Learn to learn

This is about *how* to learn, because web development means you have to be willing to keep learning.

- Tools
  - Anki: A Practical Guide to the Spaced-Repetition Powerhouse □□□□
  - Nextcloud: Your Private Cloud—On Your Terms ☁□□
  - Typora: A Polished, Distraction-Light Markdown Editor ✍
- BookStack
  - BookStack: A Clear, Structured Wiki for Teams □□
  - Install your own BookStack instance
  - Backup + Migration
  - Formeln & Diagramme
  - art10m's BookStack Hacks

Tools

Anki, Nextcloud, BookStack, Typora, LLMs, etc.

Tools

Anki: A Practical Guide to the Spaced-Repetition Powerhouse □□□

Anki is a study application built around a simple but *highly effective* idea: **you remember more when you review information right before you're about to forget it**. This approach—called **spaced repetition**—turns studying from a vague "read it again" routine into a system that *actively manages your memory over time*.

Used by language learners, medical students, programmers, and lifelong learners alike, Anki is especially good at helping you retain large amounts of information reliably—without endlessly re-reading notes.

What Anki *is* (and what it isn't)

At its core, Anki is a **flashcard program**, but it's not the kind where you cram a deck the night before and hope for the best. Instead, Anki:

1. **Schedules reviews automatically** based on how well you remember each card.
2. **Prioritizes difficult material** so you don't waste time on what you already know.
3. **Builds long-term retention** through repeated recall over expanding intervals.

What Anki *isn't*:

- It's not a replacement for **understanding**. You still need to learn concepts first.
- It's not ideal for everything (e.g., deep essay writing practice or open-ended problem solving), though it can support those areas.

The core idea: spaced repetition (in plain terms)

When you learn something, your memory of it decays over time. If you review too soon, you waste time. If you review too late, you forget and have to relearn.

Anki aims for the sweet spot by adapting the review schedule:

1. You see a card.
2. You try to answer from memory.
3. You rate how well you remembered (commonly **Again** / **Hard** / **Good** / **Easy**).
4. Anki uses that feedback to decide **when you should see it next**.

Over time, easy cards show up less often, while hard cards appear more frequently until they stabilize in memory.

How Anki is structured: decks, notes, and cards

One of Anki's most important (and initially confusing) distinctions is:

Decks

- A **deck** is a collection of cards—like a folder.
- You might have decks such as:
  1. *Spanish*
  2. *Biology*
  3. *Interview Prep*

Notes vs. Cards

- A **note** is a piece of information you enter (like a template-filled record).
- A **card** is what you actually review.

A single note can generate **multiple cards**. For example, in language learning:

1. Note:
   - Front field: *“to eat”*
   - Back field: *“comer”*
2. Cards generated:
   - Card A: *to eat → comer*
   - Card B: *comer → to eat*

This “one note, many cards” model is one reason Anki scales so well for complex learning.

Card types: more than basic Q&A

Anki supports many ways of prompting recall. The most common are:

1. **Basic (Front → Back)**
   - Great for simple facts and definitions.
2. **Basic (and reversed card)**
   - Useful for bidirectional pairs (word ↔ translation).
3. **Cloze deletion** (fill-in-the-blank)
   - Example: “The capital of France is **{{c1::Paris}}**.”
   - Excellent for learning within context, especially for sentences, processes, and structured facts.
4. **Image occlusion** (via add-on)
   - Hide parts of an image (e.g., anatomy diagrams, maps) and recall what’s covered.

Why Anki works well: active recall + scheduling □□

Anki’s effectiveness comes from combining two evidence-aligned strategies:

1. **Active recall**
   - You *produce* the answer from memory, which strengthens retrieval pathways.
2. **Spaced repetition**
   - You revisit information at increasing intervals, strengthening long-term memory efficiently.

Together, they often outperform passive review methods such as rereading or highlighting—especially when your goal is *durable retention*.

Best practices: how to make Anki truly effective

Many people try Anki, bounce off, and assume it's "not for them." Most of the time, the issue is **card design** or **workflow**, not the tool itself. These practices help a lot:

1) Keep cards atomic (one idea per card)

- Avoid: "List *all* causes of X" (too broad)
- Prefer:
  1. "What is one cause of X?"
  2. "Which cause of X relates to Y?"

