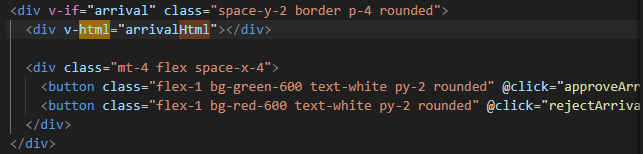
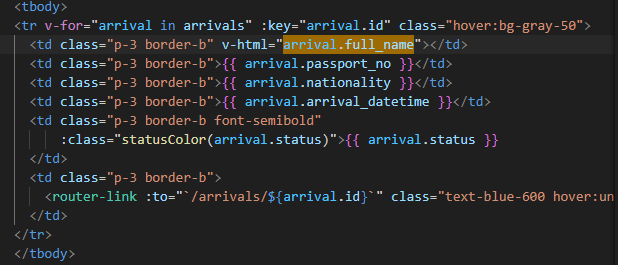
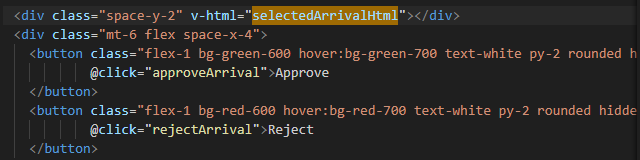
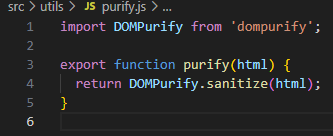
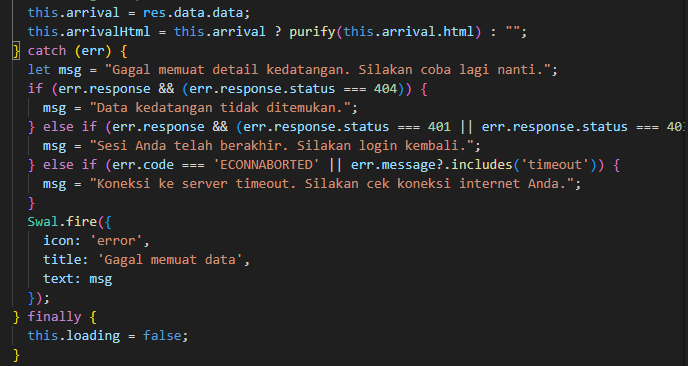
## Bug List

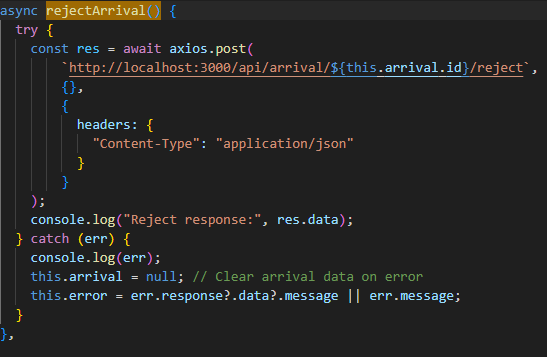
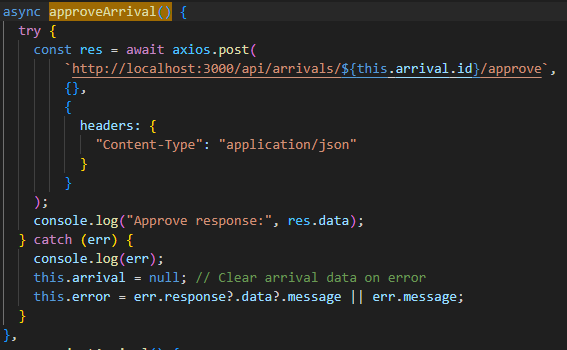
1. **XSS (Cross Site Scripting) – v-html**  
   *Rootcause*  
   Ditemukan penggunaan v-html di:

* Dashboard.vue (v-html="arrival.full\_name", v-html="selectedArrivalHtml")
* ArrivalDetail.vue (v-html="arrivalHtml")  
    
    
  *Impact*  
  Jika data yang di-render berasal dari user tanpa sanitasi, sangat rentan XSS.

Patch: Gunakan library sanitasi seperti DOMPurify sebelum assign ke variabel yang di-render dengan v-html.  


1. **CSRF**  
   *Rootcause:*  
   Tidak terlihat implementasi anti-CSRF token di frontend (umumnya di backend, tapi pastikan request POST/PUT/DELETE ke API Anda sudah memakai CSRF token atau SameSite cookie).  
   Tidak ada implementasi anti-CSRF token di bagian request POST, khususnya pada method:  
    approveArrival()

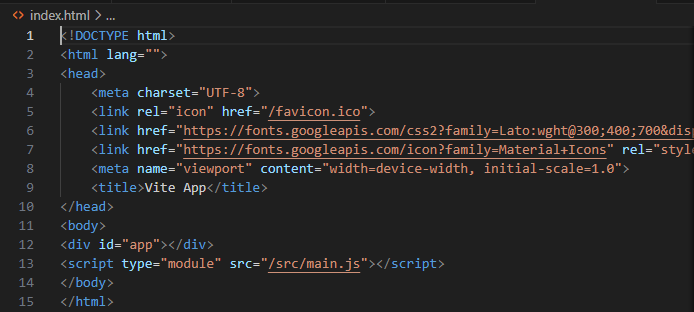
rejectArrival()



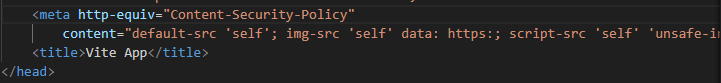
Pada kedua request tersebut, tidak ada pengiriman CSRF token di header atau body. Untuk aplikasi yang aman, biasanya backend akan meminta CSRF token (misal di header X-CSRF-Token) pada setiap request yang mengubah data (POST/PUT/DELETE). Jika backend Anda sudah mengaktifkan proteksi CSRF, frontend perlu mengirimkan token tersebut.

