHEET_ID);
  const sheet = ss.getSheets()[0];
  const data = sheet.getDataRange().getValues();

  const statusText = (action === 'approve') ? 'Approved' : 'Rejected';

  // variabel yang akan diisi dari sheet
  let projectManager = '', emailPM = '', cofNumber = '', practiceLeader = '', clientName = '', periodeProject = '';
  let resName1 = '', resRole1 = '', resName2 = '', resRole2 = '', resName3 = '', resRole3 = '', resName4 = '', resRole4 = '';
```

Gambar 3.20 Screenshot Code Sistem Approval Proyek PM 5

Pada Gambar 3.20 dapat dilihat bahwa potongan kode yang ditampilkan merupakan bagian dari Web App Google Apps Script yang berfungsi untuk menangani aksi persetujuan (*approve*) dan penolakan (*reject*) proyek pada sistem *Approval Project PM*. Kode ini dijalankan ketika *Practice Leader* menekan tombol *Approve* atau *Reject* yang terdapat pada email persetujuan proyek yang sebelumnya dikirimkan oleh sistem.

Proses dimulai melalui fungsi *doGet(e)*, yang secara otomatis dipanggil ketika tautan approval diakses melalui *browser*. Fungsi ini menerima parameter dari URL, yaitu *action* yang berisi nilai *approve* atau *reject*, serta *eventId* yang merupakan identitas unik dari proyek yang diajukan. Kedua parameter ini menjadi komponen penting untuk memastikan bahwa tindakan yang dilakukan sesuai dengan proyek yang bersangkutan. Pada tahap awal, sistem melakukan validasi terhadap parameter yang diterima. Apabila parameter *action* atau *eventId* tidak tersedia, sistem akan menghentikan proses dan menampilkan pesan kesalahan. Validasi ini diterapkan untuk menjaga keamanan dan keandalan sistem agar tidak memproses permintaan yang tidak lengkap atau tidak valid.

Setelah parameter tervalidasi, sistem membuka Google Sheets yang telah ditentukan sebagai basis data proyek menggunakan *SPREADSHEET_ID*. Spreadsheet ini berfungsi sebagai media utama penyimpanan seluruh data

pengajuan proyek PM, termasuk informasi proyek, sumber daya, dan status persetujuan.

Selanjutnya, sistem menentukan nilai status proyek berdasarkan aksi yang dilakukan oleh *Practice Leader*. Jika aksi yang diterima adalah *approve*, maka status proyek akan diubah menjadi *Approved*. Sebaliknya, jika aksi yang diterima adalah *reject*, status proyek akan diubah menjadi *Rejected*. Penentuan status ini dilakukan secara otomatis untuk memastikan konsistensi data dan menghindari kesalahan input manual.

Pada bagian berikutnya, sistem menyiapkan sejumlah variabel untuk menampung data proyek yang akan diambil dari Spreadsheet, seperti nama *Project Manager*, alamat email *Project Manager*, nomor COF/IPOF, nama *Practice Leader*, nama klien, periode proyek, serta daftar sumber daya dan peran yang terlibat. Data ini diperlukan untuk keperluan pembaruan status dan pengiriman notifikasi lanjutan.

Keberadaan mekanisme *Web App* pada sistem approval ini memungkinkan proses persetujuan proyek dilakukan sepenuhnya secara digital dan *real-time*. *Practice Leader* tidak perlu mengakses Spreadsheet secara langsung atau melakukan konfirmasi terpisah melalui email atau pesan pribadi, karena seluruh aksi *approval* sudah terintegrasi langsung dengan sistem.

Implementasi kode seperti yang ditampilkan pada Gambar 3.20 memberikan kontribusi signifikan terhadap efisiensi dan akurasi proses persetujuan proyek. Setiap keputusan yang diambil tercatat secara otomatis di dalam sistem, lengkap dengan identitas proyek dan status terbaru, sehingga memudahkan proses *monitoring* dan audit di kemudian hari.

Secara keseluruhan, mekanisme penanganan approve dan reject melalui *Web App* ini mendukung tujuan sistem *Approval Project PM* dalam menciptakan alur persetujuan yang cepat, terstruktur, dan terdokumentasi dengan baik. Proses ini juga sejalan dengan upaya digitalisasi internal perusahaan dalam mengurangi

ketergantungan pada proses manual dan meningkatkan transparansi pengelolaan proyek di lingkungan PT Dayalima Abisatya.

