BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Struktur koordinasi yang dilaksanakan di divisi teknologi selama program magang berjalan, peserta magang secara langsung dibimbing oleh *CTO* (*Chief Technology Officer*). Pak Ali Mubarokah bertanggung jawab sebagai *CTO* (*Chief Technology Officer*) yang bertanggung jawab dalam semua perencanaan dan pengembangan yang menyangkut hal hal teknis sekaligus membimbing peserta magang dalam melakukan tugasnya yaitu mengembangkan aplikasi yang ditugaskan.

Dalam hal ini divisi teknologi bertanggung jawab dalam semua hal teknis yang berkaitan dengan pembuatan, pemeliharaan, pengembangan seluruh aplikasi yang sudah dibuat oleh perusahaan.

3.2 Tugas yang Dilakukan

Tugas yang dilakukan selama kegiatan magang ini meliputi pembuatan aplikasi baru maupun pengembangan aplikasi yang sudah berjalan. Pada kegiatan magang ini peserta magang berperan sebagai *fullstack developer* yang menangani pengembangan fitur pada sisi antarmuka dan sisi server, mulai dari analisis kebutuhan, perancangan, implementasi fitur, hingga pengujian. Pada tahap perancangan antarmuka, peserta magang menyusun rancangan tampilan menggunakan *Figma* dalam bentuk *wireframe* dan/atau *prototype* sebagai acuan sebelum implementasi, sehingga alur penggunaan dan konsistensi tampilan dapat divalidasi lebih awal bersama pihak terkait. Setelah rancangan disetujui, pengembangan antarmuka dilakukan menggunakan *cshtml*, sedangkan pengembangan sisi server dilakukan menggunakan *C#.* Selain itu, peserta magang juga melakukan integrasi aplikasi dengan API untuk mendukung pertukaran data antar sistem, mempercepat proses pengembangan, serta memperkuat aspek keamanan melalui mekanisme otorisasi dan validasi input.

Pada tahap pengujian, fitur yang dikembangkan diuji secara langsung melalui aplikasi untuk memastikan fungsi berjalan sesuai kebutuhan, termasuk validasi input, alur proses, serta hasil keluaran sistem. Pengujian dilakukan sebelum fitur digunakan oleh pengguna, dan hasil implementasi serta pengujian tersebut

didokumentasikan sebagai bukti bahwa pekerjaan telah melalui proses pengecekan dan memperoleh persetujuan dari atasan/pembimbing di perusahaan.

Basis data aplikasi dipelihara menggunakan *SQL Server* dengan struktur tabel yang dirancang terstruktur dan menerapkan normalisasi untuk mengurangi duplikasi data, mencegah anomali *insert/update/delete*, serta menjaga konsistensi melalui relasi *primary key* dan *foreign key*. Dalam aktivitas pemeliharaan data, peserta magang melakukan pengecekan, pembaruan, serta eksekusi query dan validasi data menggunakan alat bantu manajemen database (misalnya DBEaver) sesuai kebutuhan operasional.

Selain membuat atau mengembangkan aplikasi, selama proses magang peserta magang juga bekerja sama dengan divisi operasional dalam melakukan migrasi data tambahan untuk kebutuhan inisialisasi data penggajian. Kegiatan ini mencakup penyiapan format data, validasi kelengkapan dan konsistensi, serta verifikasi hasil migrasi agar mendukung akurasi perhitungan penggajian. Di samping itu, peserta magang turut membantu penyusunan dokumen standar ISO 27001 sebagai bagian dari dokumentasi resmi perusahaan terkait tata kelola dan keamanan informasi.

3.3 Uraian Pelaksanaan Magang

Tugas yang dilakukan selama magang setiap minggunya adalah sebagai berikut di Tabel 3.1 .

Tabel 3.1. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang

Minggu Ke -	Pekerjaan yang dilakukan
1	Mengeksplorasi google appsheet.
2	Mengembangkan aplikasi salescrm.
3	Mengembangkan aplikasi salescrm.
4	Menerapkan dan perbaiki *bug* salescrm.
5	Mengexplore integrasi dengan accurate dengan cara hit API dari accurate.
6	Mengexplore fitur dasbor untuk aplikasi salescrm.
7	*Setup* proyek rekonsiliasi dan membuat *file* HTML untuk antarmuka.
	Lanjut pada halaman berikutnya

Tabel 3.1 (lanjutan)

Minggu Ke -	Pekerjaan yang dilakukan
8	Membuat *controller* (sisi server), membuat *routing* di UI, memasukkan sumber daya gambar, dan menghubungkan basis data dengan aplikasi.
9	Membuat UI untuk CRUD.
10	Membuat UI untuk CRUD, membuat *controller*, dan menghubungkan *controller* views dengan basis data.
11	Membuat *controller* dan menghubungkan *controller* views dengan basis data.
12	Membuat *controller* dan menghubungkan *controller* views dengan basis data.
13	Membuat *controller*, menghubungkan *controller* views dengan basis data, dan membuat dasbor dengan Alat Business Intelligence.
14	*Setup* proyek management dan setting parameter baru di basis data dan view.
15	Membuat approval hierarchy, filter untuk hierarchy dan type, dan membuat dokumen internal *testing*.
16	Bahas kebutuhan aplikasi Validasi NIK, membuat Figma untuk tampilan, *setup* proyek, dan migrasi data additional.
17	*Setup* proyek validasi NIK (menyambungkan index dan *list*), membuat *table* di basis data, membuat UI untuk fitur unduh dan unduh templat, dan membuat fungsi unggah.
18	Membuat proses Validasi NIK di sisi server.
19	Membuat UI *login* dan *landing page* di Figma dan di *web app*.
20	Membuat proses isi ulang kuota di aplikasi dan penggunaannya, dan menghubungkan UI dengan login.
21	*Testing*, membuat dokumen *testing*, dan skrip untuk di dijalankan ke UAT (*User Acceptance Testing*).
22	*Testing* dan *bug* fixing aplikasi *Document Management* dan Validasi NIK.

Lanjut pada halaman berikutnya

Tabel 3.1 (lanjutan)

Minggu Ke -	Pekerjaan yang dilakukan
23	*Testing* dan *bug* fixing aplikasi *Document Management* dan Validasi NIK, explore helpdesk ai yang akan dikembangkan, dan menambah fitur di aplikasi *Document Management*.
24	*Testing*, *bug* fixing, dan Membuat panduan pengguna *Document Management*.

3.3.1 Uji Coba Integrasi *Accurate Online*

Uji coba integrasi dengan *accurate online* dengan menggunakan *C#* konsol untuk kebutuhan integrasi dengan aplikasi benemica yang berfungsi untuk create, save, dan view voucher jurnal yang ada di *accurate online*.

Proses integrasi ini yaitu dengan megirim request ke API dari *accurate online* untuk proses *create, save, dan view* voucher jurnal yang ada di *accurate online*.

Kode berikut merupakan fungsi utama yang mengambil input *API Token* dan *Signature Secret* yang berfungsi untuk autentikasi yang digunakan bersamaan dengan waktu yang diambil saat akan mengirim request yang sudah di enkripsi.

```
1
2       static async Task Main(string[] args)
3       {
4           Console.WriteLine("=== Accurate Online Journal Voucher
    App ===\n");
5
6           Console.Write("Masukkan API Token: ");
7           string apiToken = Console.ReadLine();
8
9           Console.Write("Masukkan Signature Secret: ");
10          string signatureSecret = Console.ReadLine();
11
12          string timestamp = DateTime.Now.ToString("dd/MM/yyyy
    HH:mm:ss");
13
14          string signature;
15          using (var hmac = new HMACSHA256(Encoding.UTF8.
    GetBytes(signatureSecret)))
```

```
        {
            byte[] hash = hmac.ComputeHash(Encoding.UTF8.
GetBytes(timestamp));
            signature = Convert.ToBase64String(hash);
        }

        using (var client = new HttpClient())
        {
            client.DefaultRequestHeaders.Add("Authorization",
$"Bearer {apiToken}");
            client.DefaultRequestHeaders.Add("X-Api-Timestamp"
, timestamp);
            client.DefaultRequestHeaders.Add("X-Api-Signature"
, signature);

            bool exit = false;
            while (!exit)
            {
                // Menu switch untuk user
            }
        }

        Console.WriteLine("\nSelesai. Tekan Enter untuk keluar
...");
        Console.ReadLine();
    }
```

