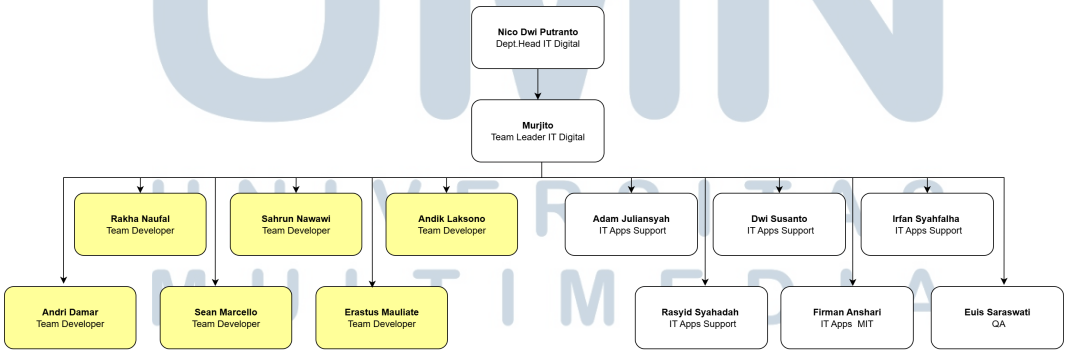
# BAB 3
# PELAKSANAAN KERJA MAGANG

## 3.1 Kedudukan dan Koordinasi

Selama pelaksanaan program magang di PT Astra Otoparts Tbk, posisi yang diberikan berada dalam lingkup pengembangan sistem pada peran sebagai Web Developer Intern. Kegiatan magang dilaksanakan di bawah pengawasan langsung dari Bapak Nico Dwi Putranto selaku supervisor, yang memberikan bimbingan teknis dan melakukan evaluasi secara berkala terhadap progres pekerjaan yang dilakukan.

Keterlibatan dalam tim pengembang pada divisi Business to Business berfokus pada proyek integrasi sistem pembayaran digital untuk platform Astraotoshop. Tim terdiri atas beberapa anggota dengan tanggung jawab berbeda pada pengembangan fitur-fitur dari berbagai platform, namun tetap menjalankan kerja kolaboratif untuk mendukung keberhasilan proyek secara keseluruhan. Komunikasi dan koordinasi yang intensif antar anggota tim menjadi faktor kunci dalam penyelesaian setiap tahapan pengembangan, mulai dari perancangan antarmuka pengguna, implementasi sistem transaksi B2B, hingga pengembangan fitur verifikasi OTP pada proses login.

Struktur organisasi perusahaan, khususnya pada IT Digital Department divisi Business to Business dalam lingkup tim developer Astra Otoparts, ditampilkan pada Gambar 3.1.




Gambar 3.1. Struktur divisi B2B IT Digital

## 3.2 Tugas yang Dilakukan

Selama masa magang di PT Astra Otoparts Tbk, kontribusi diberikan dalam proyek pengembangan sistem pembayaran pada platform web desktop Astraotoshop. Fokus utama proyek ini adalah integrasi metode pembayaran digital Astrapay dan Doku untuk menunjang proses transaksi B2B (Business to Business). Selain pengembangan sistem backend dan frontend, perhatian juga diarahkan pada keandalan proses autentikasi dan keamanan pengguna melalui implementasi sistem OTP (One-Time Password) saat login.

Kegiatan yang dilakukan mencakup beberapa tahapan penting dalam pengembangan perangkat lunak, mulai dari analisis kebutuhan sistem, perancangan alur logika, implementasi kode menggunakan bahasa pemrograman C#.NET dan JavaScript, hingga pengujian dan dokumentasi teknis. Selama proses pengembangan, juga dilakukan kolaborasi aktif dalam tim untuk menyusun skema integrasi API serta menyesuaikan struktur data yang digunakan.