```
// cari baris dengan Event ID di kolom ke-17 (index 16)
for (let i = 1; i < data.length; i++) {
  if (String(data[i][16]).trim() === eventId) {
    projectManager = data[i][1];
    emailPM        = data[i][2];
    cofNumber      = data[i][3];
    practiceLeader  = data[i][4];
    clientName     = data[i][5];
    periodeProject = data[i][6]; // <-- Periode Project ambil dari index 6
    resName1       = data[i][7];
    resRole1       = data[i][8];
    resName2       = data[i][9];
    resRole2       = data[i][10];
    resName3       = data[i][11];
    resRole3       = data[i][12];
    resName4       = data[i][13];
    resRole4       = data[i][14];

    // update kolom Status di kolom ke-16 (index 15)
    sheet.getRange(i + 1, 16).setValue(statusText);
    break;
  }
}
```

Gambar 3.21 Screenshot Code Sistem Approval Proyek PM 6

Pada Gambar 3.21 dapat dilihat bahwa potongan kode yang ditampilkan merupakan bagian lanjutan dari proses *approve* dan *reject* pada sistem *Approval Project* PM, yang berfungsi untuk mencari data proyek yang sesuai berdasarkan Event ID serta memperbarui status persetujuan proyek pada Google Sheets. Bagian kode ini menjadi penghubung antara aksi persetujuan yang dilakukan melalui email dan pembaruan data proyek secara otomatis di basis data sistem.

Proses diawali dengan perulangan (*looping*) yang digunakan untuk menelusuri seluruh baris data pada Spreadsheet, dimulai dari baris kedua karena baris pertama digunakan sebagai *header*. Sistem membandingkan nilai Event ID yang tersimpan pada kolom ke-17 Spreadsheet dengan *eventId* yang diterima dari parameter URL. Perbandingan ini dilakukan untuk memastikan bahwa sistem hanya memproses proyek yang benar-benar sesuai dengan aksi persetujuan yang dilakukan oleh *Practice Leader*.

Ketika Event ID yang sesuai ditemukan, sistem kemudian mengambil seluruh data proyek yang berkaitan dengan baris tersebut. Data yang diambil meliputi informasi *Project Manager*, email *Project Manager*, nomor COF/IPOF, nama *Practice Leader*, nama klien, periode proyek, serta daftar sumber daya dan peran yang terlibat dalam proyek. Pengambilan data ini dilakukan dengan mengacu pada indeks kolom masing-masing, sehingga sistem dapat membaca informasi proyek secara terstruktur dan konsisten.

Setelah data proyek berhasil diidentifikasi, sistem melakukan pembaruan status proyek pada kolom status yang berada di kolom ke-16 Spreadsheet. Nilai status yang dituliskan sebelumnya telah ditentukan berdasarkan aksi yang dilakukan, yaitu *Approved* atau *Rejected*. Pembaruan status ini dilakukan secara otomatis tanpa memerlukan intervensi manual dari tim *Project Management* maupun *IT Support*.

Penggunaan mekanisme pencarian berbasis Event ID ini memastikan bahwa setiap keputusan persetujuan hanya diterapkan pada satu proyek yang spesifik, sehingga meminimalkan risiko kesalahan pembaruan data. Selain itu, pendekatan ini membantu sistem menangani banyak pengajuan proyek secara bersamaan tanpa terjadi tumpang tindih data.