Kode 3.1: Source code fungsi main Journal Voucher

Kode untuk ambil voucher jurnal

```

    static async Task GetJournalVoucherList(HttpClient client)
    {
        string urlList = "https://zeus.accurate.id/accurate/
api/journal-voucher/list.do?fields=id,number,transDate,
transDateView";

        Console.WriteLine("Mengambil list jurnal voucher...\n"
);

        HttpResponseMessage response = await client.GetAsync(
urlList);
        string responseJson = await response.Content.
ReadAsStringAsync();
```

```
10
11          Console.WriteLine("=== Journal Voucher List JSON ===")
    ;
12          Console.WriteLine(responseJson);
13
14          JObject jsonObj = JObject.Parse(responseJson);
15          if (jsonObj["s"] != null && (bool)jsonObj["s"])
16          {
17              foreach (var item in jsonObj["d"])
18              {
19                  int id = (int)item["id"];
20                  string number = item["number"]?.ToString();
21                  Console.WriteLine($"\n- ID: {id} | Number: {
    number}");
22
23                  await GetJournalVoucherDetail(client, id);
24              }
25          }
26      }
```

Kode 3.2: Source code untuk ambil Journal Voucher

Kode untuk menampilkan detail dari voucher jurnal

```
1
2       static async Task GetJournalVoucherDetail(HttpClient
    client, int journalId)
3       {
4           string urlDetail = $"https://zeus.accurate.id/accurate
    /api/journal-voucher/detail.do?id={journalId}";
5
6           HttpResponseMessage response = await client.GetAsync(
    urlDetail);
7           string jsonDetail = await response.Content.
    ReadAsStringAsync();
8
9           Console.WriteLine($"\n=== Detail Journal Voucher {
    journalId} ===");
10
11          JObject jsonObj = JObject.Parse(jsonDetail);
12          if (jsonObj["s"] != null && (bool)jsonObj["s"])
13          {
14              // Proses pengambilan detail
15          }
16          else
```

```
            {
                Console.WriteLine("Gagal mengambil detail jurnal voucher.");
            }
        }
```

Kode 3.3: Source code untuk ambil detail Journal Voucher

Berikut merupakan kode untuk membuat voucher jurnal.

```

        static async Task SaveJournalVoucher(HttpClient client)
        {
            Console.WriteLine("\n=== Buat Journal Voucher Baru ===");

            // Masukan Journal Voucher

            var jvData = new JObject
            {
                ["number"] = number,
                ["transDate"] = date,
                ["description"] = description,
                ["detailJournalVoucher"] = new JArray
            };

            string urlSave = "https://zeus.accurate.id/accurate/api/journal-voucher/save.do";

            // Proses save Journal Voucher

            Console.WriteLine("\n=== Response Save Journal Voucher ===");
            Console.WriteLine(result);
        }
```

Kode 3.4: Source code proses save Journal Voucher

Berikut merupakan kode untuk membuat banyak voucher jurnal dalam satu request.

```

        static async Task BulkSaveJournalVoucher(HttpClient client)
        {
            Console.WriteLine("\n=== Bulk Save Journal Voucher ===");
```

```
        Console.Write("Berapa banyak JV yang ingin dibuat? ");
        if (!int.TryParse(Console.ReadLine(), out int count)
|| count <= 0) return;

        var dataArray = new JArray();

        for (int i = 1; i <= count; i++)
        {
            // Bulk save Journal Voucher

            dataArray.Add(jvData);
        }

        var bulkData = new JObject
        {
            ["data"] = dataArray
        };

        string urlBulkSave = "https://zeus.accurate.id/
accurate/api/journal-voucher/bulk-save.do";

        // Proses bulk save Journal Voucher
    }
```

Kode 3.5: Source code proses bulk save Journal Voucher

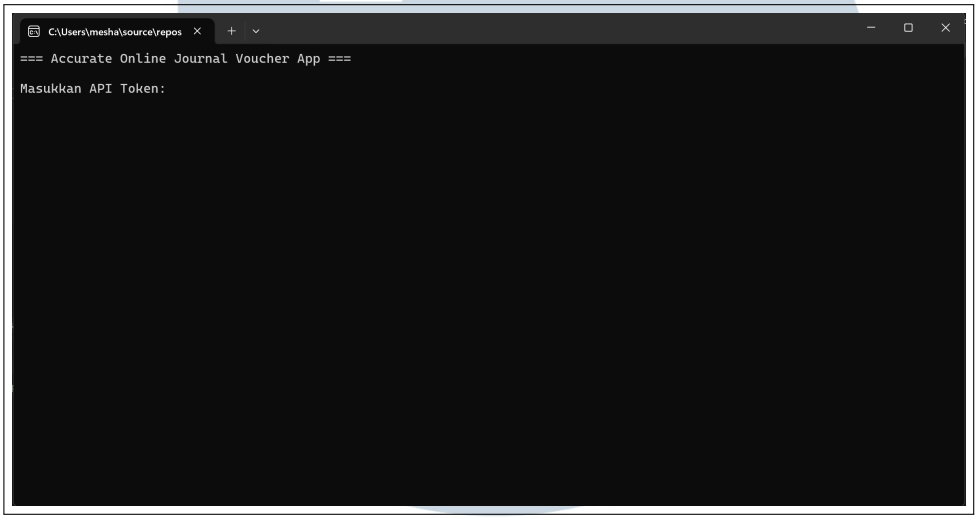
Berikut merupakan kode untuk menghapus voucher jurnal.

```

    static async Task DeleteJournalVoucher(HttpClient client)
    {
        Console.Write("\nMasukkan ID Journal Voucher yang
ingin dihapus: ");
        string id = Console.ReadLine();

        string urlDelete = $"https://zeus.accurate.id/accurate
/api/journal-voucher/delete.do?id={id}";

        HttpRequestMessage request = new HttpRequestMessage(
HttpMethod.Delete, urlDelete);

        HttpResponseMessage response = await client.SendAsync(
request);
        string result = await response.Content.
ReadAsStringAsync();
```

```
13
14          Console.WriteLine("\n=== Response Delete Journal
    Voucher ===");
15          Console.WriteLine(result);
16      }
```

Kode 3.6: Source code proses delete Journal Voucher

Berikut merupakan tampilan konsol untuk uji coba integrasi Voucher jurnal



Gambar 3.1. Konsol untuk hit API *Accurate Online*.

3.3.2 Document Management Web App

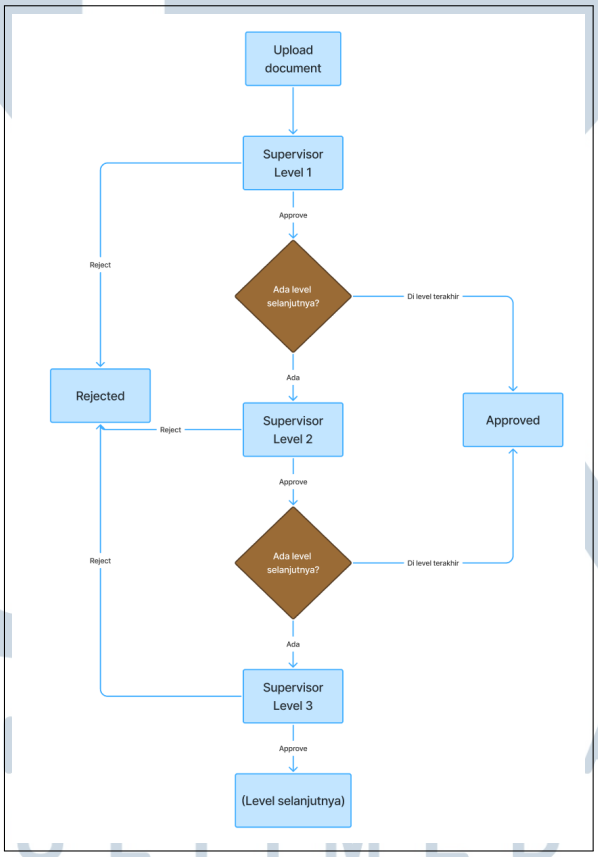
Aplikasi *Document Management* merupakan aplikasi berbasis *web* yang digunakan untuk menyimpan dan mengelola dokumen perusahaan secara terpusat. Selain penyimpanan dokumen, aplikasi menyediakan *metadata* dokumen (*tipe* dokumen, keterangan/*remark*, informasi pengunggah), fitur pengingat (*reminder*), serta mekanisme persetujuan untuk memastikan dokumen diproses sesuai alur kerja. Sebelum dilakukan pengembangan lebih lanjut, modul ini pada dasarnya hanya berfungsi sebagai tempat penyimpanan dokumen, sehingga belum mendukung pengelompokan berdasarkan *tipe* dokumen, pembatasan akses berdasarkan relevansi pengguna, maupun alur persetujuan bertingkat sesuai kebutuhan operasional.

Dalam pengembangan aplikasi ini, peserta magang melakukan *enhancement* pada modul *Document Management* agar pengguna dapat mengunggah dokumen berdasarkan *tipe* dokumen yang telah didaftarkan dan menjalankan proses persetujuan bertingkat sesuai hirarki organisasi. Akses pengguna dibatasi sehingga

pengguna hanya dapat melihat dokumen milik dirinya sendiri atau dokumen milik bawahannya, serta hanya pada *tipe* dokumen yang relevan. Pembatasan ini diterapkan pada sisi *back-end* untuk memastikan keamanan data dan mencegah pengguna mengakses dokumen yang tidak berkaitan.