Atomic cards reduce mental overload and improve accuracy.

2) Use clear prompts and unambiguous answers

- If you often think "Well… it depends," the card is likely too vague.
- Add context, specify constraints, or split into smaller cards.

3) Make cards meaningful, not just copy-pasted

- Better than memorizing a paragraph is memorizing:
  1. Key definitions
  2. Steps in a process
  3. Important distinctions
  4. Minimal examples

4) Add *just enough* context

- Context reduces “false familiarity.”
- Example: instead of memorizing “Polymorphism,” prompt with “In OOP, what is polymorphism?”

5) Build a sustainable daily habit

- The magic is consistency. Even 15–25 minutes a day can accumulate huge gains.
- If you skip many days, reviews pile up and can feel overwhelming.

The scheduling knobs: due cards, new cards, and limits ⚙

Anki typically shows you:

1. **Due reviews**
   - Cards you’ve seen before and are scheduled to return.
2. **New cards**
   - Fresh material you’re introducing.

You can control pacing by adjusting:

1. **New cards/day**
2. **Maximum reviews/day**
3. **Learning steps** (short intervals for new cards)
4. **Graduating interval** (when a card becomes “mature”)

A practical approach is to **cap new cards** so your future review load stays manageable. If you add too many new cards too quickly, the system will “collect interest” in the form of heavy daily reviews later.

Syncing and platforms: where Anki runs □□□

Anki is available across devices, usually via:

1. **Anki (desktop)** for Windows/macOS/Linux
2. **AnkiWeb** (browser-based review and syncing)
3. **Mobile apps**
   - iOS app is commonly a paid official app
   - Android has widely used options (often free)

The typical setup is:

1. Create/edit primarily on desktop (faster typing, better editing).
2. Review on mobile for convenience.
3. Sync through AnkiWeb to keep everything consistent.

Add-ons, customization, and power-user features □□

One reason Anki has such a loyal following is its flexibility:

1. **Add-ons**
   - Extend features (image occlusion, enhanced statistics, editor tools, etc.).
2. **Card templates**
   - Customize formatting with HTML/CSS for cleaner, more readable cards.
3. **Tags**
   - Organize cards across decks, filter study sessions, track sources.
4. **Filtered decks**
   - Create targeted review sessions (e.g., “all cards tagged *exam1* due this week”).

This power comes with a caution: customization is helpful, but it’s easy to spend more time tweaking than studying. A good rule is to *optimize only when a friction point repeats*.

Common pitfalls (and how to avoid them) □□

1. **Overstuffed cards**
   - Fix: split cards; use cloze deletions for structure.

2. **Memorizing without understanding**
   - Fix: learn the concept first; then Anki it.
3. **Too many new cards too fast**
   - Fix: reduce new/day; prioritize review consistency.
4. **Perfectionism**
   - Fix: accept occasional “Again”; memory training is iterative.
5. **Using shared decks blindly**
   - Fix: audit them; edit for clarity and alignment with your course or goals.

Who benefits most from Anki? □□

Anki shines when you need **reliable recall** over time—especially for:

1. **Languages**
   - Vocabulary, sentences, grammar patterns, listening prompts.
2. **Medicine & health sciences**
   - Anatomy, pharmacology, diagnosis criteria, clinical facts.
3. **STEM**
   - Definitions, formulas (with meaning), units, key theorems, step prompts.
4. **Professional knowledge**
   - Interviews, certifications, legal principles, product knowledge.
5. **Anything with lots of “must-remember” details**
   - Names, dates, terminology, procedures.

A simple, effective workflow to start □□

If you’re new, keep it straightforward:

1. **Choose one deck**
   - Start small: one subject or one course.
2. **Set a modest new-card limit**
   - Something you can sustain daily.
3. **Make a few high-quality cards per study session**
   - Focus on clarity and one idea per card.
4. **Review every day**
   - Even if it’s short—consistency matters most.
5. **Refine cards that fail repeatedly**

- Repeated failures usually mean the card is poorly designed, not that you're "bad at memory."