- Melakukan integrasi API pembayaran digital Astrapay dan Doku ke dalam sistem transaksi B2B Astraotoshop.
- Mengembangkan fitur verifikasi OTP (One-Time Password) pada proses login pengguna sebagai lapisan keamanan tambahan.
- Mengimplementasikan logika pemrosesan transaksi dan penyesuaian antarmuka pengguna terkait metode pembayaran pada platform web desktop menggunakan C#.NET.
- Melakukan pengujian dan debugging terhadap sistem pembayaran untuk memastikan fungsionalitas berjalan sesuai alur yang dirancang.

## 3.3 Uraian Pelaksanaan Magang

Periode magang dimulai pada 6 Januari 2025 di PT Astra Otoparts Tbk dengan penugasan utama dalam proyek pengembangan sistem pembayaran pada platform web desktop Astraotoshop. Pelaksanaan kegiatan difokuskan pada integrasi metode pembayaran digital Astrapay dan Doku melalui pengembangan API baru, serta penambahan sistem OTP (One-Time Password) pada proses login sebagai peningkatan keamanan pengguna.

Selama pelaksanaan proyek, dilakukan koordinasi aktif dengan tim developer untuk memastikan setiap komponen sistem berjalan sesuai perencanaan dan standar teknis yang telah ditentukan. Aktivitas pengembangan mencakup proses analisis kebutuhan, implementasi logika sistem menggunakan bahasa pemrograman C#.NET, hingga pengujian fitur untuk menjamin fungsionalitas dan stabilitas aplikasi. Seluruh tahapan mengikuti timeline yang ditetapkan dan mengacu pada prosedur kerja internal perusahaan, guna mendukung kelancaran operasional digital Astraotoshop dalam lingkungan bisnis B2B.

Pelaksanaan kegiatan magang diuraikan seperti pada Tabel 3.1.

Tabel 3.1. Rincian Pelaksanaan Kerja Magang Mingguan

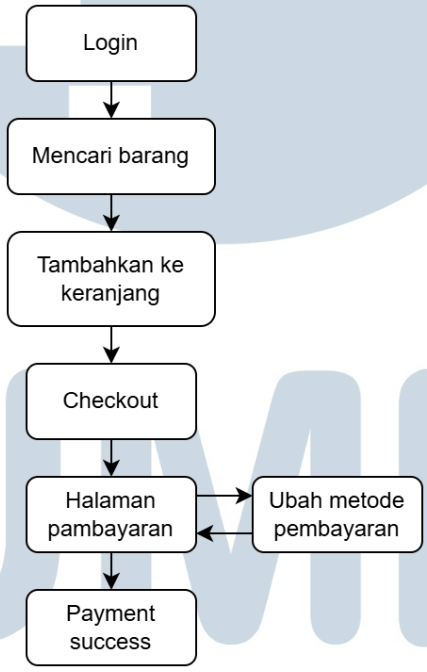
| **Minggu Ke -** | **Pekerjaan yang dilakukan** |
|---|---|
| 1 | Mengikuti meeting awal dengan vendor untuk menerima transfer knowledge terkait perpindahan proyek pengembangan ke tim internal Astra Otoparts. |
| 2 | Melanjutkan diskusi teknis dengan vendor terkait alur sistem dan dokumentasi awal proyek. |
| 3 | Mulai mempelajari bahasa pemrograman C# untuk mendukung proses pengembangan sistem internal. |
| 4 | Memperdalam pemahaman terhadap sintaks dan logika C#, serta melakukan studi kasus mandiri. |
| 5 | Mempelajari framework ASP.NET sebagai fondasi pengembangan web desktop Astraotoshop. |
| 6 | Membuat draft awal user manual sistem Business to Business (B2B) Astraotoshop. |
| 7 | Menyelesaikan dan menyusun dokumentasi user manual untuk kebutuhan internal pengguna bisnis. |
| 8 | Membantu proses testing fitur-fitur pada aplikasi Astraotoshop, serta mencatat hasil pengujian. |
| 9 | Melanjutkan proses testing serta mengikuti meeting pelatihan implementasi sistem B2B pada platform MDS. |
| 10 | Melakukan instalasi beberapa perangkat lunak pendukung proyek seperti Visual Studio 2022, Git, Postman, dan SQL Server Management Studio (SSMS) sebagai persiapan pengembangan dan pengujian aplikasi. |

