ChatGPT

Book Publishing Pipeline

This repository contains the source files and build system for a Markdown-first book. The goal is to treat the text and layout as code so that you can collaborate via pull requests, run automated tests on your chapters, and build reproducible releases in multiple formats (HTML, EPUB and PDF).

Directory structure

```
book-repo/
├── book/              # Markdown chapters and assets
│   ├── 00-front-matter.md
│   ├── 1-chapter-1.md
│   └── styles/        # CSS styles for HTML/EPUB builds
│       ├── book.css
│       └── fonts.css
├── metadata/          # Book metadata for Pandoc (title, author, ISBN, etc.)
│   └── metadata.yaml
├── .github/
│   └── workflows/     # GitHub Actions workflow for tests and builds
│       └── build.yml
├── Makefile           # Local build commands (make html, make epub, make pdf)
└── README.md          # This file
```

The `build/` directory is created during builds and ignored via `.gitignore`.

Getting started

1. **Install dependencies**:
2. Pandoc – used to convert Markdown to HTML/EPUB/PDF.
3. A PDF engine such as `xelatex` or `wkhtmltopdf` depending on your preference.
4. Node.js is *not* required unless you modify the GitHub Actions or use additional tooling.
5. **Clone this repository** and install any required linters (`markdownlint`, etc.) if you plan to run tests locally.
6. **Build the book locally**. From the repository root run:

```
make        # builds html, epub and pdf into the build/ directory
make clean  # removes the build/ directory
```

The Makefile uses Pandoc with the metadata and CSS to generate formatted output. You can override variables such as `EPUB_ENGINE` or supply additional arguments in the Makefile if needed.

1. **Preview the HTML** by opening `build/book.html` in your browser. EPUB and PDF files are also placed in `build/`.

Continuous integration

The project includes a GitHub Actions workflow (`.github/workflows/build.yml`) that runs on every push to the `main` branch. It performs a few tasks:

1. **Lint Markdown** using `markdownlint` to enforce consistent formatting and structure.
2. **Check for broken links** with Lychee.
3. **Smoke test Pandoc conversion** by building HTML. If this step fails the pull request is blocked.
4. **Build all formats** (HTML, EPUB, PDF) once tests pass and upload them as artifacts on the workflow run.

Pull requests to `main` will trigger the same workflow, allowing maintainers to review the rendered output before merging. You can extend or modify the workflow to run additional linters, spell checkers or style guides.

Writing guidelines

- Write your chapters in individual Markdown files under `book/`. Files are ordered lexicographically when building; prefix chapter filenames with numbers (e.g. `01-introduction.md`, `02-background.md`) to control sequence.
- Use semantic Markdown headers (`#`, `##`, `###`, …) and avoid manually numbering headings – Pandoc will generate a table of contents if required.
- Keep lines short (80–100 characters) to make diffs easier to review. A `.markdownlint.json` file can be added to customise rules.
- Place images in a subdirectory (e.g. `book/images/`) and reference them with relative paths in your Markdown.

Styling and fonts

The CSS in `book/styles/book.css` controls the appearance of the HTML and EPUB output. It imports custom fonts defined in `fonts.css` and sets typography, spacing and colours. You can override or extend these styles to suit your design – however, be aware that PDF output using LaTeX may not respect all CSS settings.

If you add new fonts, place them in `book/fonts/` and declare them in `fonts.css` using `@font-face`. For example:

```
@font-face {
  font-family: "Inter";
```

```
  src: url("../fonts/Inter-Regular.woff2") format("woff2");
  font-weight: normal;
  font-style: normal;
}
```

Contributing

1. Create a feature branch off of `main` (e.g. `feature/chapter-2`).
2. Write or edit Markdown files, update styles or metadata as needed.
3. Run `make` locally to ensure the book builds without errors.
4. Commit your changes and open a pull request. The CI workflow will run automatically.
5. Once your pull request is approved and merged, the GitHub Actions workflow on `main` will produce the release artifacts.

License

Specify your licence here (e.g. MIT, CC BY-SA 4.0). Replace this section with the appropriate text or include a separate `LICENSE` file in the repository.

Acknowledgements

This workflow uses open-source tools like Pandoc, GitHub Actions, and Lychee. Thanks to the contributors of these projects for enabling a streamlined publishing pipeline.