*Impact:*   
Aksi sensitif bisa dilakukan tanpa sepengetahuan user, karena backend tidak bisa membedakan request asli dari user atau dari attacker.

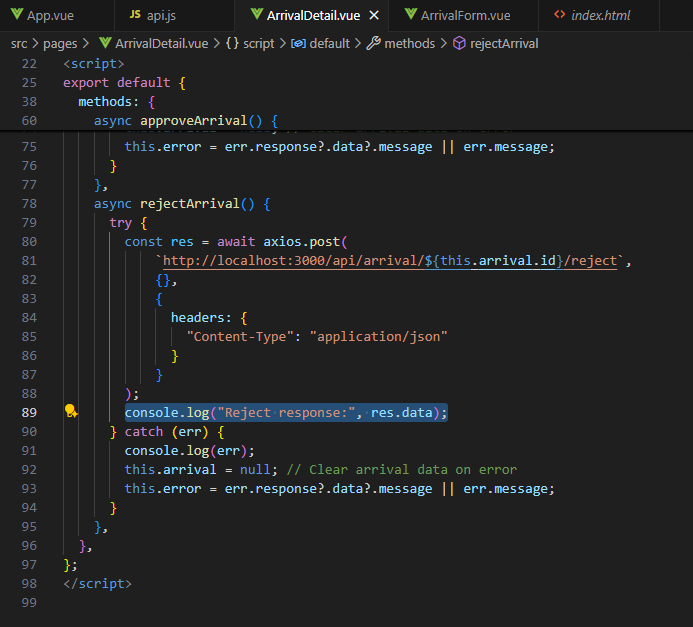
1. **Mixed Content**

*Rootcause:*  
Tidak ada pengaturan CSP di frontend.Tidak ada meta tag <meta http-equiv="Content-Security-Policy" ...> di file HTML utama (index.html).  
  
*Impact:*

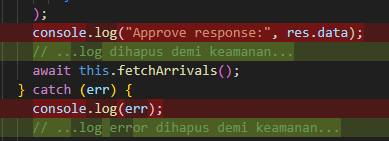
* Aplikasi lebih rentan terhadap serangan XSS (Cross-Site Scripting), karena browser tidak membatasi sumber script, style, atau resource lain yang boleh dijalankan.
* Attacker bisa lebih mudah menyisipkan/meng-inject script berbahaya dari domain luar, baik melalui input user, third-party script, atau dependency yang tidak aman.
* Potensi pencurian data user, pengambilalihan session, deface halaman, atau penyebaran malware meningkat.
* Sulit mendeteksi dan membatasi domain jahat yang mencoba menyisipkan konten/script ke aplikasi Anda.

Patch: Tambahkan CSP yang ketat, minimal default-src 'self'.  


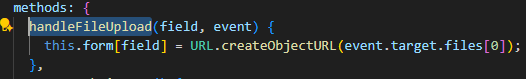
1. **Improper Error Handling**

Rootcase: Pastikan error detail tidak ditampilkan ke user di production.  
  
Kode di atas akan menampilkan isi variabel error secara langsung ke user jika terjadi error. Jika isi error berasal dari response backend yang berisi detail teknis (misal: stack trace, pesan error internal), maka detail tersebut bisa terlihat oleh user.

*Impact:*  
Informasi sensitif (data user, response API, error detail) bisa terlihat di browser console oleh siapa saja, termasuk attacker. Bisa membocorkan struktur data, flow aplikasi, atau bahkan credential jika tidak sengaja di-log.

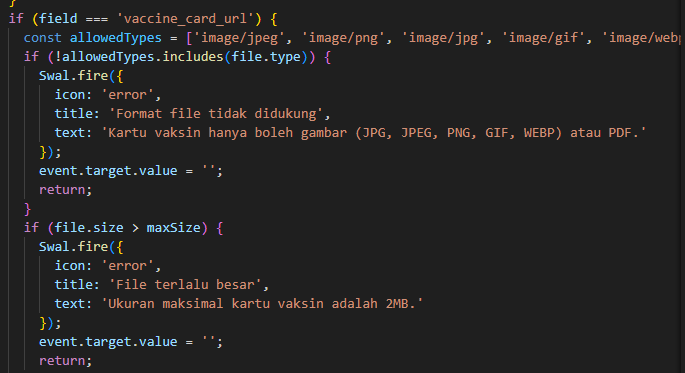
Patch: delete console log  


1. **Improper Input Validation**

*Rootcause:* File Upload tidak divalidasi, baik dari ukuran maupun dari extensi file

Impact:

* User bisa meng-upload file berbahaya (misal: script, executable, file besar, atau file yang tidak sesuai format).
* Potensi serangan malware, XSS (jika file HTML/script di-upload dan diakses), atau membebani server dengan file besar.

Patch: Pastikan backend tidak menerima data sensitif langsung dari frontend tanpa validasi.  