```
// kirim notifikasi ke PM jika email tersedia
if (emailPM) {
  const subject = `Permintaan Project PM Diagnostic ${statusText}`;
  const htmlBody = `
    <p>Yth. ${projectManager},</p>
    <p>Menginformasikan bahwa Permintaan Approval untuk Project PM Diagnostic Anda dengan data sebagai berikut :</p>
    <ul>
      <li>👤 Project Manager: ${projectManager}</li>
      <li>✉ Email: ${emailPM}</li>
      <li>📄 COF/IPOF NUMBER: ${cofNumber}</li>
      <li>👤 Practice Leader: ${practiceLeader}</li>
      <li>👤 Client Name: ${clientName}</li>
      <li>📅 Periode Project: ${periodeProject}</li>
      <li>👤 Resource 1: ${resName1} (${resRole1})</li>
      <li>👤 Resource 2: ${resName2} (${resRole2})</li>
      <li>👤 Resource 3: ${resName3} (${resRole3})</li>
      <li>👤 Resource 4: ${resName4} (${resRole4})</li>
    </ul>
    <p>Telah di <b>${statusText}</b> oleh <b>${practiceLeader}</b>.</p>
    <p>Sekian terima kasih.</p>
  `;

  MailApp.sendEmail({
    to: emailPM,
    subject: subject,
    htmlBody: htmlBody
  });
}

return HtmlService.createHtmlOutput(`<h2>Request ${statusText}</h2><p>COF/IPOF NUMBER: ${cofNumber}</p>`);
```

Gambar 3.22 Screenshot Code Sistem Approval Proyek PM 7

Pada Gambar 3.22 dapat dilihat bahwa potongan kode yang ditampilkan merupakan bagian akhir dari alur proses sistem *Approval Project PM*, yang berfungsi untuk mengirimkan notifikasi hasil persetujuan proyek kepada *Project Manager* setelah *Practice Leader* melakukan aksi *approve* atau *reject*. Bagian kode ini memastikan bahwa setiap keputusan yang diambil pada tahap persetujuan dapat segera diketahui oleh pihak yang bertanggung jawab atas proyek tersebut.

Proses diawali dengan pengecekan keberadaan alamat email *Project Manager* melalui kondisi if (emailPM). Pengecekan ini dilakukan untuk memastikan bahwa notifikasi hanya dikirim apabila data email *Project Manager* tersedia, sehingga sistem dapat menghindari kegagalan pengiriman email akibat data yang tidak lengkap.

Selanjutnya, sistem membentuk subjek email yang berisi informasi jenis proyek, yaitu *Project PM Diagnostic*, serta status hasil persetujuan yang telah ditentukan sebelumnya dalam variabel statusText. Penggunaan status *Approved* atau *Rejected* pada subjek email bertujuan agar *Project Manager* dapat langsung memahami hasil keputusan tanpa harus membuka isi email secara keseluruhan.

Pada bagian isi email, sistem menyusun pesan dalam format HTML yang bersifat informatif dan terstruktur. Isi email diawali dengan sapaan kepada *Project Manager*, kemudian dilanjutkan dengan pemberitahuan bahwa permintaan approval proyek telah diproses. Setelah itu, ditampilkan rincian lengkap data proyek, yang meliputi nama *Project Manager*, alamat email, nomor COF/IPOF, nama *Practice Leader*, nama klien, periode proyek, serta daftar sumber daya beserta peran masing-masing yang terlibat dalam proyek.

Penyajian informasi proyek secara rinci dalam email bertujuan untuk memberikan konteks yang jelas terkait proyek yang dimaksud, sehingga *Project Manager* dapat langsung melakukan tindak lanjut sesuai dengan hasil persetujuan yang diterima. Selain itu, sistem juga menampilkan informasi mengenai siapa *Practice Leader* yang memberikan keputusan, sehingga alur tanggung jawab dan pengambilan keputusan dapat ditelusuri dengan jelas.

Setelah format email selesai disusun, sistem menjalankan fungsi `MailApp.sendEmail()` untuk mengirimkan notifikasi ke alamat email *Project Manager*. Proses pengiriman ini dilakukan secara otomatis dan *real-time* setelah status proyek diperbarui di Spreadsheet, sehingga tidak terjadi jeda antara keputusan persetujuan dan penyampaian informasi kepada pihak terkait.

Pada bagian akhir, sistem mengembalikan tampilan halaman sederhana melalui `HtmlService.createHtmlOutput()` yang menampilkan informasi status permintaan. Tampilan ini berfungsi sebagai konfirmasi visual bahwa proses *approval* atau *rejection* telah berhasil diproses oleh sistem, sekaligus menjadi penanda bahwa alur persetujuan telah selesai dijalankan.