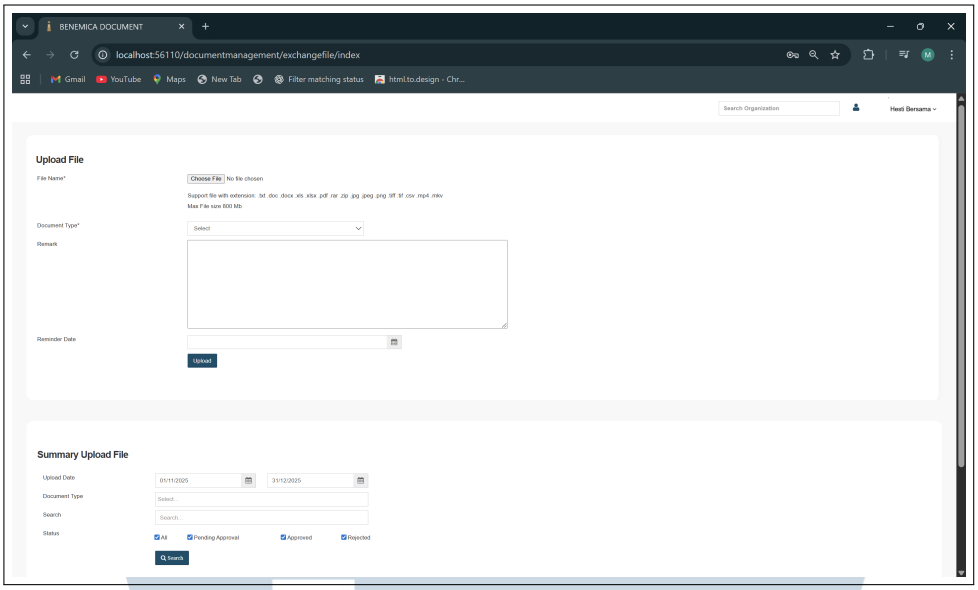
Aplikasi dikembangkan menggunakan *.NET Framework* dengan bahasa *C#* pada sisi *back-end* (*controller* dan logika bisnis), tampilan menggunakan *cshtml* pada sisi *front-end*, serta *JavaScript* untuk validasi input dan fungsi antarmuka yang diperlukan. Data *metadata* dokumen dan informasi hirarki pengguna disimpan pada *SQL Server*, sedangkan *file* dokumen disimpan pada media penyimpanan yang terintegrasi dengan aplikasi.

Berikut merupakan *flowchart* dari cara kerja *document management* yang sudah di menyempurnakan.



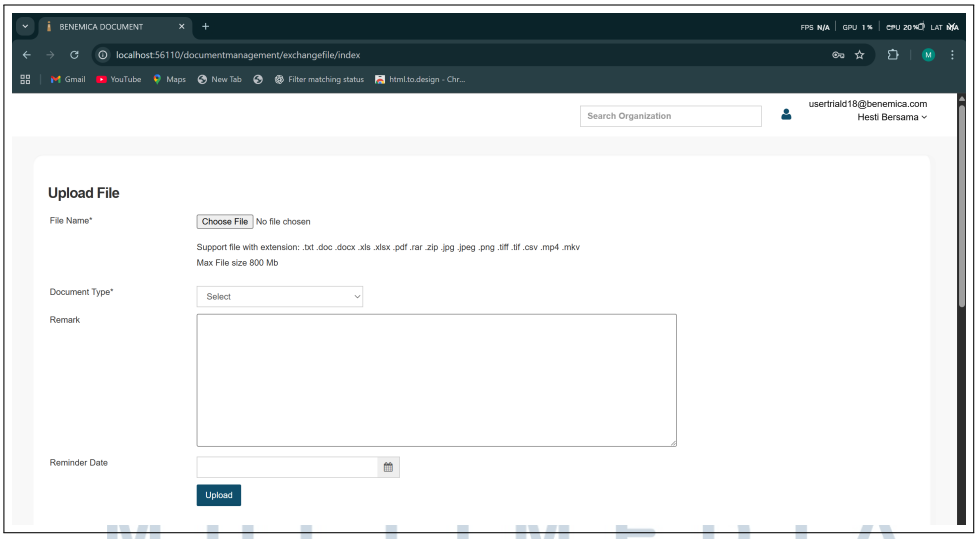
Gambar 3.2. *Flowchart* cara kerja *document management*.

Berikut merupakan tampilan dari *document management*, halaman unggah dan *list* berada di satu halaman yang sama (halaman *list* di parsial halaman unggah).



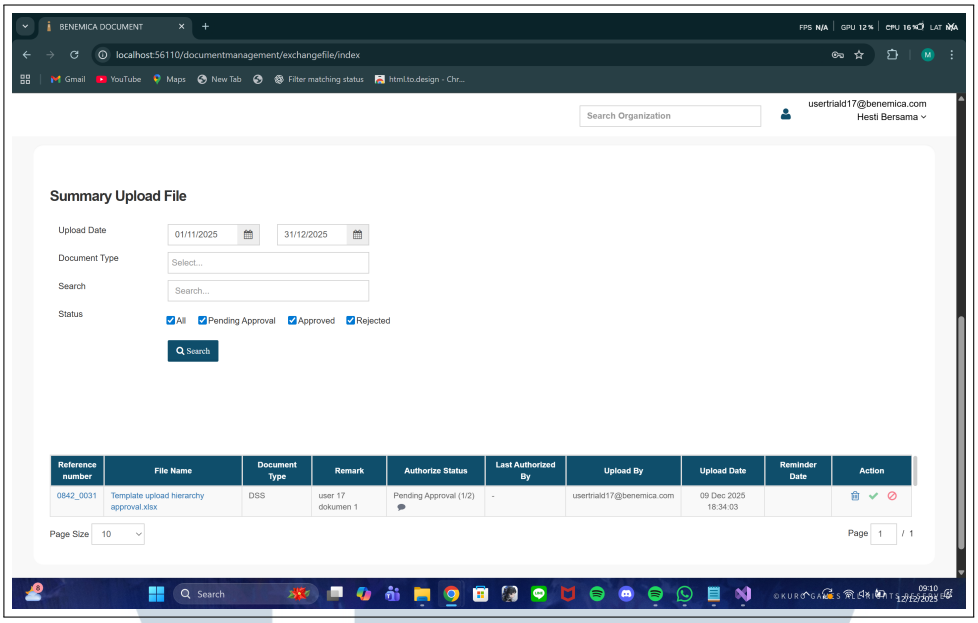
Gambar 3.3. Tampilan *document management*.

Berikut merupakan halaman unggah untuk *document management*. Terdapat field untuk unggah *file* dengan kebutuhan nya, tipe dokumen, remark, dan tanggal pengingat. Dokumen yang di unggah akan otomatis berada di level pertama, dan status menunggu persetujuan.



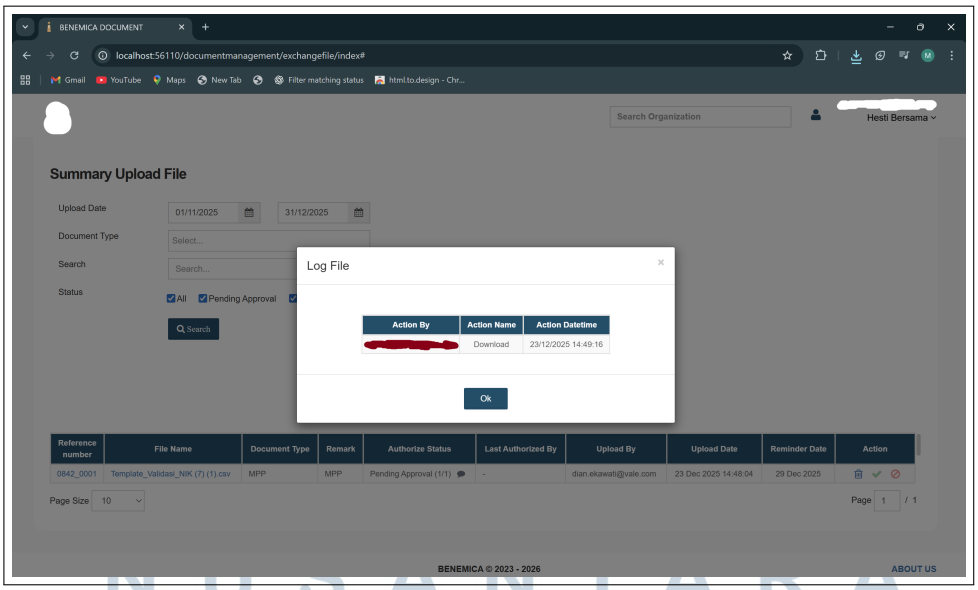
Gambar 3.4. Halaman unggah *document management*.

