VesselVerse (website)

VesselVerse

A Dataset and Collaborative Framework for Vessel Annotation

VesselVerse Webpage is a React single-page website built with Vite. It showcases the VesselVerse dataset and provides interactive viewers and documentation for medical images (NIfTI, VTK, Three.js-based viewers).

What's new (2025-11-23)

- Carousel on the Learn More page: includes the tutorial video and two spotlight Google Slides embeds (auto-advance, manual controls).
- Timeline improvements: timeline steps animate in/out continuously while scrolling (appear/disappear on enter/leave).
- UI helpers: Back-to-Top butto (persisted preference).
- Paper page: author logos, improved author cards and responsive layout; FAQ entries added.

Developer note: several small UI/UX polish changes were applied across pages (hover lift/shadow, responsive grid for authors).

At a glance

- Website pages: `Home`, `Dataset`, `Framework`, `Paper`, `LearnMore` (under `src/pages`)
- Interactive components: `NiiViewer`, `VolumeViewer`, `VesselAnimation` (under `src/components`)
- Static assets: stored in `public/` (images, example NIfTI files)
- Scripts: `dev`, `build`, `preview`, `deploy` in `package.json`

PROF

Paper

VesselVerse is described in a MICCAI 2025 conference paper that introduces the dataset and the collaborative framework for vessel annotation. The paper summarizes the data sources, the multi-expert annotation and consensus pipeline, and demonstrates interactive tools for visualization and validation.

Download the paper (MICCAI 2025):

https://papers.miccai.org/miccai-2025/paper/0087_paper.pdf

See the `Paper` page on the website for the BibTeX citation, author list, and contact information.

Quickstart (develop locally)

Prerequisites

- Node.js LTS (recommended: 20.x). Many dev dependencies require Node >=18.
- npm (bundled with Node) or `pnpm`/`yarn` if you prefer.

Using `nvm` (recommended)

1. Install `nvm` (if not already installed):

   ```
   curl -fsSL https://raw.githubusercontent.com/nvm-
   sh/nvm/v0.39.6/install.sh | bash
   # then reload your shell or run:
   export NVM_DIR="$HOME/.nvm"
   [ -s "$NVM_DIR/nvm.sh" ] && . "$NVM_DIR/nvm.sh"
   ```

2. Install and use the LTS Node version:

   ```
   nvm install --lts
   nvm use --lts
   node -v
   npm -v
   ```

Install and run

```
# from the repo root (where package.json lives)
npm install
npm run dev
# open the URL printed by Vite (usually http://localhost:5173)
```

PROF

If something fails, try a clean reinstall:

```
rm -rf node_modules package-lock.json
npm install
```

Change port (optional):

```
npm run dev -- --port 3000
# or on POSIX shells:
```

```
PORT=3000 npm run dev
```

Build & Preview

Create a production build and preview it locally:

```
npm run build
npm run preview
# preview serves the built files and shows the production behavior
```

Note: there is a `postbuild` step that copies `dist/index.html` → `dist/404.html` to support GitHub Pages.

Deploy

Publish the `dist` folder to GitHub Pages with:

```
npm run deploy
```

Project structure (important files)

- `index.html` — app entry
- `src/main.jsx` — React entry & router
- `src/pages` — top-level pages
- `src/components` — UI components and viewers
- `public/` — static assets and example data
- `package.json` — scripts & dependencies

PROF

Troubleshooting

- If `npm run dev` throws syntax/import errors mentioning `node:fs/promises` or similar, your Node is too old — use Node >=18 (LTS 20 recommended).
- Remove `node_modules` and `package-lock.json` and reinstall if dependencies fail.
- If SPA routes 404 on refresh in production, ensure the server serves `index.html` as a fallback (the repo copies `404.html` for gh-pages).

Recommended next steps for contributors

- Add a `.nvmrc` file with the Node version (e.g. `20`) for consistency.
- Add `CONTRIBUTING.md` with instructions for working with dataset files and viewers.

- Add GitHub Actions to run `npm ci`, `npm run lint`, and `npm run build` on PRs.

React + Vite (short reference)

This project uses Vite as the build/dev tool and React for UI. Key points:

- `@vitejs/plugin-react` (Babel) or `@vitejs/plugin-react-swc` (SWC) enable JSX transforms and Fast Refresh.
- Use React.lazy + Suspense and dynamic imports to split heavy viewer code (three, vtk.js, niivue).
- Use `npm run preview` to test production behavior locally after `npm run build`.