| Minggu Ke - | Pekerjaan yang dilakukan |
|---|---|
| 11 | Melanjutkan proses instalasi dan konfigurasi tools yang diperlukan, serta mulai mengeksplorasi struktur direktori dan dependensi awal pada proyek dengan membuka solusi `B2B R2.sln` di Visual Studio. |
| 12 | Melakukan penelusuran terhadap komponen dan dependensi internal dari solusi `B2B R2.sln`, serta memahami alur kerja dasar dari aplikasi tersebut. |
| 13 | Libur Hari Raya Idul Fitri (tidak ada kegiatan magang pada minggu ini). |
| 14 | Melanjutkan pemahaman terhadap source code proyek dengan fokus pada solusi `B2B.Web.Desktop.sln`, mencakup struktur proyek, relasi antar modul, serta konfigurasi awal pada sisi front-end dan back-end. |
| 15 | Mempelajari lebih lanjut struktur dan alur kerja dari solusi `B2B R2.sln` dan `B2B.Web.Desktop.sln`, dengan fokus pada integrasi antar modul yang diterapkan dalam aplikasi. |
| 16 | Melanjutkan proses pemahaman terhadap source code proyek secara menyeluruh, mencakup bagaimana data diproses dari sisi front-end hingga back-end dalam kedua solusi tersebut. |
| 17 | Menganalisis struktur lama integrasi pembayaran di web desktop Astraotoshop dan menyiapkan struktur baru menggunakan API terkini. |
| 18 | Mengimplementasikan integrasi API baru pada halaman keranjang sistem pembayaran B2B. |
| 19 | Melanjutkan integrasi API baru pada halaman ubah metode pembayaran (change payment). |
| 20 | Menyelesaikan proses integrasi API baru di halaman pembayaran (payment) dan memastikan fungsi berjalan dengan baik. |
| 21 | Mendiskusikan rancangan logika OTP (One-Time Password) bersama tim serta merancang tampilan UI login. |
| 22 | Membuat dan menghubungkan API OTP untuk proses autentikasi pengguna pada halaman login. |
| 23 | Menyusun dan mengatur struktur database untuk menyimpan data OTP serta mengatur validasi waktu berlaku OTP. |

| Minggu Ke - | Pekerjaan yang dilakukan |
|---|---|
| 24 | Melakukan debugging dan finalisasi fitur OTP agar dapat berjalan dengan lancar sesuai skenario penggunaan. |

## 3.4 Perancangan Flowchart

Sebelum CR (Change Request) dilaksanakan, dilakukan proses perancangan sistem guna memastikan alur kerja dan struktur sistem sesuai dengan kebutuhan pengguna pada platform web Astraotoshop. Salah satu elemen penting dalam tahap ini adalah perancangan flowchart, yang digunakan untuk menggambarkan alur navigasi pengguna dari satu halaman ke halaman lainnya dalam proses transaksi B2B, khususnya pada fitur pembayaran.




Gambar 3.2. Flowchart pembelian di AstraOtoshop

Flowchart dirancang secara sistematis untuk memetakan urutan interaksi pengguna, seperti proses login, pemilihan metode pembayaran, hingga konfirmasi transaksi. Dengan adanya flowchart ini, tim pengembang dapat memahami dan menyusun struktur sistem secara lebih efisien serta mengurangi potensi kesalahan dalam implementasi.

### 3.4.1 Tugas 1: Implementasi API Pembayaran Baru pada Web Desktop Astraotoshop

Fokus utama kegiatan dalam proyek pengembangan Astraotoshop B2B Web Desktop adalah integrasi metode pembayaran baru menggunakan API dari penyedia layanan pembayaran eksternal. Proses dimulai dari halaman *Entry Order* (keranjang), dengan melakukan penyesuaian pada bagian pemilihan metode pembayaran. Integrasi dilakukan agar sistem dapat menampilkan opsi pembayaran yang bersumber dari API baru secara real-time.

