

LAB3_Report

Name: Quang Hung Tran

ID: n01650970

Introduction

This lab upgrades the Incident Tracker system by replacing in-memory storage with JSON file persistence. A new ARCHIVED status was implemented with strict transition rules and dashboard configuration to control visibility.

Project Structure and File Purpose

- `app.js` - Starts the Express server and connects all routes.
- `incidents.routes.js` - Defines all API endpoints for CRUD and bulk upload.
- `incidents.store.js` - Handles reading and writing incidents to JSON file.
`validate.js` - Validates input and enforces status transition rules.
- `config.js` - Stores configuration such as file path and status constants.
`incidents.json` - Stores incident data permanently in JSON format.
- `api.js (frontend)` - Sends API requests from frontend to backend.

Key Functions Explanation

Backend

`backend/server.js`

Custom functions: *(không có function đặt tên)*

Logic chính:

- `app.listen(PORT, callback)` (callback là function)
 - Purpose: start API server on a port.
-

backend/src/app.js

Endpoints / middleware handlers (tính như function trong report):

- `app.get("/health", (req,res) => ...)`
 - Purpose: health check endpoint.
- `app.use("/api/incidents", incidentsRouter)`
 - Purpose: mount incidents routes.
- `app.use((req,res) => ...)`
 - Purpose: fallback 404 "Route not found".
- `app.use((err, req, res, next) => ...)`
 - Purpose: global error handler (500).

Ghi trong report: mấy cái `(req,res)=>{}` này là "route handler / middleware function".

backend/src/routes/incidents.routes.js

Route handler functions (endpoints):

- `router.get("/")`
 - Purpose: return list of incidents
 - Lab 3: có thể thêm query `showArchived`.
- `router.get("/:id")`
 - Purpose: return single incident by id.
- `router.post("/")`
 - Purpose: create new incident (validate first).
- `router.patch("/:id/status")`
 - Purpose: update incident status (validateStatusChange).

- `router.post("/bulk-upload", upload.single("file"), async ...)`
 - Purpose: parse CSV and create multiple incidents.

Other objects / helpers used in file (không phải function riêng):

- `multer.memoryStorage()` + `upload.single("file")`
 - Purpose: handle file upload.
-

backend/src/store/incidents.store.js

Functions (đặt tên rõ ràng):

- `listAll({ includeArchived } = {})`
 - Purpose: return incidents list; optionally include archived.
- `findById(id)`
 - Purpose: find single incident by id.
- `createIncident(data)`
 - Purpose: create a new incident object + save.
- `updateStatus(id, status)`
 - Purpose: update status + save.

Nếu ông có thêm persistence thì trong report cũng liệt kê thêm helper kiểu:

`ensureDataFile()`, `readAll()`, `writeAll()` (nếu file ông có).

backend/src/utils/csv.js

Functions:

- `parseCsvBuffer(buffer)`
 - Purpose: convert CSV file buffer into JS objects (rows).
-

backend/src/utils/validate.js

Functions:

- `validateCreateIncident(body)`

- Purpose: validate input when creating incident.
 - `validateStatusChange(current, next)`
 - Purpose: enforce allowed status transitions (Lab 3 ARCHIVED rules nằm ở đây).
-

Frontend

`frontend/services/api.js`

Functions:

- `handleJson(res)`
 - Purpose: parse response JSON + throw readable error.
 - `health()`
 - Purpose: call backend health endpoint.
 - `listIncidents(showArchived = false)`
 - Purpose: fetch list; Lab 3 adds `?showArchived=true`.
 - `getIncident(id)`
 - Purpose: fetch single incident.
 - `createIncident(payload)`
 - Purpose: create incident via POST.
 - `changeIncidentStatus(id, status)`
 - Purpose: PATCH status update.
 - `bulkUploadCsv(file)`
 - Purpose: upload CSV to backend.
-

`frontend/components/Layout.js`

Function / Component:

- `Layout(props)`

- Purpose: shared page layout wrapper.

frontend/components/ErrorBanner.js

Function / Component:

- `ErrorBanner({ message })`
 - Purpose: show error message UI.

frontend/pages/_app.js

Function / Component:

- `App({ Component, pageProps })`
 - Purpose: Next.js root wrapper for all pages.

frontend/pages/index.js

Function / Component:

- `Home()`
 - Purpose: homepage / navigation entry.

frontend/pages/dashboard.js

Functions / Components:

- `Dashboard()`
 - Purpose: dashboard page UI.
- `StatusColumn(...)`
 - Purpose: render status columns (UI helper component).

frontend/pages/incidents/index.js

Functions / Components:

- `IncidentsList()`
 - Purpose: show incidents table/list.