Berikut merupakan halaman *list* untuk *document management*. Terdapat filter untuk memudahkan pencarian data sesuai dengan data yang difilter dan filter dapat digunakan lebih dari 1 sekaligus. Selain filter juga terdapat *list file* yang telah diunggah, pengguna hanya dapat melihat *file* yang punya aksesnya saja.

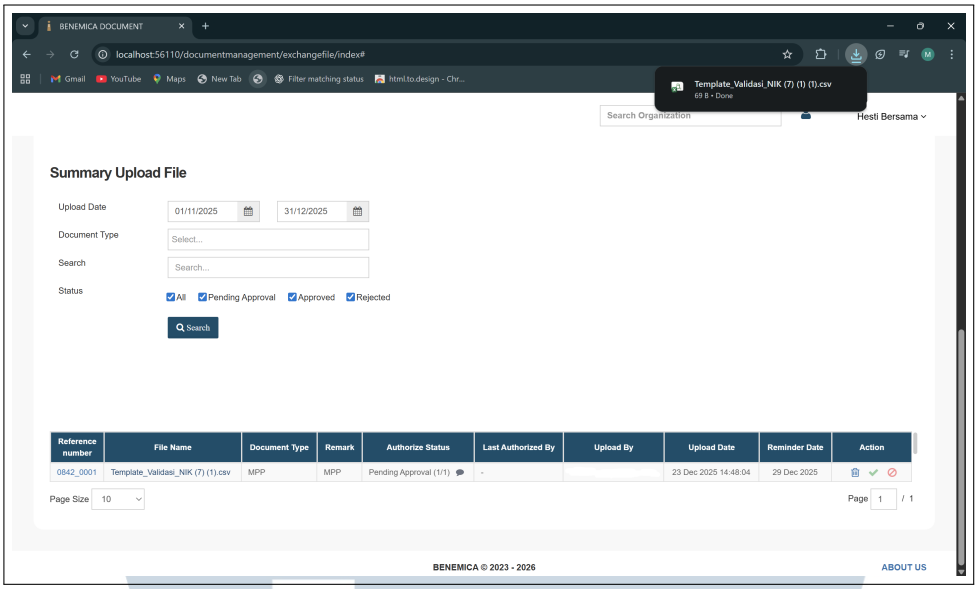


Gambar 3.5. Halaman *list document management*.

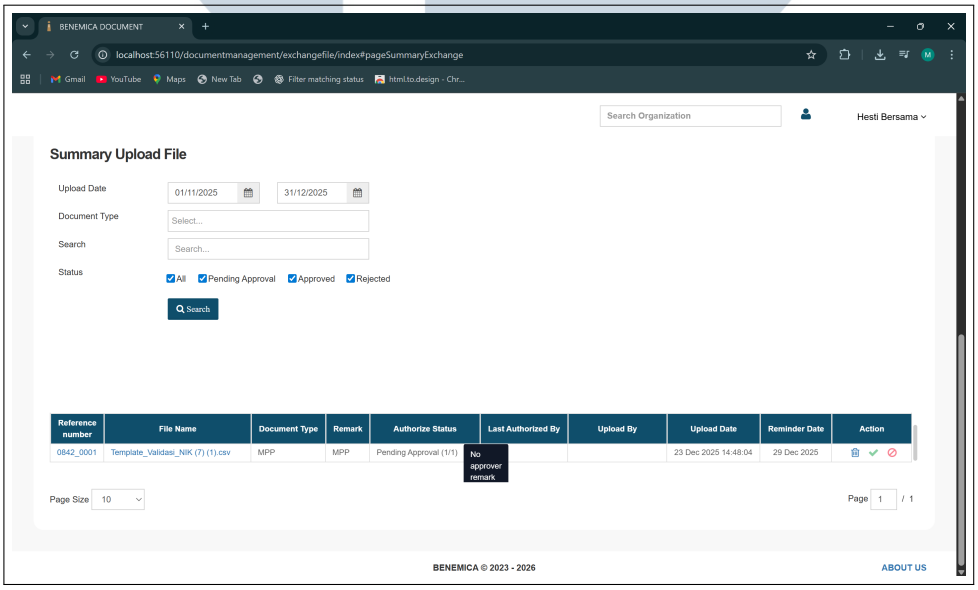
Di *list* data, terdapat beberapa fitur yaitu log dari data tersebut yang berada di kolom nomor referensi yang di klik. Log akan menampilkan *timestamp* dan siapa yang mengunduh, menyetujui, dan menolak dokumen tersebut. Untuk mengunduh *file* tersebut, dapat mengklik kolom *File Name*.



Gambar 3.6. Popup log *document management*.



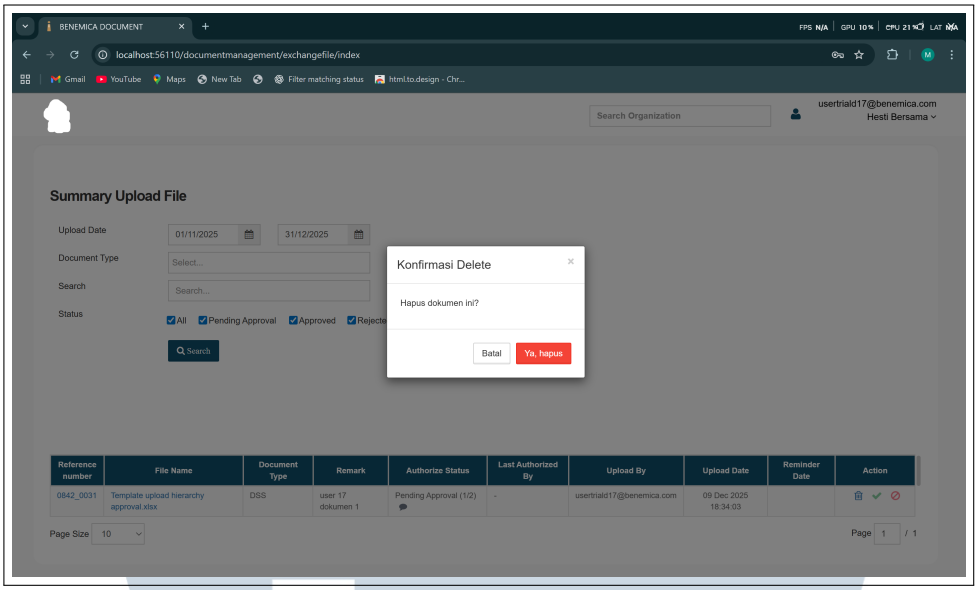
Gambar 3.7. Unduh *file document management*.



Gambar 3.8. Popup remark *document management*.

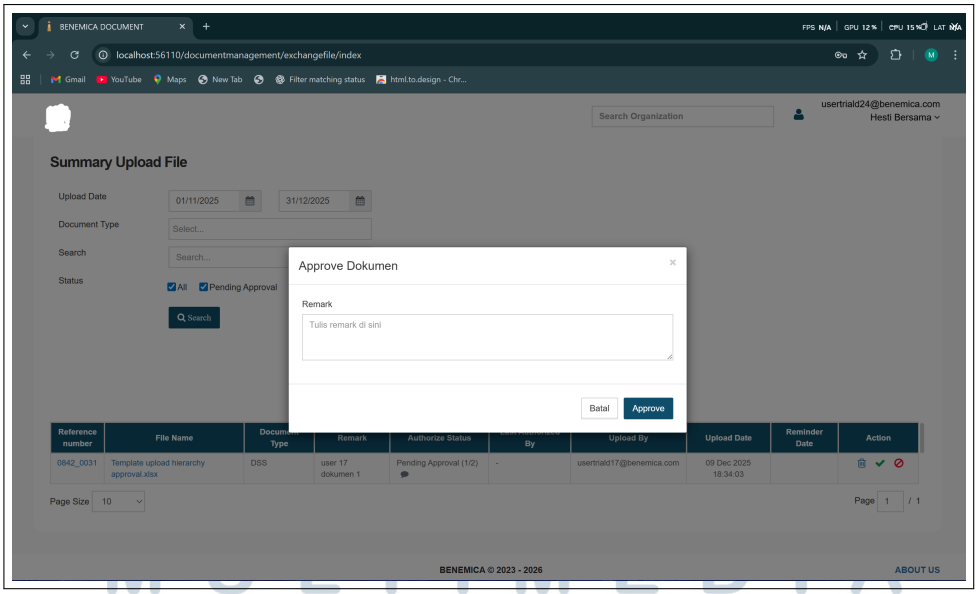
Selain itu untuk memunculkan remark, dapat arahkan kursor atau klik tombol *chat*, di kolom aksi dapat hapus, menyetujui, dan menolak dokumen.