3.2.4 Update & Maintenance

Kegiatan update dan maintenance sistem dilakukan secara berkala, khususnya pada periode akhir pelaksanaan magang sebelum masa kontrak berakhir. Tahapan ini bertujuan untuk memastikan bahwa seluruh sistem yang telah dikembangkan tetap berjalan dengan baik dan siap digunakan oleh masing-masing divisi. Update dan maintenance dilakukan sebagai bagian dari proses evaluasi akhir terhadap sistem *booking* kendaraan, sistem *Ticketing IT Support*, dan sistem *approval Project PM*.

Pada tahap ini, dilakukan proses *testing run* terhadap setiap sistem yang telah dibuat untuk memastikan seluruh fungsi berjalan sesuai dengan perancangannya. Pengujian meliputi pengecekan alur *input data*, proses otomatisasi, pengiriman notifikasi, serta pembaruan *status* pada sistem. Apabila dalam proses pengujian ditemukan kendala atau *bug*, maka dilakukan perbaikan pada *script* Google Apps Script yang digunakan agar sistem dapat kembali berfungsi secara normal.

Selain pengujian teknis, kegiatan maintenance juga mencakup komunikasi langsung dengan divisi pengguna sistem. Apabila tidak ditemukan kendala secara teknis, dilakukan konfirmasi kepada pihak terkait mengenai pengalaman penggunaan sistem, termasuk adanya hambatan atau kesulitan dalam pengoperasian. Masukan yang diperoleh dari pengguna digunakan sebagai bahan evaluasi untuk menentukan apakah diperlukan penyesuaian atau penyempurnaan

sistem lebih lanjut. Dengan adanya proses update dan maintenance ini, sistem yang dikembangkan diharapkan tetap stabil, fungsional, dan sesuai dengan kebutuhan operasional PT Dayalima Abisatya.

3.3 Kendala yang Ditemukan

Selama melaksanakan kegiatan magang dan proses pengembangan sistem di PT Dayalima Abisatya, terdapat beberapa kendala yang dihadapi. Kendala-kendala tersebut dijabarkan sebagai berikut.

1. Penggunaan Google Apps Script sebagai Bahasa Pemrograman Baru

Salah satu kendala utama yang dihadapi adalah keharusan untuk menggunakan Google Apps Script sebagai bahasa pemrograman inti dalam pengembangan seluruh sistem otomatisasi. Google Apps Script merupakan lingkungan yang baru dan belum pernah digunakan sebelumnya, sehingga proses adaptasi membutuhkan waktu yang cukup panjang. Selain itu, tidak terdapat *senior* atau mentor teknis yang secara khusus dapat memberikan bimbingan karena seluruh staf IT telah memiliki tanggung jawab pada proyek mereka masing-masing. Akibatnya, proses pembelajaran dilakukan secara mandiri bersama rekan magang lainnya, yang membuat proses pengembangan sistem menjadi lebih menantang dan memerlukan upaya tambahan dalam memahami dokumentasi dan sintaks Google Script.

2. Komunikasi Antar Tim yang Tidak Konsisten

Kendala berikutnya terkait dengan komunikasi lintas divisi yang tidak selalu berjalan dengan jelas dan terstruktur. Beberapa unit pengguna (*user*) memberikan informasi yang tidak lengkap atau berubah-ubah selama proses pengembangan sistem, sehingga mempengaruhi *timeline* penyelesaian proyek. Kurangnya detail pada penjelasan kebutuhan sering menyebabkan interpretasi yang berbeda, revisi berulang, serta penundaan pada tahap implementasi. Kondisi ini berdampak pada keterlambatan penyelesaian sistem tertentu dan menunda dimulainya proyek berikutnya.

3. Munculnya Bug pada Sistem Akibat Kurangnya Pengalaman

Karena Google Apps Script merupakan teknologi baru yang dipelajari selama masa magang, proses pengembangan sering kali menghasilkan *bug* atau *error*

pada sistem. Mulai dari kesalahan logika, fungsi yang tidak berjalan stabil, hingga integrasi antara Google Forms, Spreadsheets, dan email yang tidak sesuai ekspektasi. Penyelesaian *bug* membutuhkan proses *trial-and-error* yang cukup intensif, ditambah dengan keterbatasan sumber daya teknis membuat proses *debugging* berlangsung lebih lama dari perkiraan. Meski demikian, kendala ini juga menjadi kesempatan untuk mempelajari alur *debugging* dan meningkatkan kemampuan teknis dalam pemrograman.