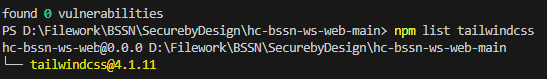
1. **Outdated Third-Party Component**

*Rootcause*:  
Audit dependency dengan npm audit dan hapus library yang tidak dipakai. Ketika dicek menggunakan *npm outdated,*  Tailwind CSS sudah outdated  


*Impact:*

* Potensi vulnerability di dependency lama yang belum di-patch.
* Beberapa plugin atau tools modern mungkin tidak kompatibel dengan versi lama.

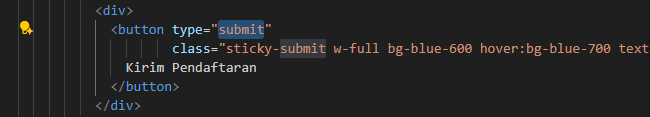
*Patch:*

Update tailwind ke versi terbaru

1. **Submit Button Validation**

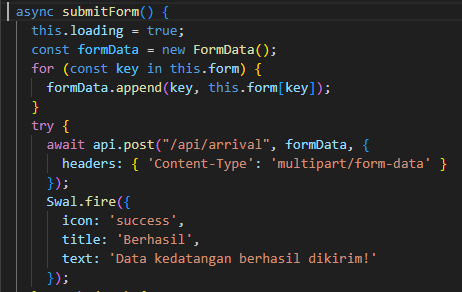
*Rootcause:*

Tidak ada atribut :disabled atau binding ke state loading, sehingga tombol bisa diklik berkali-kali selama proses submit berlangsung.



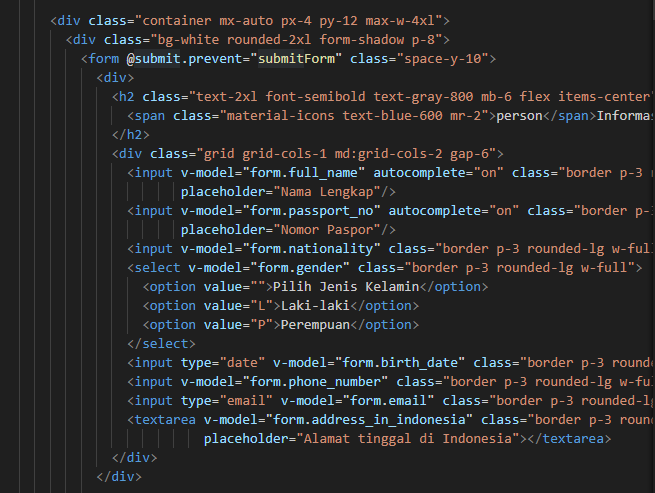
*Impact:*

* User bisa klik submit berkali-kali, sehingga data bisa terkirim ke backend lebih dari satu kali (double submit).
* Bisa menyebabkan data ganda di database (misal: pendaftaran dobel).
* Membebani server dengan request berulang.

Patch:  
Tambah validasi button, apabila sedang loading maka akan menjadi disabled  


1. **Text Input Validation**

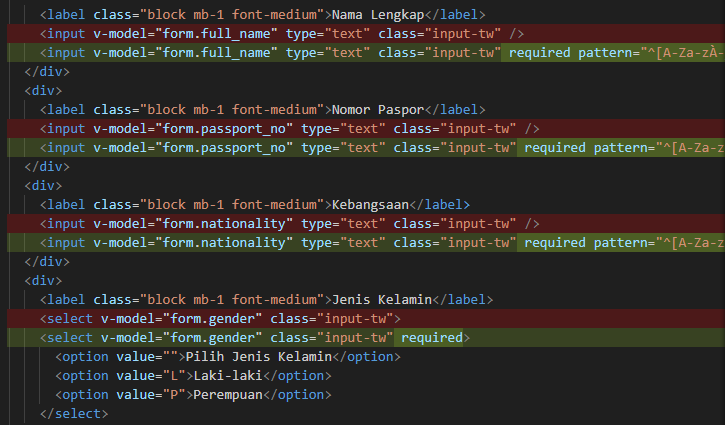
*Rootcause*:  
Tidak ada sanitasi input form maupun pengaturan min/max length karakter untuk input teks di ArrivalForm.vue. Semua input teks seperti form.full\_name, form.passport\_no, form.nationality, dan lain-lain hanya menggunakan v-model tanpa validasi panjang karakter atau filter karakter berbahaya.

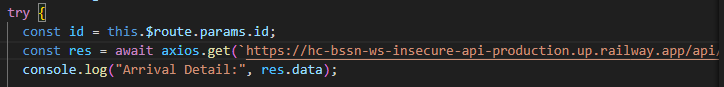


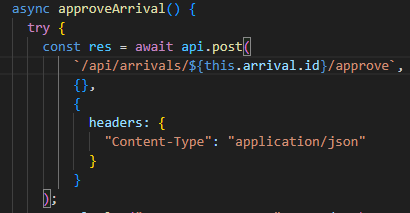
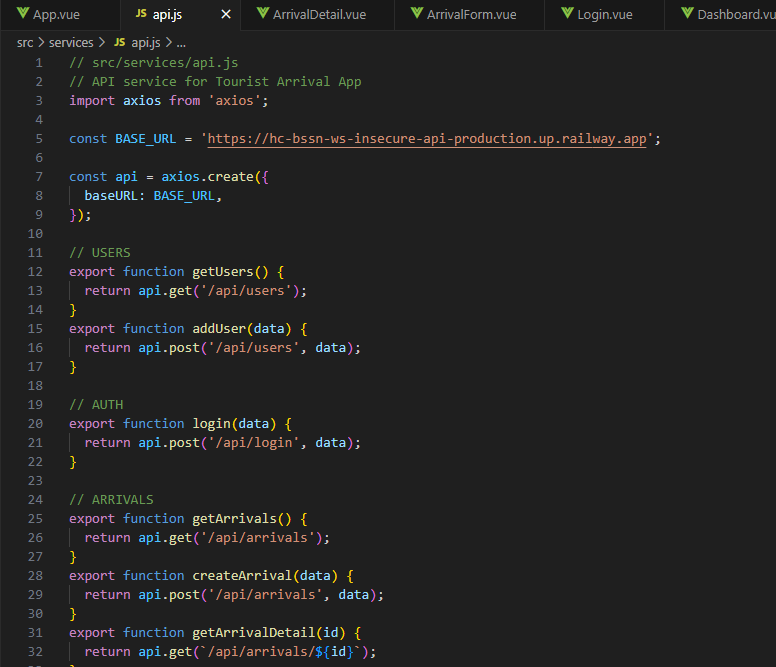
Impact:

* User bisa memasukkan script atau karakter berbahaya yang bisa menyebabkan XSS (Cross-Site Scripting) jika tidak disaring di backend.
* User bisa mengisi data dengan teks yang sangat panjang, menyebabkan error, membebani database, atau merusak tampilan aplikasi.
* Data yang masuk ke backend menjadi tidak konsisten, sulit divalidasi, dan berpotensi menyebabkan bug atau celah keamanan.
* Potensi abuse, misal spam data, deface, atau crash aplikasi karena input yang tidak wajar.

Patch:

Tambahkan validasi tipe kerakter, batas minimal da maksimal karakter, tipe file serta ukuran file  


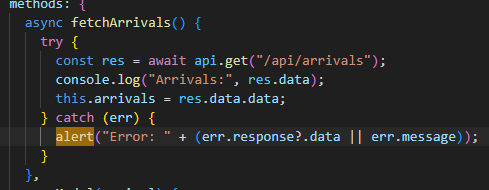
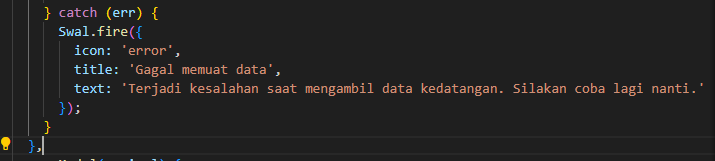
1. **Change Base Url***Rootcause;  
   -* Tidak ada konfigurasi terpusat untuk endpoint API di frontend.  
   - Kurangnya **awareness** bahwa hardcode URL mempersulit maintenance & meningkatkan risiko kebocoran informasi.  
   *Impact:  
   Base URL API terekspos di file JavaScript yang di-load user, memudahkan attacker melakukan direct API call.*  
   *Patch:*Buat **satu file config** yang menyimpan base\_url, lalu impor di semua file.  
   **Before:  
   **

**After:**Ada penambahan code pada [api.js](http://api.js) dan perubahan di masing-masing function****

1. **Remove Error Alert  
   Rootcause:**Programmer menampilkan error dari backend atau dari proses frontend langsung menggunakan alert(errorMessage), tanpa sanitasi atau formatting.

**Impact:**

Error asli bisa mengandung informasi sensitif (SQL error, stack trace, file path) yang memberi petunjuk pada attacker.

**Patch:**Ganti alert() dengan toast, modal, atau banner yang lebih user-friendly.  
Before:  
  
After:  
Menggunakan SweetAlert untuk UI improvement dan juga mengubah teks wording notifikasi  


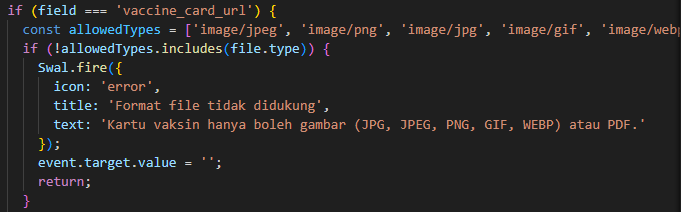
1. **Upload File Renaming  
   Rootcause:**User mengupload file dengan nama asli, dan sistem menyimpannya tanpa mengganti nama atau memfilter karakter berbahaya.