Berikut merupakan popup hapus. Hapus dokumen berfungsi untuk menghapus dokumen yang ada.



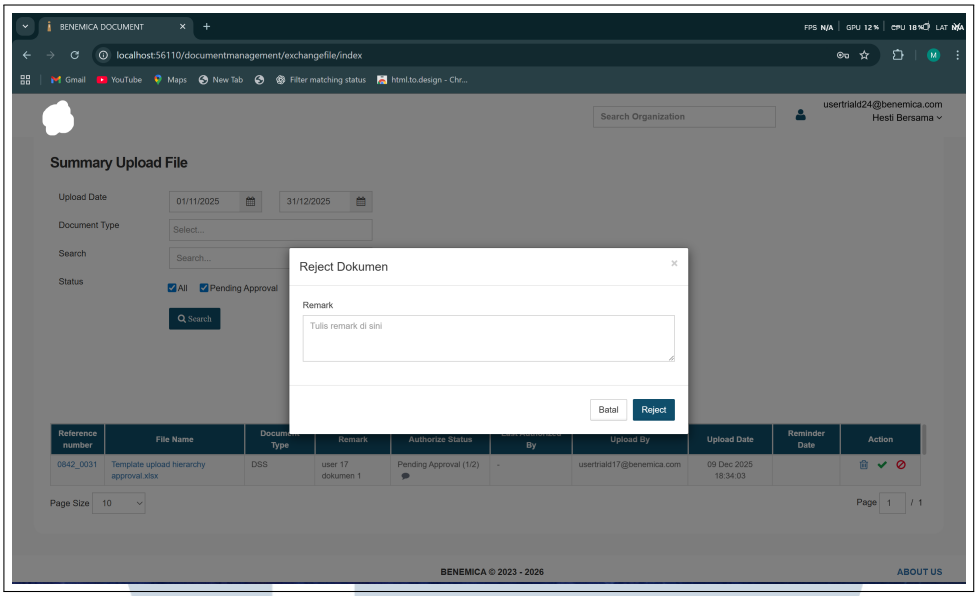
Gambar 3.9. Popup hapus *document management*.

Berikut merupakan popup menyetujui. Menyetujui hanya bisa dilakukan oleh atasan pengunggah data di level sekarang. Menyetujui berfungsi untuk menyetujui dokumen yang di unggah bawahan tersebut.



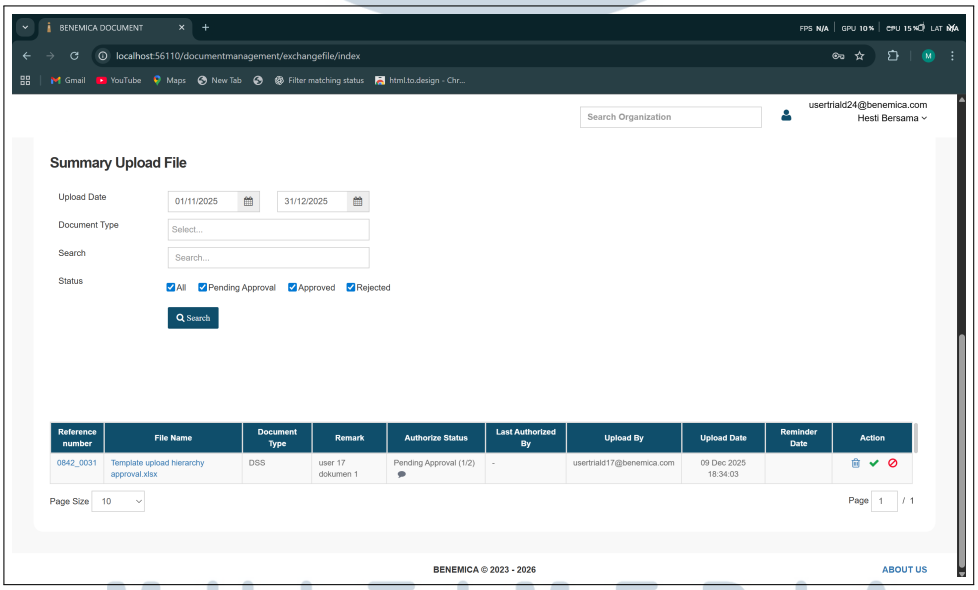
Gambar 3.10. Popup menyetujui *document management*.

Berikut merupakan popup menolak. Menolak hanya bisa dilakukan oleh atasan pengunggah data di level sekarang. Menolak berfungsi untuk menolak dokumen yang diunggah bawahan tersebut.



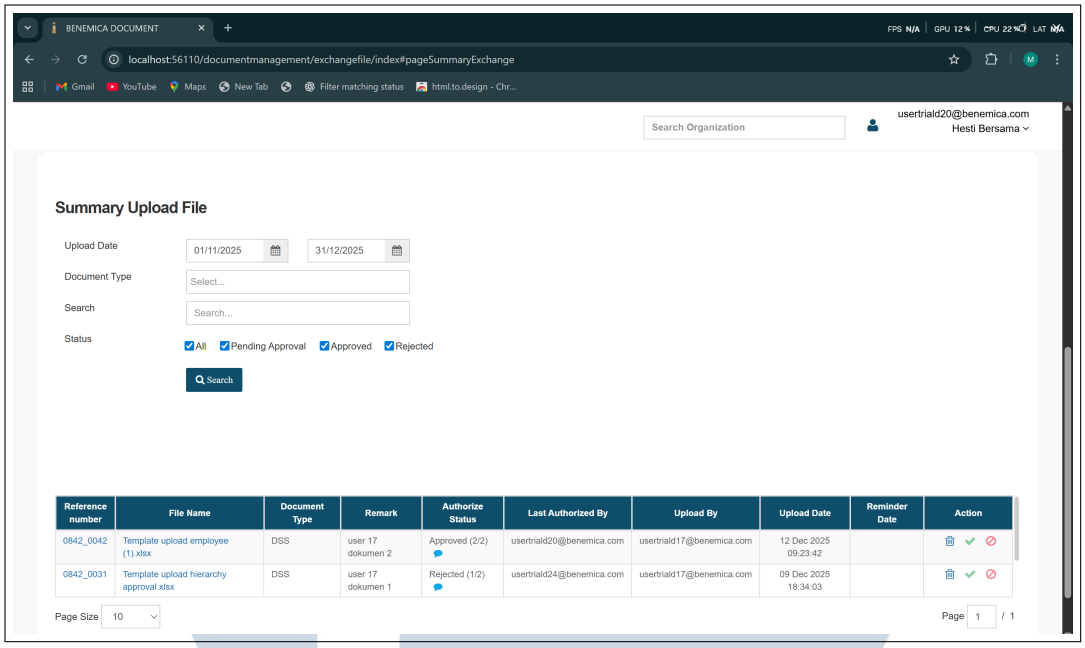
Gambar 3.11. Popup menolak *document management*.

Jika dokumen sudah di menyetujui, tapi belum di level terakhir, maka status akan tetap menunggu persetujuan seperti pada gambar dibawah ini.



Gambar 3.12. Tampilan saat menyetujui tetapi belum di level tertinggi.

Jika dokumen sudah di menyetujui di level tertinggi, maka status akan berubah menjadi disetujui seperti gambar di bawah ini. Tetapi jika dokumen di menolak oleh atasan, akan berubah menjadi ditolak mau di level berapa saja.



Gambar 3.13. Tampilan saat menyetujui di level tertinggi dan menolak.