Pada halaman *Payment*, sistem dimodifikasi untuk menampilkan halaman pembayaran dari pihak ketiga melalui *iframe*, sesuai dengan metode pembayaran yang dipilih sebelumnya. Tampilan halaman juga disesuaikan agar tetap responsif dan konsisten dengan gaya visual Astraotoshop.

Halaman *Change Payment Method* juga mengalami pembaruan, terutama pada komponen dropdown yang kini menampilkan data metode pembayaran dari API baru. Seluruh perubahan ini ditujukan untuk meningkatkan akurasi, kecepatan, dan kenyamanan proses transaksi bagi pengguna B2B Astraotoshop.

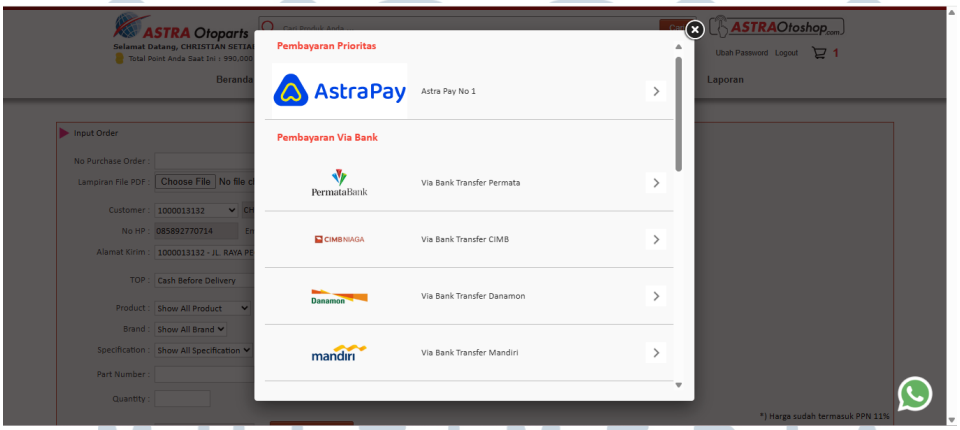
Setiap tahapan implementasi dilakukan dengan mempertimbangkan struktur alur kerja yang sudah ada, serta disesuaikan dengan standar integrasi yang ditetapkan. Uji coba dilakukan secara berkala untuk memastikan bahwa integrasi berjalan stabil tanpa mengganggu sistem eksisting.

```
let tdELement = document.getElementById("
    tdGrandTotalAfterVoucherOrPoint");
console. Log(tdELement);
let totalText = tdELement.innerText. trim(); // Contoh: "676.448"
let totalCLeaned = totalText.replace(/[. ,]/g, ''); // Hapus semua
    titik + "676448"
let totalValue = parseInt(totalCleaned, 10); // Ubah ke integer
console. Log("Cleaned total:", totalValue);

ApiService.get("payment/payment-methods/Web.IO/" + totalValue)
    .then(function (response) {
        if (response.statusCode === 200) {
            let data = response.data;
            let sb = 'stable class="FieldText" cellpadding="0"
    cellspacing="0" width="100%">';
            data. forEach(function (group) {
                // Judul grup pembayaran
                sb += '<tr height="50px">
std></td>
std class="FieldTextPress" colspan="4" style="font-size:12pt;">&
    nbsp; <b>${group.paymentMethodGroupNane}</b></td>
</tr>';

                // List channel/metode pembayaran dalan grup
    tersebut
```

```
                group.paymentChannels. forEach(function (iten) {
                    sb += '<tr style="height:50px; border-bottom:
    solid 1px silver; cursor:pointer;" onclick="Grid_ItemConnand("'
                    sb += '<td width="10px">  ;< /td>';
                    sb += '<td align="center" width="80px"><img
    src="${item.imageURL}" alt="" style="height :80px;" /></td>';
                    sb += '<td width="20px">  ;< /td>';
                    if (iten.isALlowed) {
                        sb += '<td>${item.paymentName}</td>';
                    } else {
                        sb += '<td><table cellpadding="0"
    cellspacing="0" width="100%" style="margin-top: 10px; margin-
    botton:10px
str><td>${iten.paymentName}</td></tr>
str>std class="FieldTextPress" style="font-style:italic;">${item.
    unavailableDescription || ""}</td></tr>
</table></td>';
                    }

                    sb += 'std align="center" width="50px"><img
    src="/Asset/Image/Icon/IconRightArrow.png" alt="" style="height
     :3';
                    sb += '</tr>';
                });
            });
        }
    });
```