- `load(nextShowArchived = showArchived)` (*inner function*)
 - Purpose: fetch list from API and update state.
-

`frontend/pages/incidents/create.js`

Functions / Components:

- `CreateIncident()`
 - Purpose: create incident form page.
 - `onSubmit(e)` (*inner handler*)
 - Purpose: submit form → call API create.
-

`frontend/pages/incidents/[id].js`

Functions / Components:

- `IncidentDetails()`
 - Purpose: incident details page.
 - `load() / load(id)` (*inner function*)
 - Purpose: fetch incident detail.
 - `onUpdateStatus()` (*inner handler*)
 - Purpose: call API to update status.
-

`frontend/pages/bulk-upload.js`

Functions / Components:

- `BulkUpload()`
 - Purpose: bulk upload UI page.
- `onUpload()` (*inner handler*)
 - Purpose: send CSV file to backend bulk-upload.

Application Workflow

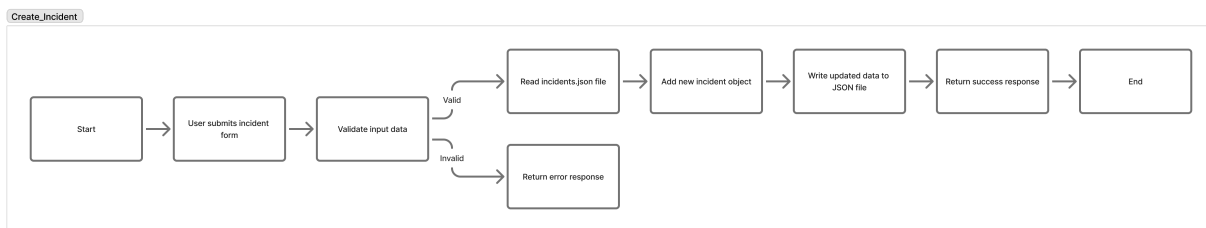
User action → Frontend sends request → Backend route processes request → Validation → JSON file read → Data modified → JSON file written → Response returned → UI

Flowcharts

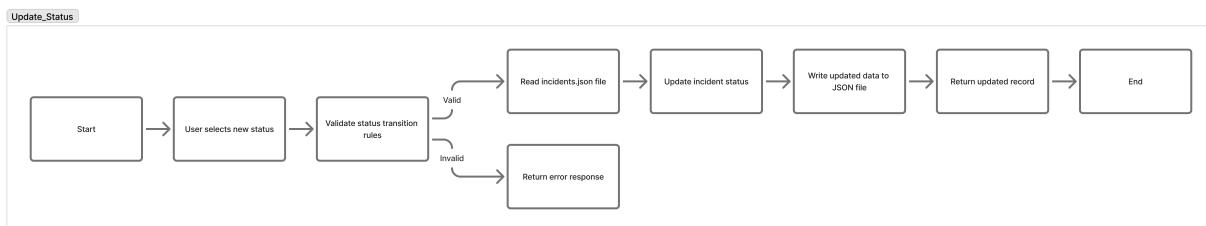
Read Incident



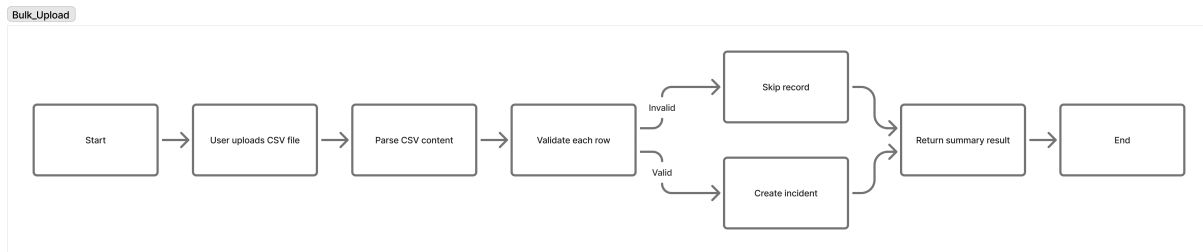
Create Incident Flow



Update Status Flow



Bulk Upload Flow



Conclusion

The system now supports persistent JSON storage and controlled ARCHIVED status

transitions. Archived incidents are hidden by default and can be displayed using a checkbox configuration. This improves reliability and business rule enforcement.