3.3.3 *Validasi NIK Web App*

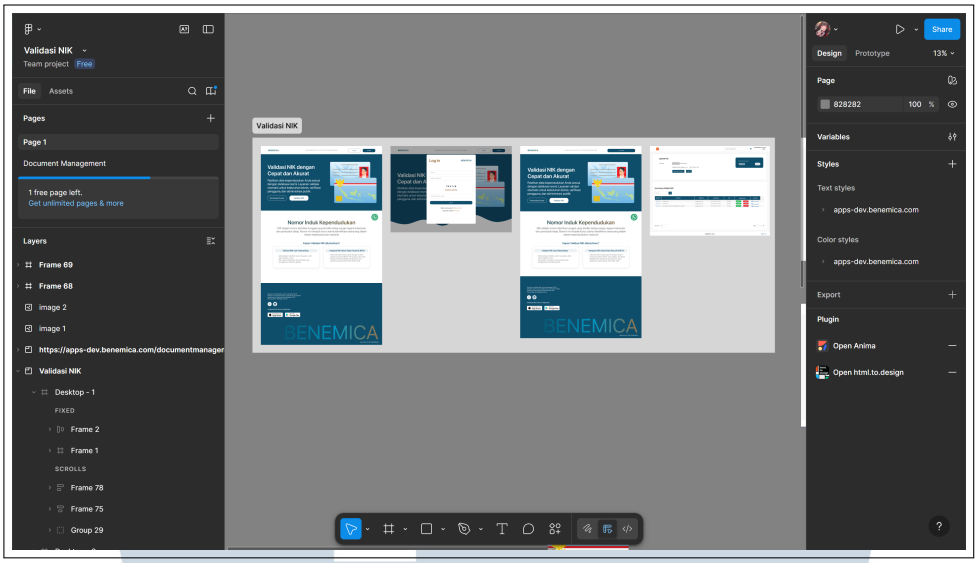
Aplikasi *Validasi NIK* merupakan aplikasi berbasis *web* yang digunakan untuk membantu proses verifikasi data identitas sebagai pemeriksaan awal (*pre-validation*) sebelum data digunakan pada proses administrasi perusahaan. Verifikasi ini bertujuan untuk meningkatkan kualitas data, mengurangi risiko penggunaan identitas tidak valid, serta menekan potensi penyalahgunaan data yang dapat mengarah pada praktik penipuan (*fraud*) seperti pembuatan data karyawan fiktif atau pengajuan yang menggunakan identitas yang tidak sesuai. Proses validasi menggunakan mekanisme *kuota*, yaitu sistem akan melakukan pemotongan sebesar satu kuota setiap kali pengguna melakukan validasi satu NIK, sehingga penggunaan validasi dapat dikendalikan dan dipantau. Validasi dilakukan dengan cara mengunggah *file* berformat *CSV* yang berisi nomor referensi dan NIK dalam satu *file*. Setelah proses validasi selesai, hasil validasi dapat ditinjau dan diunduh melalui *list* unggah yang menampilkan riwayat *upload* pengguna, termasuk status proses dan ringkasan hasilnya.

Aplikasi ini dikembangkan menggunakan *.NET Framework*. Halaman unggah serta keamanan *login* mengikuti *template* standar dari perusahaan untuk menjaga konsistensi antarmuka dan standar keamanan. Sementara itu, tampilan *login* dan halaman *landing page* dikembangkan oleh peserta magang dari awal

hingga selesai sebagai bagian dari penyusunan antarmuka yang sesuai kebutuhan pengguna.

Pada sisi *back-end*, aplikasi menggunakan bahasa *C#* untuk mengelola alur validasi, pengurangan kuota, serta pengolahan dan penyimpanan hasil validasi agar dapat ditelusuri kembali (*audit trail*). Mekanisme pengurangan kuota diterapkan karena proses validasi memanfaatkan layanan verifikasi resmi yang bersifat berbayar, sehingga setiap permintaan validasi perlu dihitung sebagai satu pemakaian layanan. Dengan adanya kuota, penggunaan validasi dapat dikelola secara terukur, mendukung pengendalian biaya operasional, serta memudahkan pencatatan dan pelaporan pemakaian per pengguna. Pada sisi *front-end*, aplikasi menggunakan *cshtml* untuk tampilan, serta *JavaScript* untuk kebutuhan antarmuka dan validasi input secara terbatas (misalnya pemeriksaan format unggahan dan kelengkapan kolom).

Perancangan *front-end Validasi NIK Web App* dilakukan menggunakan *Figma* dengan menyusun *layout* untuk halaman *landing page*, tampilan *login*, serta halaman validasi yang memuat fitur unggah *CSV* dan ringkasan hasil validasi. Setelah struktur halaman terbentuk, peserta magang menerapkan fitur *Auto Layout* pada komponen seperti *card*, form, dan area tombol agar jarak antar elemen konsisten dan memudahkan penyesuaian ukuran tampilan. Selanjutnya, komponen pendukung seperti tombol, *input field*, tabel, dan indikator status dirancang sebagai acuan implementasi, termasuk penentuan posisi serta peran tombol pada tiap halaman (misalnya akses *login*, pembelian kuota, unggah, dan unduh hasil). Pada tahap akhir, peserta magang menambahkan *resource* gambar seperti ilustrasi dan ikon, kemudian menyesuaikan ukuran dan penempatannya agar tampilan selaras dengan identitas visual perusahaan.



Gambar 3.14. Aset figma Validasi NIK.

Berikut ini merupakan *source code* dari pustaka yang digunakan dalam *controller* serta *setup* halaman untuk mengambil data pengguna yang sedang *login* dan melakukan deklarasi variabel yang dibutuhkan.

```
1
2     #region login
3     public class ValidasiNIKController : BaseController
4     {
5         //GET data user login
6         //GET error message
7
8
9         int SUCCESS = 0;
10        int FAIL = 0;
11        int KUOTA = 0;
12        public static string success = "[001_BENEMICA_VALID]";
13        public static string not_valid = "[001
    _BENEMICA_VALIDATION_TAXPAYER_INVALID != 16]";
14        public static string success_system = "[002_BENEMICA_VALID
    ]";
15        public static string fail_system = "[002_BENEMICA_INVALID]
    ";
16        public static string failure_system = "[002
    _BENEMICA_VALIDATION_NIK_FAILURE]";
17        string[] legendLines =
18        {
19            success + ",berhasil tervalidasi",
```

```
            not_valid + ",gagal tervalidasi ( bukan 16 character )",
            success_system + ",berhasil tervalidasi",
            fail_system + ",gagal tervalidasi",
            failure_system + ",gagal tervalidasi"
        };

        public NIKValidationController()
        {
            //GET data user login
        }

        #endregion
    }
```

Kode 3.7: Source code fungsi template download Validasi NIK

Berikut ini merupakan kode dari unduh templat. Unduh templat berfungsi untuk memberitahu pengguna tentang template untuk unggah berkas NIK yang akan di validasi. /

```

#region Template
public ActionResult Template()
{
    var csv = "Nomer,NIK\r\n" +
              "1,[isi dengan 16 nomor nik, dan pastikan kolom a bertipe text jika dibuka di excel]\r\n" +
              "2,[isi dengan 16 nomor nik, dan pastikan kolom a bertipe text jika dibuka di excel]\r\n";
    var bytes = new System.Text.UTF8Encoding(true).GetBytes(csv);
    return File(bytes, "text/csv", "Template_Validasi_NIK.csv");
}
#endregion
```

Kode 3.8: Source code download template upload Validasi NIK

Berikut ini merupakan kode untuk fungsi unggah. Sebelum di masukan ke basis data dan di validasi, sistem akan mengecek apakah kuota cukup untuk dilakukan pemotongan, jika kuota cukup maka akan lanjut ke proses penyimpanan ke basis data dan validasi NIK.

```

#region Upload
public ActionResult Upload(HttpPostedFileBase file)
{
    //Validasi File dan Kuota
```

```
6
7       // Save Ke database
8
9       // Validasi NIK
10  }
11  #endregion
```

Kode 3.9: Source code upload Validasi NIK

Berikut ini merupakan fungsi untuk membaca *file* yang di unggah dan validasi NIK. Private void ValidateNIK berfungsi untuk membaca *file* dan mengambil baris yang berisi NIK, kemudian masuk ke method Validasi. private static char Delimiter berfungsi untuk membaca pemisah dari input pengguna supaya data yang di unggah aman untuk di validasi.

private void Validasi berfungsi untuk mem validasi NIK dengan membandingkan NIK dari sumber yang resmi secara langsung, sehingga hasi validasi valid. method Validasi di panggil di Method ValidateNIK setelah mengambil baris yang berisi NIK.

```
1
2   #region Validasi NIK
3   private void ValidateNIK(Stream fileStream, Guid headerId, string
        createdBy)
4   {
5           // Fungsi Read File
6           {
7
8               // Baca row file yang diambil
9
10              Validasi(item)
11
12              // Proses pengurangan kuota
13          }
14  }
15
16  private static char Delimiter(Stream s)
17  {
18      if (!s.CanSeek) return ',';
19      long pos = s.Position;
20      string first;
21      using (var sr = new StreamReader(s, Encoding.UTF8, true, 1024,
        true))
22          first = sr.ReadLine() ?? "";
23      s.Position = pos;
```

```
24
25      int cComma = 0, cSemi = 0, cTab = 0;
26      foreach (var ch in first)
27      {
28          if (ch == ',') cComma++;
29          else if (ch == ';') cSemi++;
30          else if (ch == '\t') cTab++;
31      }
32
33      if (cSemi > cComma && cSemi >= cTab) return ';';
34      if (cTab > cComma && cTab >= cSemi) return '\t';
35      return ',';
36  }
37
38  private void Validasi()
39  {
40      // Proses Validasi
41  }
42  #endregion
```