Kode 3.1: Kode Payment Channel

Kode 3.1 merupakan potongan kode yang digunakan untuk mengatur tampilan dan urutan metode pembayaran berdasarkan data dari API.



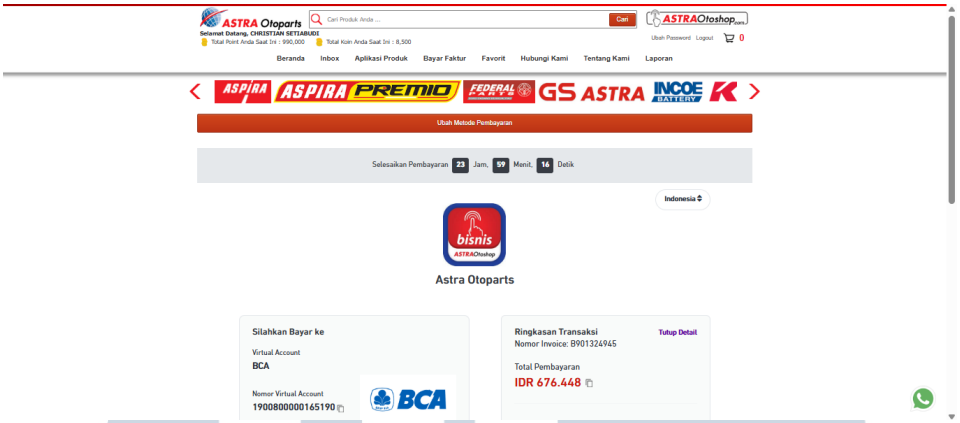

Gambar 3.3. Halaman Entry Order

Gambar 3.3 merupakan tampilan halaman Entry Order (Keranjang) yang menunjukkan integrasi metode pembayaran. Pada halaman ini, pengguna dapat memilih metode pembayaran yang ditampilkan berdasarkan data dari Payment Channel API.

```
function ucPaymentChannel_btnSelectPaymentChanneL_CLick(sender,
    args)
{
ucPaymentChanneL_nShow(this, $('#ddLTOP').val());
}

function LinkDelete_CLick(sender, args)
{
    fShowConfirm('#ucModalConfirm1', 'Information', 'Hapus Item?',
    'Ya', 'Tidak', function()
    {
        var mdl            = new Object();
        mdl.pList_CartID   = fCollectProductChecked();

        fPost2(1, _AppPath + '/UI/eCommerce/eCommerceController.
    asmx/fDeleteCartByList', "ndLCart", mdl, 'fDeleteCartByList');
    });
}

function btnSave_CLick(sender, args)
{
    if (!fIsVaLidC)) { return; }
    fHoldButton($('#btnSave'));
    bIsMobile = false;

    if ("wr" == '< fGetQueryString("mode")')
    {
        if ('C003' == $("#ddLTOP").val() || 'C004' == $("#ddLTOP')
    .val())
        {
        fShowConfirn('#ucModalConfirm1', 'Confirmation', 'Process
    Order?', 'Yes', 'No', function() { btnSave2_CLick(this, null);
        }
        else if ('9999' == $C'#ddLTOP').val())
        {
        var mdLPage                = new Object();
            _arrSKUID              = fCollectProductCheckedC);
            mdLPage.pParameter1    = _strCartSessionID;
            mdLPage.pListParaneter1 = _arrSKUID;
```

Kode 3.2: Kode Entry Order

Kode 3.2 merupakan potongan kode yang mengatur tampilan serta logika aksi saat tombol pada halaman Entry Order diklik.