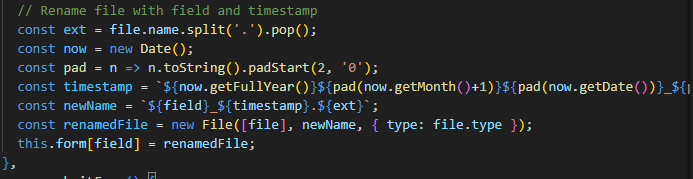
**Impact:**

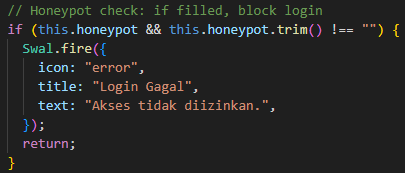
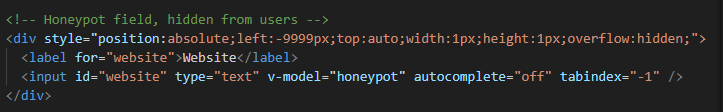
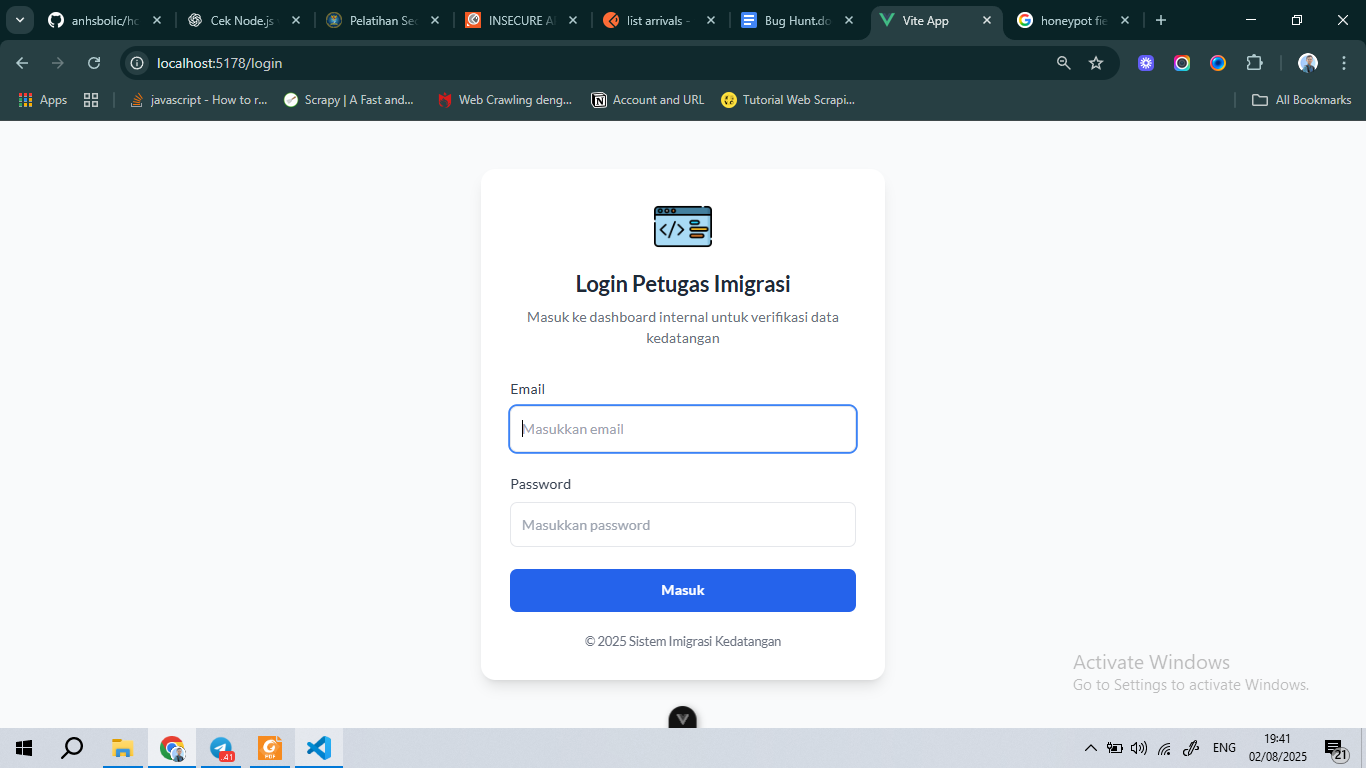
* Nama file seperti <script>alert(1)</script>.jpg bisa dieksekusi jika ditampilkan tanpa sanitasi di halaman web.
* Nama file asli bisa mengandung informasi sensitif (nama user, struktur sistem)
* Jika nama file dipakai langsung di URL, attacker bisa menebak nama file lain.