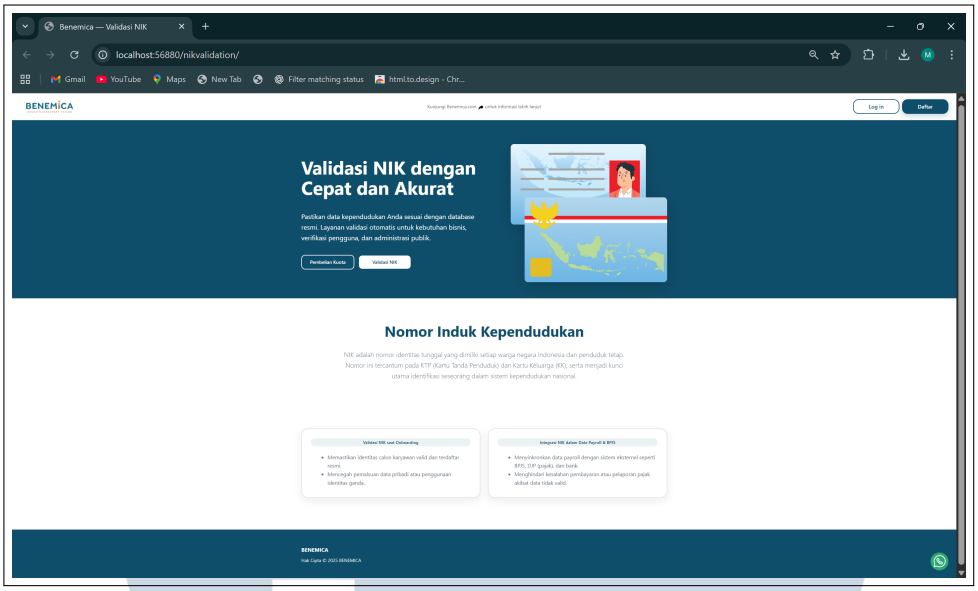
Kode 3.10: Source code baca file Validasi NIK

Berikut ini merupakan kode untuk Unduh Hasil. Setelah di validasi, pengguna dapat melihat hasilnya dengan men klik tombol unduh pada baris data. Hasil validasi akan dimunculkan di csv yang di unduh.

```
1
2   #region Quota
3   [HttpPost]
4   public ActionResult Quota(int SelectedQuota, int SelectedPrice)
5   {
6       // Proses topup quota
7   }
8   #endregion
```

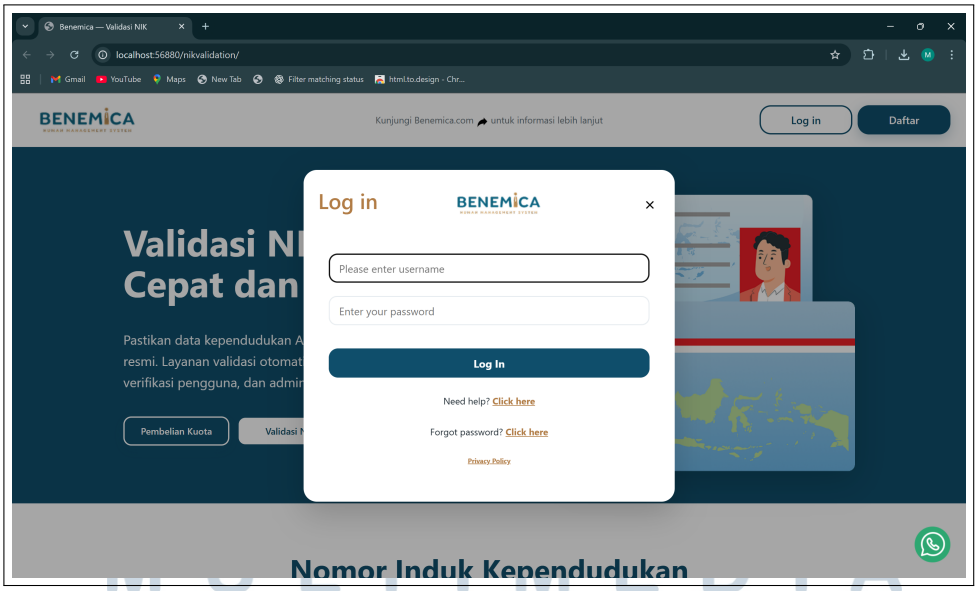
Kode 3.11: Source code Topup quota Validasi NIK

Berikut merupakan tampilan untuk halaman sebelum login, terdapat beberapa informasi mengenai validasi NIK dan penyedia jasa, terdapat juga nomor kontak yang dapat dihubungi jika ada pertanyaan dari pengguna. Jika pengguna belum login, maka button akan mengarahkan pengguna untuk login terlebih dahulu.



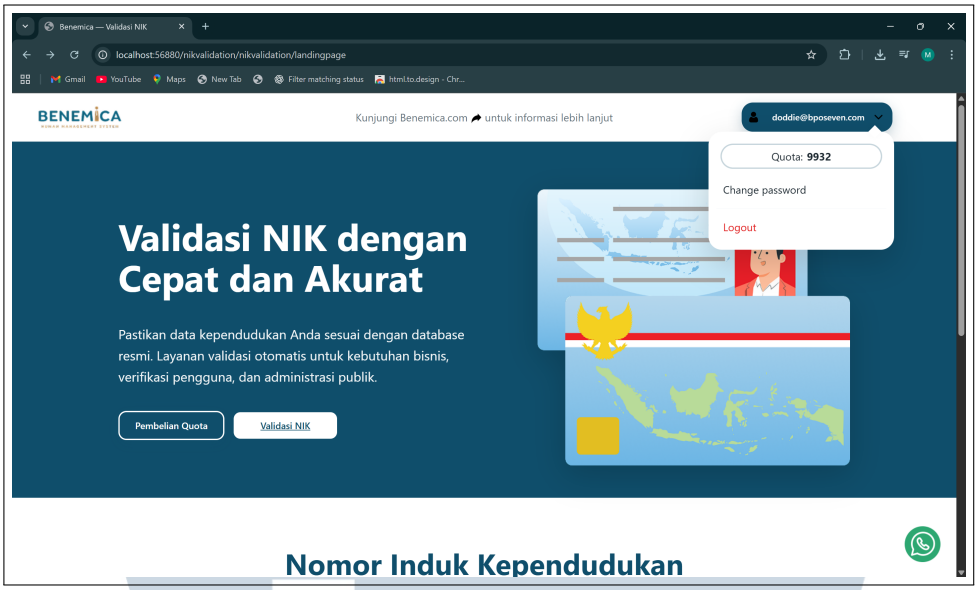
Gambar 3.15. Tampilan halaman sebelum login Validasi NIK.

Berikut merupakan tampilan popup login saat pengguna akan login. Pengguna perlu untuk memasukan pengguna dan sandi untuk dapat login ke dalam aplikasi.



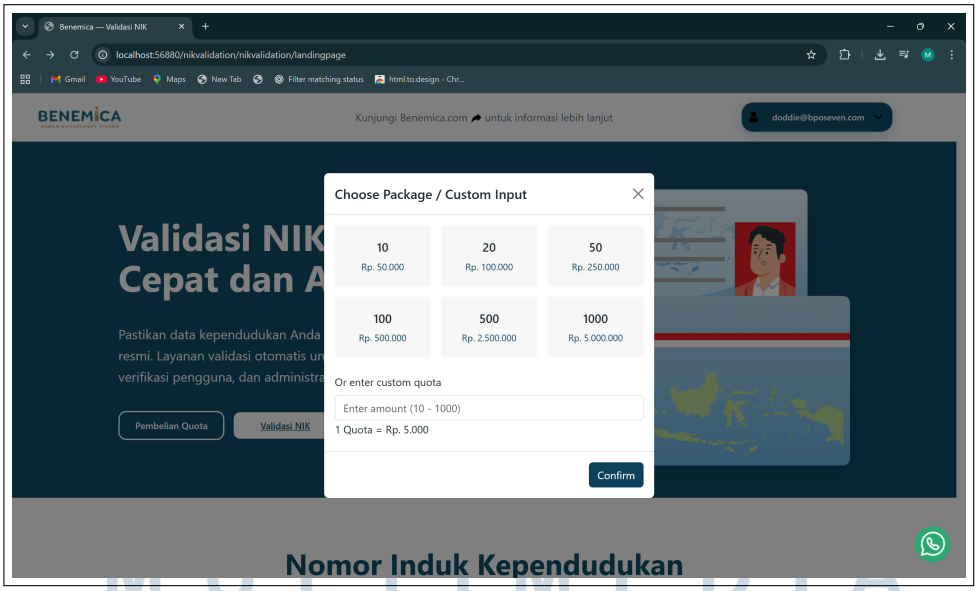
Gambar 3.16. Tampilan popup login Validasi NIK.

Berikut merupakan tampilan Halaman awal setelah pengguna berhasil login. Setelah masuk *landing page*, pengguna baru dapat melakukan proses isi ulang dan validasi NIK.