Gambar 3.4. Halaman Payment

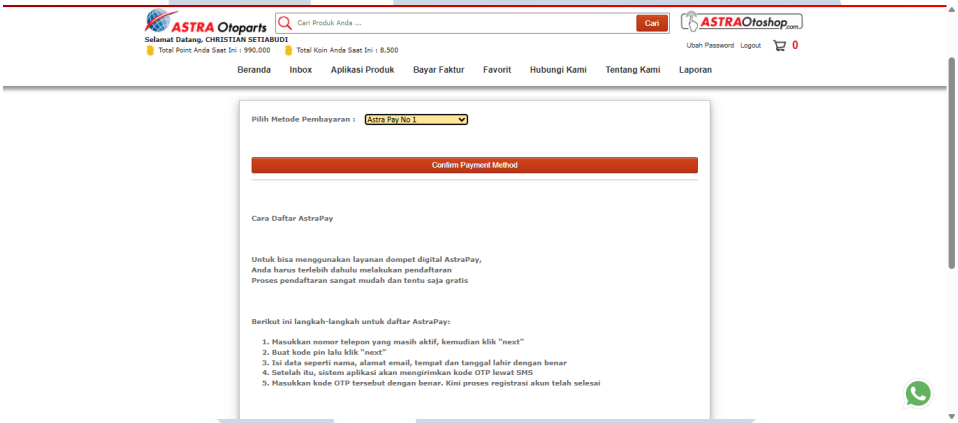
Gambar 3.4 menunjukkan tampilan halaman Payment yang memuat hasil dari API metode pembayaran. Informasi dari API ditampilkan dalam sebuah *iframe*, dan tersedia tombol ”Ubah Metode Pembayaran” untuk memungkinkan pengguna memilih metode pembayaran lain jika diperlukan.

```
var transactionID = fGetQueryString("id")>
ApiService.get("payment/request-payment/" + transactionID)
    .then(function (response) {
        if (response.statusCode === 200 && response.data) {
            if (response.data.isPaid == true) {
                // Redirect ke Home.aspx jika sudah dibayar
                window.Location.href = "/UI/NewOrder/Home.aspx";
                return;
            }
                var provider = response.data.paymentProvider;
                var dokuUrl = response.data.paymentDokuUrL;
                getPaymentDokuUrl = dokuUrl;
            else if (provider == "Doku") {
                $('#iframePayment').attr('src', dokuUrl);
            } else {
                getCmsContent();
            }
        } else {
            console.error("Gagal mendapatkan data pembayaran:",
   response);
        }
    })
    .catch(function (error) {
        console.error("Error saat ambil data pembayaran:", error);
    });
}
function openAstraPayTab() {
    if (getPaymentDokuUrl) {
        window.open(getPaymentDokuUrL, '_blank');
    } else {
        alert("Link pembayaran tidak tersedia.");
    }
}

```

```
34 function getCmsContent() {
```

Kode 3.3: Kode Halaman Payment

Kode 3.3 merupakan potongan kode yang mengatur tampilan halaman Payment. Kode ini mencakup proses pemanggilan API metode pembayaran, penampilan hasil dalam iframe, serta penanganan aksi tombol ”Ubah Metode Pembayaran”.




Gambar 3.5. Halaman Change Payment Method

Gambar 3.5 menunjukkan tampilan halaman Change Payment Method yang digunakan ketika pengguna ingin mengubah metode pembayaran setelah melakukan proses checkout.

```
1  function backHistory() {
2      if (document.referrer) {
3          window. Location = document.referrer;
4      } else {
5          window.history.back();
6      }
7  }
8
9  function btnConfirmPaymentMethodMethod( {
10     var transactionID = '<%=fGetQueryString("id")%>';
11     var selectedPaymentCode = document.querySelector('select[id$="
       ddlPaymentMethod"]').value;
12     if (!selectedPaymentCode) {
13         alert("Silakan pilih metode pembayaran terlebih dahulu.");
14         return;
15     }
16     var payload = {
17         paymentChanneLCode: selectedPaymentCode
18     };
19     ApiService.put("payment/change-payment/" + transactionID,
       payLoad)
20         .then(function (response) {
21             if (response && response.data && response.data.isPaid
       === true) {
22                 // Redirect ke Home.aspx jika sudah dibayar
```

```
                window.Location.href = "/UI/eComnerce/Home.aspx";
                return;
            }
            backHistory();
            console.Log('res: ', response)
        })
        .catch(function (error) {
            console.error("Error saat memanggil API:", error);
            alert("Terjadi kesalahan saat memproses permintaan.");
        });
}
```

