Buku Panduan

Technical Guide Error Handling Website

Website Catalog Produk

jusfabel.shop

Di susun oleh :

- **Adelio Bifarel**
- **Muhammad Radia Widiazto**
- **Ratri Yulia Annisa. T**
- **Robertus Ega Kurniawan**

Universitas Gunadarma

Fakultas Ilmu Teknologi dan Informasi

Jurusan Sistem Informasi

1. Pendahuluan

Panduan ini menjelaskan cara menangani error dari sisi frontend, backend, user, maupun admin website justfabel. Fokusnya adalah memastikan pengalaman pengguna yang mulus dan sistem yang tangguh saat menghadapi error.

---

2. Error Handling pada Backend

Masalah Umum

- Koneksi ke database gagal.
- Kredensial database salah.
- Server database tidak tersedia.
- Gagal menyimpan atau mengambil data dari database.

Solusi

1. **Gunakan blok try-catch untuk Koneksi Database:**

```php
<?php
try {
    $koneksi = new PDO("mysql:host=localhost;dbname=jusfabel", "username", "passwo
    $koneksi->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch (PDOException $e) {
    die("Koneksi gagal: " . $e->getMessage());
}
?>
```

2. **Validasi Data Input pada Backend:**

   - Pastikan data dari frontend sudah tervalidasi sebelum disimpan:

```php
if (empty($_POST['nama']) || empty($_POST['email'])) {
    die("Nama dan email wajib diisi.");
}
if (!filter_var($_POST['email'], FILTER_VALIDATE_EMAIL)) {
    die("Format email tidak valid.");
}
```

3. **Gunakan Blok Try-Catch untuk Query Database:**

```
try {
    $query = $koneksi->prepare("INSERT INTO users (nama, email) VALUES (?, ?)");
    $query->execute([$_POST['nama'], $_POST['email']]);
} catch (PDOException $e) {
    die("Gagal menyimpan data: " . $e->getMessage());
}
```

4. **Error Logging pada Backend:**
   - Gunakan `error_log()` untuk mencatat kesalahan:

```
error_log("Error pada proses penyimpanan data", 3, "error_backend.log");
```

---

3. Error Handling pada Frontend

Masalah Umum

- Validasi form tidak dilakukan atau tidak akurat.
- Tidak ada umpan balik (feedback) saat error terjadi.
- Tampilan error tidak ramah pengguna.

Solusi

1. **Validasi Form di Frontend:**
   - Gunakan JavaScript untuk memastikan form diisi dengan benar:

```
<form id="myForm" onsubmit="return validateForm()">
    <input type="text" id="nama" placeholder="Nama">
    <input type="email" id="email" placeholder="Email">
    <button type="submit">Kirim</button>
</form>

<script>
    function validateForm() {
        const nama = document.getElementById('nama').value;
        const email = document.getElementById('email').value;

        if (!nama || !email) {
            alert('Semua field wajib diisi.');
            return false;
        }

        const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
        if (!emailRegex.test(email)) {
            alert('Format email tidak valid.');
            return false;
        }

        return true;
    }
</script>
```

2. **Tampilkan Pesan Error yang Ramah Pengguna:**
   - Gunakan elemen HTML untuk menampilkan pesan error:

```
<div id="errorMessage" style="color: red; display: none;"></div>

<script>
    function showError(message) {
        const errorDiv = document.getElementById('errorMessage');
        errorDiv.textContent = message;
        errorDiv.style.display = 'block';
    }
</script>
```

3. **Tangani Error Ajax/Fetch:**
   - Pastikan error dari backend ditangani dengan baik di frontend:

```
<script>
    fetch('/api/submit', {
        method: 'POST',
        body: new FormData(document.getElementById('myForm'))
    })
    .then(response => {
        if (!response.ok) {
            throw new Error('Terjadi kesalahan pada server.');
        }
        return response.json();
    })
    .then(data => {
        alert('Data berhasil dikirim!');
    })
    .catch(error => {
        showError(error.message);
    });
</script>
```

4. **Gunakan CSS untuk Styling Pesan Error:**
   - Buat tampilan pesan error lebih menarik:

```
#errorMessage {
    color: #fff;
    background-color: #f44336;
    padding: 10px;
    border-radius: 5px;
    margin-top: 10px;
}
```

---

4. Error Handling pada Sisi User

Masalah Umum

- Input user tidak sesuai format.
- Kesalahan saat mengirimkan form.
- Halaman atau data tidak ditemukan.

Solusi

1. **Validasi Input User:**
   - Pastikan semua input user sudah tervalidasi di sisi frontend dan backend:

```
if (empty($_POST['username']) || empty($_POST['password'])) {
    die("Username dan password wajib diisi.");
}
```

2. **Berikan Feedback yang Informatif:**
   - Jika terjadi kesalahan, tampilkan pesan yang menjelaskan kesalahan dan cara memperbaikinya:

```
echo "Login gagal. Pastikan username dan password benar.";
```

3. **Tangani Kesalahan Halaman Tidak Ditemukan (404):**
   - Buat halaman 404 khusus:

```
<h1>404 - Halaman Tidak Ditemukan</h1>
<p>Maaf, halaman yang Anda cari tidak tersedia.</p>
```

---

5. Error Handling pada Sisi Admin

Masalah Umum

- Gagal login ke panel admin.
- Kesalahan saat mengelola data (CRUD).
- Akses halaman tanpa otorisasi.

Solusi

1. **Validasi Login Admin:**
   - Pastikan hanya admin yang terverifikasi dapat mengakses panel:

```
session_start();

if (!isset($_SESSION['admin_logged_in'])) {
    header("Location: login.php");
    exit();
}
```

2. **Tangani Kesalahan CRUD:**
   - Saat menghapus atau memperbarui data:

```
try {
    $query = $koneksi->prepare("DELETE FROM produk WHERE id = ?");
    $query->execute([$id_produk]);
} catch (PDOException $e) {
    die("Gagal menghapus data: " . $e->getMessage());
}
```

3. **Tampilkan Pesan Error yang Jelas:**
    - Jika terjadi error saat mengelola data, tampilkan pesan error yang dapat membantu admin:

```
echo "Gagal memperbarui data. Periksa input Anda.";
```

4. **Tangani Akses Tanpa Izin:**
    - Redirect user yang mencoba mengakses halaman admin tanpa login:

```
if (!$_SESSION['is_admin']) {
    header("Location: forbidden.php");
    exit();
}
```

---

6. Kesimpulan

Error handling yang baik pada backend, frontend, serta sisi user dan admin, akan meningkatkan keandalan aplikasi sekaligus memberikan pengalaman yang lebih baik bagi semua pengguna. Dengan validasi yang tepat, pesan error yang jelas, dan logging yang terstruktur, aplikasi akan menjadi lebih aman dan mudah dikelola.