**Patch:**

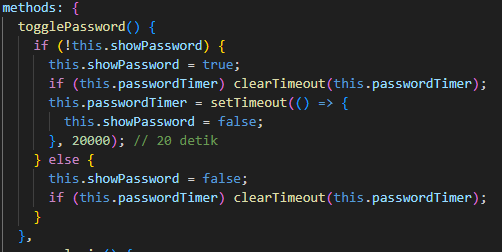
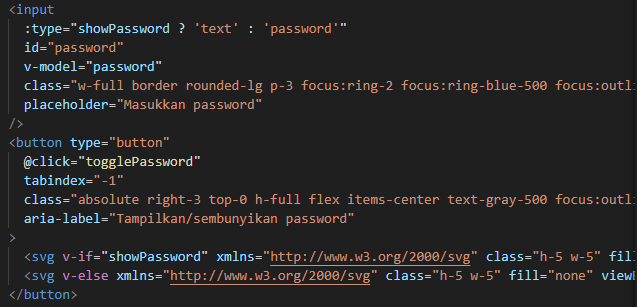
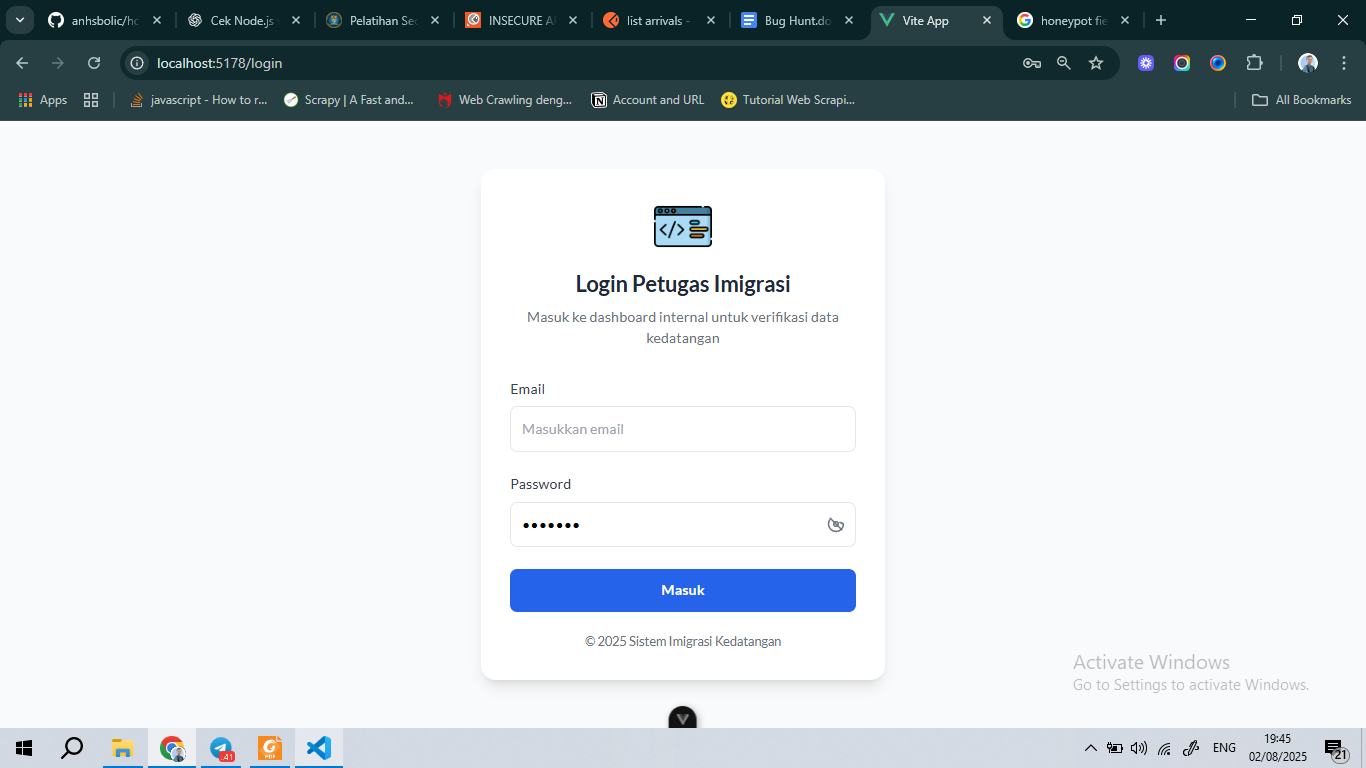
Rename file menggunakan Atau pakai timestamp + random string:

Before:  


After:  


1. **Honeypot Field Implementation**Rootcause:   
   Membuat sebuah field tersembunyi untuk yang hanya bisa diakses oleh bot, bukan manusia. Mencegah spam dan crawling  
     
   Before:   
   Tidak ada field HoneyPot.  
     
   Patch:  
   Honeypot field sudah ditambahkan di form login. Field ini tersembunyi dari user, tapi jika diisi (oleh bot), login otomatis diblokir dengan pesan "Akses tidak diizinkan." Cara ini efektif untuk menangkal bot sederhana tanpa mengganggu user asli.  
   
2. **Automatic Hide Password**Rootcause:  
   Password adalah hal yang sensitif, sehingga perlu dilakukan hide agar tidak bisa diketahui banyak orang dan tidak terekspos  
   

**Impact:**Orang lain didekat user bisa mengintip password yang sedang diketikkan

**Patch:**Menambahkan button show/hide, yang dalam waktu 20 detik akan otomatis ter-hide  


1. **Clear all data when logout  
   Rootcause:  
   Logout button diimplementasikan hanya sebagai redirect ke halaman login, tanpa:**

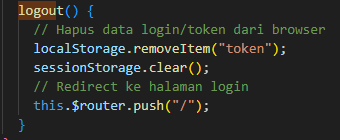
* Menghapus session token atau auth cookie di browser.
* Membersihkan localStorage/sessionStorage yang berisi data sensitif.

**Impact**

* Jika token/session tidak dihapus, siapa pun yang punya akses ke browser (misalnya di komputer publik) bisa langsung login tanpa kredensial.
* Data sensitif di localStorage/sessionStorage atau IndexedDB tetap tersimpan dan bisa dibaca oleh skrip berbahaya (XSS).

Patch:

**Hapus semua token** dari browser storage:

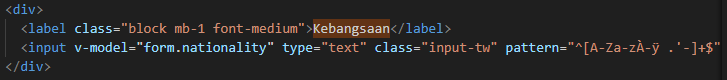
*Before:*  
*After:*

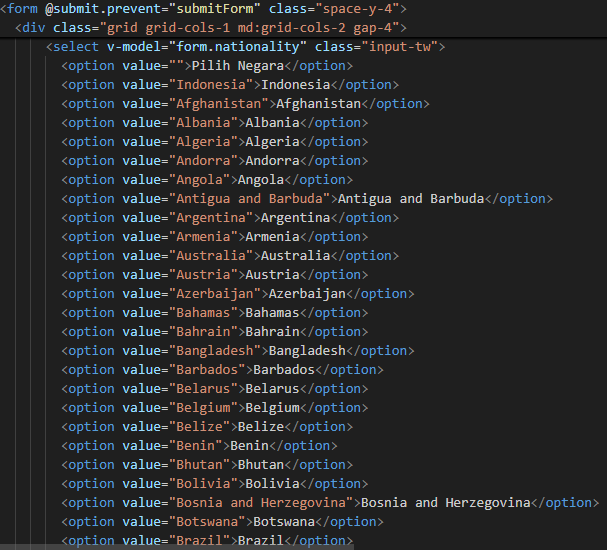
1. **Form Tampering  
   Rootcause:**Mencegah user menambahkan data yang tidak seharusnya ditambahkan, yaitu dengan menyediakan input field berisi opsi yang telah tersedia. Field kebangsaan awalnya berupa input text bebas. User bisa mengubah nilai di frontend (misalnya lewat DevTools atau manipulasi request).Bisa memasukkan data kebangsaan yang tidak valid atau tidak terdaftar.