Kode 3.4: Kode Halaman Change Payment Method

Kode 3.4 merupakan potongan kode yang mengatur halaman Change Payment Method. Kode ini menampilkan dropdown list berisi metode pembayaran dari API khusus untuk perubahan pembayaran, serta mengatur logika tampilan berbeda jika pengguna memilih metode Astrapay di mana akan ditampilkan CMS dibandingkan metode pembayaran lain yang tidak menampilkan CMS.

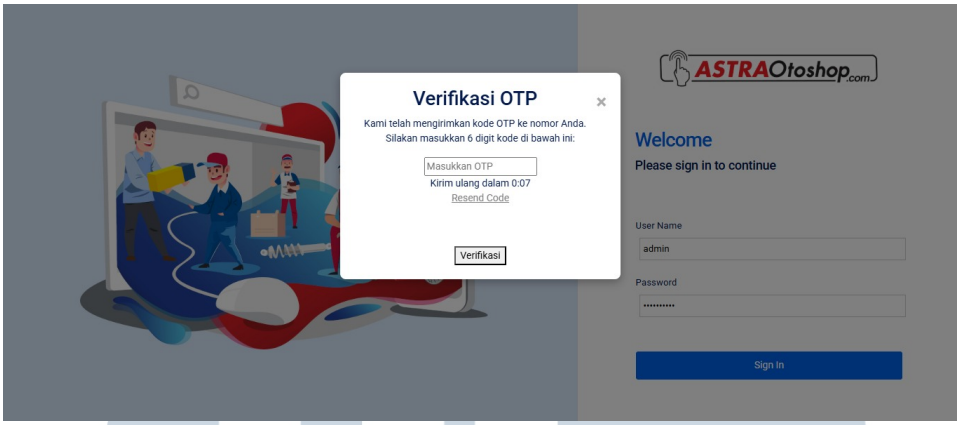
### 3.4.2 Tugas 2: Penambahan Fitur OTP pada Halaman Login Admin Panel

Proyek selanjutnya adalah pengembangan fitur keamanan tambahan berupa One-Time Password (OTP) pada halaman login admin panel Astraotoshop. Pengembangan dimulai dengan mempelajari struktur dan alur kerja source code yang sudah ada untuk memahami sistem login secara menyeluruh.

Langkah pertama dilakukan dengan menambahkan kolom baru pada database untuk menyimpan kode OTP yang akan dikirim dan divalidasi. Database yang digunakan adalah SQL Server Management Studio (SSMS). Setelah itu, dirancang antarmuka input OTP pada halaman login, yang akan muncul setelah pengguna berhasil memasukkan username dan password, namun sebelum diarahkan ke dashboard admin.

Pembuatan API untuk pengiriman dan verifikasi OTP dilakukan menggunakan framework ASP.NET dengan bahasa pemrograman C. Di sisi antarmuka, proses pengambilan dan validasi OTP dikembangkan menggunakan JavaScript untuk memastikan interaksi yang responsif dengan pengguna.

Query untuk pengelolaan data OTP ditambahkan ke dalam repository agar terintegrasi dengan sistem database yang sudah berjalan. Proses debugging dilakukan untuk memastikan fitur OTP dapat berjalan dengan baik, mulai dari pengiriman kode, input pengguna, hingga validasi dan pengalihan halaman secara aman dan lancar. Seluruh proses dirancang agar terintegrasi dengan mulus dalam sistem login yang telah ada sebelumnya.