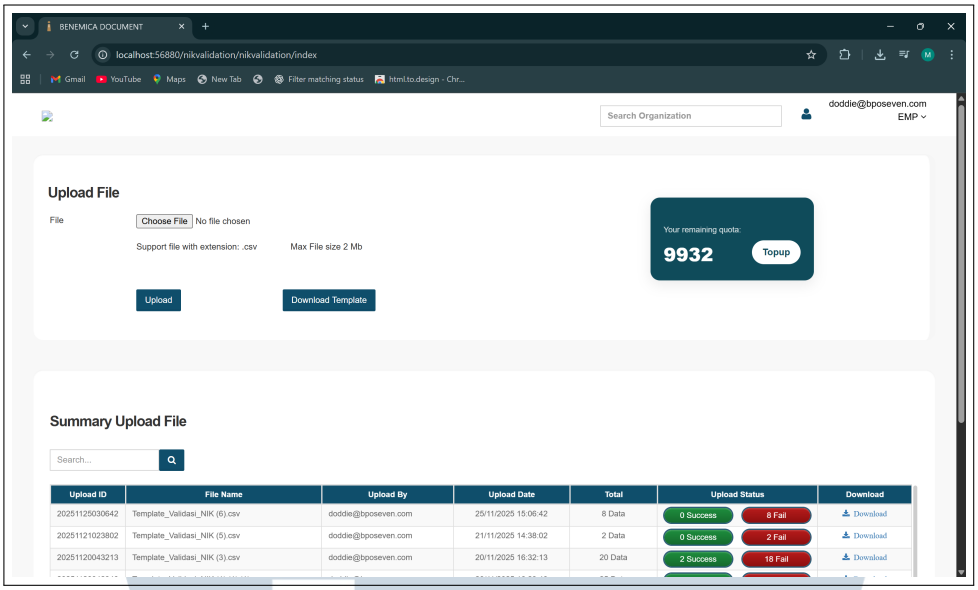
Gambar 3.17. Tampilan landing page Validasi NIK.

Berikut merupakan popup isi ulang yang muncul saat user klik tombol pembelian quota. popup berisi harga paket dan harga satuan yang dapat dipilih user.



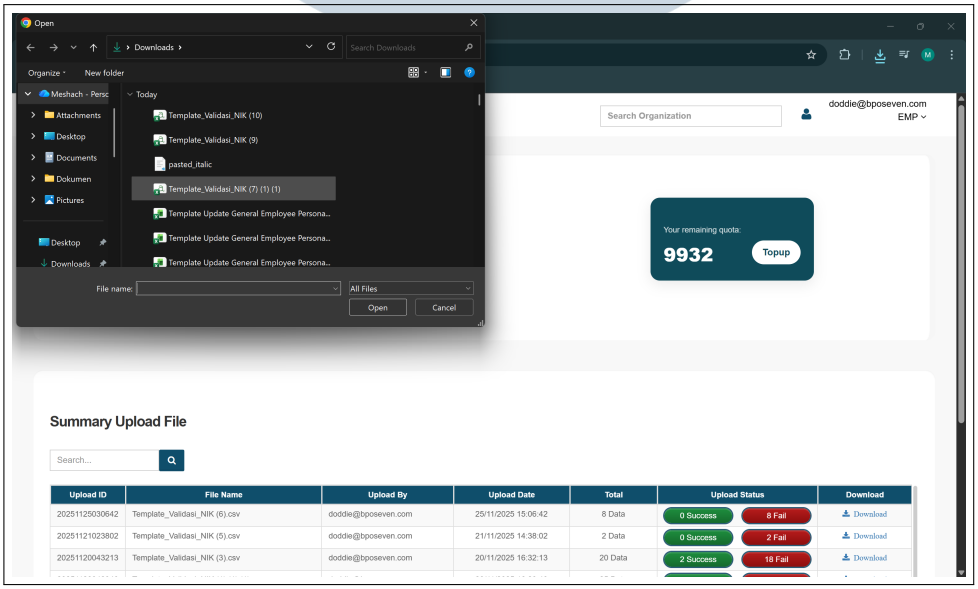
Gambar 3.18. Tampilan popup topup di landing page Validasi NIK.

Berikut merupakan tampilan halaman unggah. Halaman unggah berisi field untuk unggah, sisa kuota pengguna, dan halaman list.



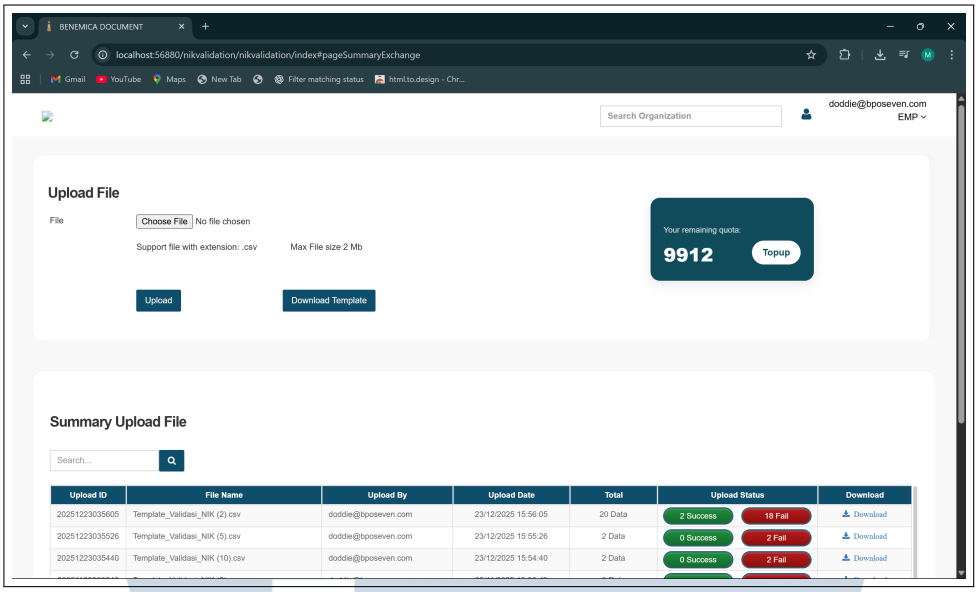
Gambar 3.19. Tampilan halaman upload Validasi NIK.

Berikut merupakan tampilan saat Penjelajah Berkas dibuka untuk memilih *file* yang akan di unggah dan di validasi.



Gambar 3.20. Tampilan popup file explorer.

Berikut merupakan tampilan setelah *file* berhasil di unggah. Akan ada kolom yang menunjukan berapa baris data yang di unggah, dan ada status unggah yang menunjukan berapa baris yang berhasil dan yang gagal. Selain itu juga ada tombol untuk unduh hasil yang sudah divalidasi. Kuota yang ada juga akan dikurangi berdasarkan jumlah data yang di unggah.



Gambar 3.21. Tampilan halaman setelah upload.

3.3.4 Kendala yang Ditemukan

Selama pelaksanaan kerja magang di divisi teknologi, peserta magang menghadapi beberapa kendala baik dari sisi teknis maupun pola kerja. Salah satu kendala utama adalah keterbatasan pemahaman awal terhadap .NET Framework yang digunakan sebagai teknologi utama dalam pengembangan aplikasi perusahaan. Kondisi ini menyebabkan peserta magang memerlukan waktu adaptasi untuk memahami struktur proyek, alur kerja aplikasi, serta konsep pengembangan yang diterapkan.

Selain itu, pola kerja yang lebih banyak dilakukan secara individu menuntut peserta magang untuk menyelesaikan permasalahan secara mandiri, mulai dari analisis kebutuhan hingga penyelesaian teknis, yang pada awalnya menjadi tantangan tersendiri.

Kendala lainnya adalah proses debugging dan pengujian aplikasi yang membutuhkan ketelitian tinggi, terutama karena setiap fitur saling terintegrasi dan kesalahan kecil dapat memengaruhi fungsi sistem secara keseluruhan.

3.3.5 Solusi atas Kendala yang Ditemukan

Untuk mengatasi kendala-kendala tersebut, peserta magang melakukan beberapa upaya penyesuaian dan peningkatan kemampuan. Dalam menghadapi keterbatasan pemahaman terhadap .NET Framework, peserta magang mempelajari

dokumentasi resmi, referensi teknis, melakukan eksplorasi terhadap source code aplikasi yang telah ada, dan bertanya kepada pembimbing hal yang kurang jelas.

Pola kerja individual disikapi dengan meningkatkan kemandirian dalam mencari solusi melalui studi literatur dan sumber daring, serta tetap menjaga koordinasi secara berkala dengan pembimbing lapangan untuk memperoleh arahan dan masukan.

Sementara itu, untuk mengatasi kendala dalam proses debugging dan pengujian, peserta magang menerapkan pengujian secara bertahap pada setiap fitur, memanfaatkan tools debugging yang tersedia, serta mendokumentasikan hasil pengujian sebagai bahan evaluasi.

Dengan penerapan langkah-langkah tersebut, kendala yang dihadapi dapat diminimalkan dan proses pengembangan aplikasi dapat berjalan dengan lebih efektif.