4. Tenggat Waktu (Deadline) yang Ketat

Beberapa proyek memiliki batas waktu penyelesaian yang cukup singkat, sementara proses perancangan sistem membutuhkan pemahaman mendalam terhadap kebutuhan *user* dan alur kerja internal. Kombinasi antara waktu pengerjaan yang terbatas, kebutuhan revisi, dan *debugging* yang memakan waktu menyebabkan tekanan tambahan selama proses pengembangan. *Deadline* yang ketat ini membuat manajemen waktu menjadi tantangan tersendiri, terutama ketika beberapa sistem dikerjakan secara paralel.

3.4 Solusi atas Kendala yang Ditemukan

Untuk mengatasi berbagai kendala yang muncul selama pelaksanaan magang dan proses pengembangan sistem di PT Dayalima Abisatya, beberapa solusi berikut diterapkan agar seluruh kegiatan dapat tetap berjalan dengan baik dan menghasilkan output yang sesuai kebutuhan perusahaan.

1. Pendalaman Google Apps Script Secara Mandiri dan Berkelanjutan

Mengingat keterbatasan pembimbing teknis dan minimnya pengalaman sebelumnya dalam menggunakan Google Apps Script, solusi yang dilakukan adalah mempelajari *platform* tersebut secara mandiri melalui dokumentasi resmi Google, forum komunitas *developer*, serta referensi dari proyek serupa. Selain itu, dilakukan eksplorasi terhadap *library* bawaan, fungsi dasar, *trigger* otomatis, dan integrasi antara Google Forms, Spreadsheets,

serta Gmail. Pembelajaran dilakukan secara bertahap dan berulang sambil mengerjakan proyek langsung, sehingga pemahaman meningkat seiring proses pengembangan. Upaya ini cukup efektif membantu mengurangi hambatan teknis dan mempercepat proses adaptasi terhadap bahasa pemrograman baru.

2. Melakukan Klarifikasi dan Koordinasi Berulang dengan User

Untuk mengatasi komunikasi antar tim yang kurang konsisten, upaya yang dilakukan adalah menjadwalkan sesi klarifikasi tambahan dengan *user* terkait (GA, PMO, HR, maupun IT Support). Pertemuan tambahan ini dimanfaatkan untuk memastikan kebutuhan sistem benar-benar dipahami dengan detail sebelum masuk ke tahap *coding*. Selain itu, setiap perubahan kebutuhan langsung dicatat dan dikonfirmasi ulang melalui pesan tertulis agar tidak terjadi misinterpretasi. Langkah ini membantu mereduksi revisi besar di tengah pengerjaan serta memberikan alur komunikasi yang lebih jelas selama pengembangan sistem berlangsung.

3. Penerapan Metode Debugging Bertahap dan Pengujian Berulang

Untuk mengatasi munculnya bug pada sistem, solusi yang diterapkan adalah melakukan *debugging* secara bertahap sesuai modul yang sedang dikerjakan. Setiap fungsi diuji secara terpisah sebelum digabungkan ke dalam sistem final. Selain itu, dilakukan simulasi pengisian *form*, pengecekan *response* email, dan *monitoring* data di Google Sheets untuk memastikan seluruh mekanisme berjalan sesuai rencana. Pengujian berulang bersama *user* juga dilakukan agar *error* dapat ditemukan dengan cepat dan diperbaiki sebelum sistem digunakan secara resmi. Pendekatan ini membantu meningkatkan stabilitas dan keandalan sistem.

4. Manajemen Waktu yang Lebih Terstruktur untuk Menghadapi Deadline

Mengingat tenggat waktu beberapa proyek cukup ketat, solusi yang dilakukan adalah membuat prioritas pekerjaan berdasarkan tingkat urgensi dan kompleksitas sistem. Setiap proyek dipecah menjadi bagian-bagian kecil agar progres bisa dicapai secara bertahap. Dengan membagi waktu secara

lebih terstruktur, misalnya fokus pada analisis kebutuhan di pagi hari dan pengembangan kode di sore hari, sehingga pekerjaan menjadi lebih terkontrol dan dapat selesai tepat waktu. Selain itu, komunikasi rutin dengan *Head of IT* membantu menyesuaikan prioritas ketika ada perubahan kebutuhan mendadak atau penyesuaian *timeline*.

