Test Case builder

A full-stack app to compose, validate, and manage UI test cases as JSON files.

- Frontend: React + Vite + Tailwind (gradient icon buttons, glyph icons, two-pane UI)
- Backend: Node.js + Express with AJV schema validation and file-based storage
- Storage: Each test case is a JSON file under `backend/data/`

Features

- Create, edit, duplicate, delete test cases
- Import/export single or multiple cases (drag & drop JSON supported)
- Batch create editor with inline validation and duplicate-order detection
- Drag-and-drop and keyboard reordering (Alt+↑/Alt+↓) updates `testOrder` and persists to backend
- Runner-aligned fields: action, selector/selectorType, path, data, waitTime, iterate, customName
- Validation types with conditional required fields (attribute/cssProperty)
- Unique `testOrder` enforced on server; missing orders auto-assigned
- Required field indicators with ARIA hints
- Consistent glyph iconography (Material Symbols webfont)
- Gradient-colored icon buttons; plain slide toggle for Enabled

Repository layout

- `backend/` Express API, schema, and JSON storage
  - `server.js` — REST API for CRUD and import
  - `lib/schema.js` — AJV schema + helpers
  - `data/` — JSON files per test case
- `frontend/` React app (Vite)
  - `src/components/` — Dashboard, TestCaseForm, BatchCreate, etc.
  - `index.html`, `index.css` — App shell and global styles
  - `vite.config.ts`, `tailwind.config.js`, `postcss.config.js`

Prerequisites

- Node.js 18+ (recommended)

Setup and run (development)

Run backend (port 4000 by default):

```
cd backend
npm install
node server.js
```

Run frontend (Vite dev server):

```
cd frontend
npm install
npm start
```

- Frontend dev server will start (typically http://localhost:5173)
- Backend runs on http://localhost:4000 with CORS enabled

Build (frontend)

```
cd frontend
npm run build
npm run preview  # optional local preview of the built app
```

Preview runs on http://localhost:4173 by default.

Using the app

- Left pane: searchable list of test cases with actions (edit, export, duplicate, delete). Drag handle to reorder; order persists to backend.
- Right pane: editor area.
    - Single mode: detailed form for one test case.
    - Batch mode: accordion view, one open at a time; quickly compose many cases; Save All writes them to backend.
- Import: toolbar button or drop JSON files into the dropzone.
- Export: one or all cases as JSON.
- Enabled: slide toggle icon shows current state; can be edited inside forms.

Data model (schema)

High-level shape (simplified):

```
{
    "description": "string",
    "enabled": true,
    "testOrder": 1,
    "testSteps": [
        {
            "stepName": "string",
            "action":
"goto|click|fill|type|press|hover|waitForTimeout|custom",
            "selector": "string",
            "selectorType": "css|xpath|id|text|testId",
            "path": "string",
            "data": "string",
            "waitTime": 0,
            "iterate": false,
            "customName": "string",
            "soft": false,
            "validations": [
                {
                    "type": "toBeVisible|toBeHidden|toHaveTitle|toHaveURL|toHaveText|toHaveValue|toHaveAttribute|toHaveCSS|toHaveClass",
                    "selector": "string",
                    "selectorType": "css|xpath|id|text|testId",
                    "path": "string",
                    "data": "string",
                    "message": "string",
                    "soft": false,
                    "attribute": "string",
                    "cssProperty": "string"
                }
            ]
        }
    ]
}
```

Notes:

- For `action: goto`, `path` is required.
- For `click|hover|fill|type|press`, `selector` is required.
- For `fill|type|press`, `data` is required.
- For `waitForTimeout`, `waitTime >= 0` is required.
- For `custom`, `customName` is required.
- For validations:
  - `toHaveText|count|text` expect `data`.
  - `toHaveAttribute` expects `attribute`.
  - `toHaveCSS` expects `cssProperty`.

API

Base URL: `http://localhost:4000/api`

- `GET /testcases` → list all cases (sorted by `testOrder`)
- `POST /testcases` → create one or many cases
  - Body: object or array of objects per schema
  - Server enforces unique `testOrder`; auto-assigns missing
- `PUT /testcases/:filename` → update existing case
- `DELETE /testcases/:filename` → delete case

Response examples are standard JSON. Errors include a message and optional details (e.g., duplicate testOrder diagnostics).

Storage

- Files are saved in `backend/data/*.json` using a normalized name from the description (`normalizeFilename`), with numeric suffixes when needed.

Accessibility & UX

- Required fields are indicated with asterisks and ARIA attributes.
- Icon-only buttons have titles (tooltips) and accessible labels.
- Consistent glyph icon sizes; gradient icon buttons; plain toggle for Enabled.

Troubleshooting

- Duplicate order errors on save/import: fix or remove conflicting `testOrder`; server assigns missing orders automatically.
- To reset data, stop the backend and remove JSON files under `backend/data/` (keep backups as needed).
- If styles look off, ensure Tailwind build is running (Vite dev server) and `frontend/index.css` includes Tailwind layers.

Roadmap / Ideas

- Clickable enable/disable toggle in the list with immediate backend update
- Tests for schema validation and UI forms
- Optional auth layer and user workspaces

License

Not specified.

UI Testcase Tool

A small tool to create, edit, import, export, and order UI test case JSON files.

Features

- Create and edit test cases with steps and validations
- Import JSON (file picker or drag-and-drop). Accepts a single object or an array
- Export a single test or all tests as JSON
- Drag-and-drop reordering with persistent testOrder
- Keyboard reordering: Alt+ArrowUp / Alt+ArrowDown on a card
- Bulk selection: Delete Selected, Duplicate Selected
- Inline validation errors in the form

Project structure

- `backend/`: Node/Express API storing JSON files in `backend/data/`

- `frontend/`: React + Vite UI

Run locally

Backend (first time):

```
cd backend
npm install
node server.js
```

This starts the API at http://localhost:4000

Frontend:

```
cd frontend
npm install
npm run start
```

Open the URL shown by Vite (typically http://localhost:5173)

Sample JSON

See `frontend/samples/test-cases.json` for an example import file.

Notes

- New files are named automatically by the backend based on description
- If testOrder is empty on create, the backend assigns the next order
- Validation errors from the server are shown in the form