License & Code of Conduct

Consider adding `LICENSE`, `CONTRIBUTING.md` and `CODE_OF_CONDUCT.md` to clarify reuse and contribution rules.
- npm (bundled with Node) or `pnpm`/`yarn` if you prefer.

~~~
Using nvm (recommended)
1. Install `nvm` (if not already installed):
   ```bash
   curl -fsSL https://raw.githubusercontent.com/nvm-
sh/nvm/v0.39.6/install.sh | bash
   # then reload your shell or run:
   export NVM_DIR="$HOME/.nvm"
   [ -s "$NVM_DIR/nvm.sh" ] && . "$NVM_DIR/nvm.sh"
   ```
2. Install and use the LTS Node version:
   ```bash
   nvm install --lts
   nvm use --lts
   node -v
   npm -v
   ```

Install dependencies and run dev server
```bash
# from the repo root (where package.json lives)
npm install
npm run dev
# open the URL printed by Vite (usually http://localhost:5173)
```

If you prefer a clean reinstall:
```bash
rm -rf node_modules package-lock.json
npm install
```

Change dev server port (optional):
~~~

~~~~
 ```bash
 npm run dev -- --port 3000
 # or on POSIX shells:
 PORT=3000 npm run dev
 ```

 ## Build & Preview
 Create a production build and preview it locally:
 ```bash
 npm run build
 npm run preview
 # preview serves the built files and shows the production behavior
 ```

 This project also includes a `postbuild` step that copies
`dist/index.html` to `dist/404.html` to support GitHub Pages SPA
routing.

 ## Deploy
 The repository includes a `deploy` script that uses `gh-pages` to
publish the `dist` folder to the `gh-pages` branch:
 ```bash
 npm run deploy
 ```

 ## Project structure (important files)
 - `index.html` — app entry
 - `src/main.jsx` — React entry & router
 - `src/pages` — top-level pages (Home, Dataset, Framework, Paper,
LearnMore)
 - `src/components` — UI components (NiiViewer, VolumeViewer,
VerticalTimeline, etc.)
 - `public/` — static assets (images, example data)
 - `package.json` — scripts & dependencies

 ## Troubleshooting
 - If `npm run dev` fails with a syntax/import error referencing
`node:fs/promises` or similar, your Node is likely too old (use Node
>=18). Install/activate an LTS Node via `nvm` and retry.
 - If you see dependency errors, try removing `node_modules` and the
lockfile and reinstalling.
 - If SPA routes return 404 on refresh in production, serve with a
fallback to `index.html` (the repo adds `404.html` for gh-pages).

 Logs and common commands
 - View vite output: the dev server prints the local URL to stdout when
running `npm run dev`.
 - Rebuild cache: `rm -rf node_modules && npm ci`

 ## Recommended next steps for contributors
 - Add `.nvmrc` with the Node version (e.g. `20`) for consistency.
 - Add `CONTRIBUTING.md` describing local data usage and viewer tests.
 - Add CI: GitHub Actions to run `npm ci`, `npm run lint`, and `npm run
~~~~

PROF

```
build` on PRs.

 If you'd like, I can add `.nvmrc` and a simple GitHub Actions workflow
next.

 ## Appendix: Original Vite template notes

 For reference, this project was started from the official Vite React
template. The original template documentation is retained here for
contributors who want the upstream guidance.

 ### React + Vite (from template)

 This template provides a minimal setup to get React working in Vite
with HMR and some ESLint rules.

 Currently, two official plugins are available:

 - [@vitejs/plugin-react](https://github.com/vitejs/vite-plugin-
react/blob/main/packages/plugin-react/README.md) uses [Babel]
(https://babeljs.io/) for Fast Refresh
 - [@vitejs/plugin-react-swc](https://github.com/vitejs/vite-plugin-
react-swc) uses [SWC](https://swc.rs/) for Fast Refresh

 ### Expanding the ESLint configuration

 If you are developing a production application, we recommend using
TypeScript and enable type-aware lint rules. Check out the [TS template]
(https://github.com/vitejs/vite/tree/main/packages/create-vite/template-
react-ts) to integrate TypeScript and [`typescript-eslint`]
(https://typescript-eslint.io) in your project.
```