Gambar 3.6. Tampilan Modal OTP Login pada Admin Panel

Gambar 3.6 menunjukkan tampilan modal OTP yang muncul setelah pengguna berhasil memasukkan username dan password pada halaman login admin panel. Pengguna diminta memasukkan kode OTP yang dikirim untuk proses verifikasi lanjutan sebelum masuk ke Home.

```
function btnSubmit_Click(sender, args) {
    fHoldButton($('#btnSubmit'));
    // Lakukan Login dahulu
    fPost('Auth','fAuth', fCollect(), 'fAuth');
    }
    // Tampilkan Modal
    function showOTPModal() {
        document.getELementById("otpModal").style.display = "block
    ";
    }
    // Sembunyikan Modal
    function closeOTPModal() {
        document.getElementById("otpModal").style.display = "none
    ";
    }
    // Verifikasi OTP
    function verifyOTP() {
        const otp = document.getElementById("otpInput").value.trim
    ();
        const username = document.getElementById("txtClientCode").
    value.trim();

        ApiService.post('/admin-auths/verify-otp', { username, otp
     }, {}, 'https://localhost/B2BR2.API')
            .then(res => {
                if (res.status === "success") {
                    alert("OTP benar. Login dilanjutkan.");
                    closeOTPModal();
                    window.Location = /B2BR2.Admin/Home/';
                } else {
                    alert(res.message || "OTP salah. Silakan coba
    Lagi.");
                }
            })
            .catch(err=> {
```

```
30                  alert("Gagal menverifikasi OTP.");
31                  console.error(err);
32              });
33      }
34
35  function fPost_Callback(sender, args)
36      if (sender= "fAuth") {
```

Kode 3.5: Kode OTP Login pada Admin Panel

Kode 3.5 merupakan potongan kode yang mengatur proses pengiriman dan validasi OTP pada halaman login admin, melibatkan pemanggilan API serta logika untuk mencocokkan input pengguna dengan OTP yang tersimpan di database.

## 3.5 Kendala dan Solusi yang Ditemukan

### 3.5.1 Kendala

Selama proses pengembangan fitur pada aplikasi Astraotoshop, terdapat beberapa kendala teknis dan non-teknis yang dihadapi, antara lain:

- Kompleksitas source code yang tinggi karena merupakan proyek lanjutan dari vendor sebelumnya, dengan struktur kode yang belum terdokumentasi dengan baik dan menggunakan gaya pemrograman yang tidak mengikuti standar modern.
- Kurangnya dokumentasi resmi dari pihak vendor sebelumnya mengenai alur sistem dan struktur database, sehingga mempersulit proses pemahaman logika yang telah ada.
- Integrasi API pembayaran baru memerlukan penyesuaian pada beberapa halaman sekaligus, yang saling bergantung satu sama lain, seperti halaman *EntryOrder*, *Payment*, dan *Change Payment Method*.
- Perubahan kecil pada salah satu bagian sistem dapat berdampak pada fungsi lainnya, sehingga proses *debugging* membutuhkan waktu lebih lama.

### 3.5.2 Solusi

Beberapa langkah yang dilakukan untuk mengatasi kendala-kendala di atas adalah sebagai berikut:

- Melakukan pembacaan kode secara menyeluruh secara bertahap dan dibantu dengan proses *debugging* manual untuk memahami alur kerja sistem dari sisi *frontend* hingga *backend*.
- Menambahkan komentar tambahan dan dokumentasi internal secara berkala guna mempermudah pengembangan lanjutan dan menghindari kesalahan pemahaman.
- Menggunakan *tools debugging* dan *console log* untuk menelusuri jalur eksekusi dan respons API secara detail.
- Menjalin komunikasi intensif dengan rekan *developer* dan melakukan koordinasi harian jika dibutuhkan untuk mempercepat proses klarifikasi dan validasi pengembangan fitur.
- Menggunakan *database* lokal untuk keperluan *testing* sementara sebelum implementasi akhir pada *database* produksi.