**Impact:**

* Data kebangsaan di database tidak konsisten, muncul nilai yang tidak valid seperti “Negara123” atau “HackerLand”.
* Jika nilai kebangsaan diproses tanpa sanitasi, bisa memicu SQL Injection atau XSS melalui input yang dimanipulasi.

**Patch:**

Ganti input text bebas menjadi <select> dengan opsi kebangsaan resmi.  
**Before:  
**

**After:  
**

1. **Disable Autocomplete  
   Rootcause:**

Mencegah agar pengguna browser selain user sah mengetahui email / password, dan masuk tanpa credential yang sah

**Impact:**

* Orang di sekitar bisa melihat **dropdown suggestion** berisi email/username saat user mengetik.
* Di komputer publik atau device yang dipakai bersama, **email/password tersimpan di browser** dan bisa digunakan orang lain tanpa izin. Bisa menyebabkan **akun diambil alih** jika password tersimpan otomatis.

**Patch:**

Gunakan autocomplete="off" **hanya** pada perangkat publik/kiosk.  
Before:  


**Ater:  
**

1. **User Location Checking  
   Rootcase:**

Aplikasi digunakan hanya di Indonesia, sehingga lokasi devicenya tidak boleh diluar negeri

**Impact:**Agar Hanya user yang berlokasi di Indonesia yang bisa mengakses. User diluar Indonesia tidak bisa akses

**Patch:**

Menambahkan validasi berdasarkan device location, sehingga hanya bisa diakses oleh Indonesia  
Before: \*belum ada validasi  
After:  

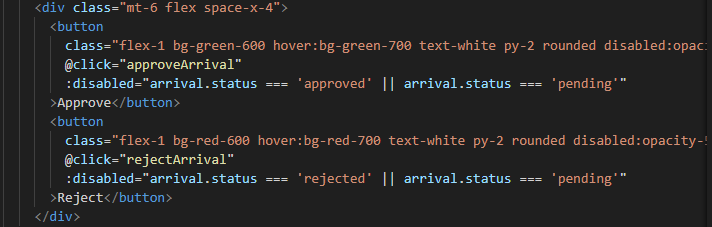

1. **UI Level Access Control  
   Rootcause:**User bisa mencoba melakukan aksi yang sudah tidak valid sesuai dengan statusnya

**Impact:**

Approval bisa diproses lebih dari sekali jika ada multiple klik.

**Patch:**

Membatasi elemen UI berdasarkan status atau hak akses pengguna.  
Before:  

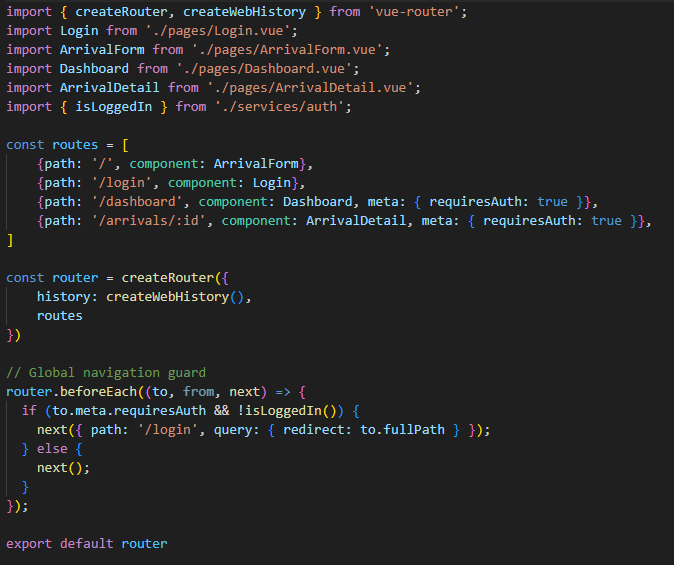

After:  


1. **Role Access Missing**

**Rootcause:**User bisa akses dashboard tanpa login, dan user bisa ke halaman login meskipun statusnya sudah login

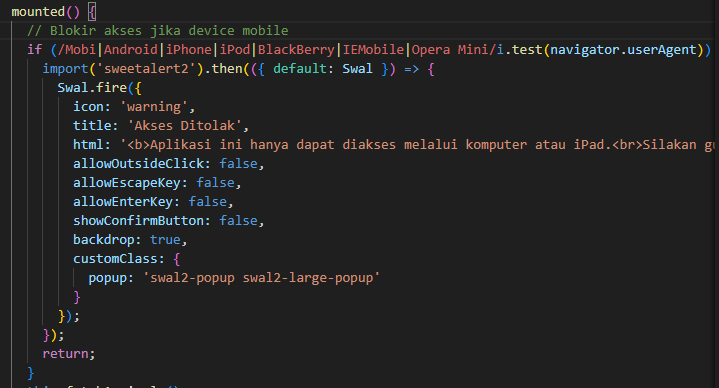
**Impact:**Data dan fitur di dashboard bisa diakses tanpa autentikasi.

Informasi sensitif bisa terlihat publik atau dicuri.

**Patch:**  
Tambahkan validasi dan tokenBefore: \*tidak ada redirect dan validasi token  
After: ada validasi token dan redirect  


1. **Desktop Access Only**

**Rootcause:**User staff hanya bisa akses melalui desktop dan Ipad saja, dan tidak bisa melalui mobile pada halaman Dashboard, Login, dan ArrivalDetails.  
**Impact:**Akses untuk role **staff** dibatasi hanya lewat perangkat desktop dan iPad.Jika menggunakan perangkat mobile (smartphone), halaman Dashboard, Login, dan ArrivalDetails akan diblokir.

**Patch:**Validasi Device di Frontend  
Before: \*tidak ada validasi  
After: 

1. **Session Expiration  
   Rootcause:**

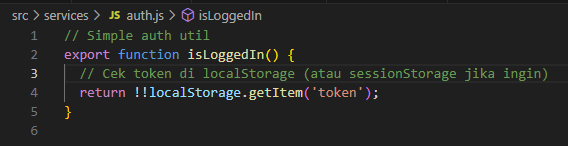
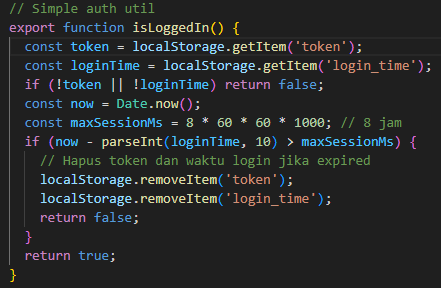
Batas waktu maksimum suatu sesi login sebelum user harus login ulang. Logout otomatis setelah waktu tertentu sejak login, walaupun user masih aktif (misalnya 8 jam).

**Impact:**

Absolute timeout wajib di sistem sensitif untuk mencegah:

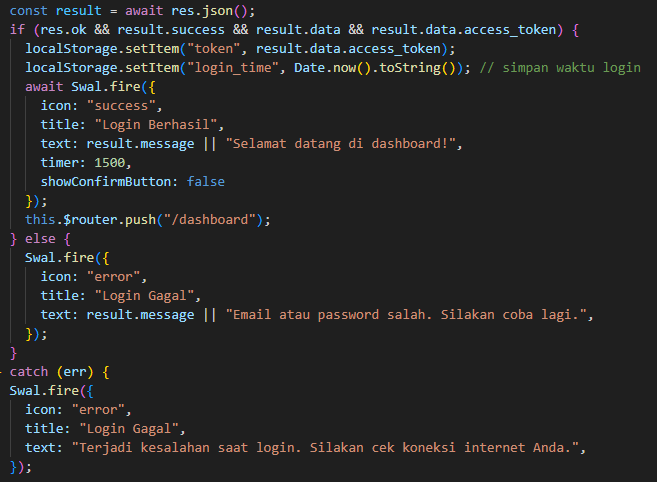
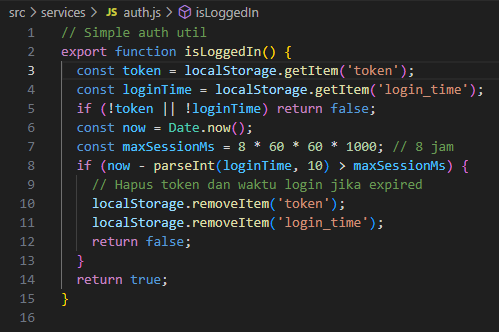
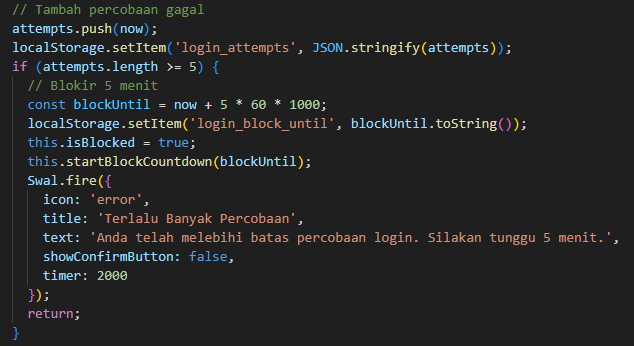
Session hijacking (attacker yang mendapatkan token bisa menggunakannya selamanya jika tidak ada batas).

Persistent session abuse.

**Patch**Before: \*tidak ada pembatasan waktu login  
After:  
Sekarang, saat login berhasil, waktu login akan disimpan. Fungsi isLoggedIn akan otomatis logout jika sudah lebih dari 8 jam sejak login, dan logout manual juga menghapus waktu login. Sesi user akan benar-benar dibatasi maksimal 8 jam.  


1. **Brute Force Protection  
   Rootcause:** Mencegah percobaan login massal dengan membatasi 5 percobaan gagal per 15 menit. Apabila masih gagal, maka akan di pending selama 5 menit. User tidak bisa login dan harus menunggu hingga waktu timer habis.   
   **Impact:**

* Penyerang mencoba menebak password dengan mencoba kombinasi username + password secara massal dan cepat.
* Tanpa pembatasan, attacker bisa melakukan ribuan percobaan per menit menggunakan bot atau script otomatis.

**Patch:**  
Before:  
  
  
After:

**Next Improvement:**

1. Menggunakan obscurity
2. Menggunakan validasi NIK di form secara otomatis
3. Menambahkan Captcha
4. Gunakan SSO untuk page Login karyawan
5. Pakai UUID, jangan pakai ID incremental untuk page detai arrival

**Link Repo:**

https://github.com/ArfHbb/SecureFrontend