Closing thoughts: Anki as a long-term learning system □□

Anki is most powerful when you treat it less like a flashcard app and more like a **personal memory schedule**. It rewards clarity, consistency, and good card design—helping you convert short-term learning into long-term knowledge with impressive efficiency.

If you tell me *what you're studying* (e.g., Japanese, anatomy, programming interviews), I can suggest **specific card types**, **settings**, and **example cards** tailored to your goal.

Tools

Nextcloud: Your Private Cloud—On Your Terms ☁

Nextcloud is a **self-hosted collaboration and file platform** that gives you many of the conveniences of mainstream cloud services—file syncing, sharing, online documents, calendars, chat, and more—while keeping **control of your data** in your own hands. Instead of handing your files and activity metadata to a third party, you can run Nextcloud on **your own server**, a trusted hosting provider, or within your organization's infrastructure.

It's popular with **privacy-conscious individuals**, **families**, **schools**, **nonprofits**, and **enterprises** that want the flexibility of modern cloud workflows without giving up governance, compliance, or customization.

What Nextcloud *is* (and what it isn't)

At its core, Nextcloud is a **web application** (written primarily in PHP) that you deploy on a server alongside a database and storage. Users access it via:

- A **web interface** (browser)
- **Desktop clients** (Windows, macOS, Linux)
- **Mobile apps** (iOS, Android)
- Standard protocols like **WebDAV** for broader compatibility

It *is*:

1. A **file sync and share** system with rich collaboration features
2. A **platform** with an "app store" model—many capabilities are add-ons
3. A **hub** for productivity tools (docs, talk, mail, groupware, etc.)
4. A **self-managed** alternative to services like Google Drive, Dropbox, and Microsoft 365 (depending on your app choices)

It *isn’t*:

1. A “set it and forget it” SaaS product—**you manage updates, backups, and uptime**
2. Automatically zero-knowledge by default (you can *add* end-to-end encryption features, but you’ll want to understand the tradeoffs)

The core experience: Files, sync, sharing □□□

Nextcloud Files is the foundation most people start with.

Key features you’ll likely use immediately

1. **Sync across devices**
   - Desktop clients keep folders in sync, much like Dropbox.
   - Selective sync helps control disk usage.
2. **Sharing controls**
   - Share with internal users or create **public links**.
   - Optional link protections:
     - **Password protection**
     - **Expiration dates**
     - **Read-only vs. edit permissions**
     - Upload-only “file drop” style shares
3. **Versioning and recycle bin**
   - Previous file versions can be restored after accidental edits.
   - Deleted files can be recovered (policy-dependent).
4. **Activity tracking**
   - Audit-friendly logs show file activity (who changed what, when) □□

Collaboration and productivity: Beyond “just storage” □□□

Nextcloud becomes especially compelling when you treat it as a collaboration hub rather than only a file server.

Common “hub” capabilities

1. **Nextcloud Office integration** □□
   - Work on documents, spreadsheets, and presentations in the browser.
   - Typically powered by **Collabora Online** or **OnlyOffice**.
   - Supports real-time collaboration, comments, and basic workflows.
2. **Talk (chat, audio/video calls)** □□□□
   - Team chat, group calls, screen sharing (feature set can vary by deployment).
   - Useful for internal communication without relying on external chat platforms.
3. **Calendar, Contacts, Tasks** □□
   - CalDAV/CardDAV support for broad compatibility.
   - Plays well with phones and desktop clients that support those standards.
4. **Mail**
   - Optional webmail-like experience (often best for smaller deployments or specific workflows).

Privacy, security, and governance □□□□

Nextcloud’s biggest advantage is **control**: where data lives, who can access it, how it’s retained, and how it’s audited.

Practical security building blocks

1. **Transport security (TLS/HTTPS)**
   - Encrypts traffic between users and your server.
2. **Encryption at rest**
   - Server-side encryption options exist, but require careful key management.
   - Many admins prefer full-disk encryption + strong access controls.
3. **End-to-end encryption (E2EE)**
   - Available for certain use cases, but not always compatible with every workflow (e.g., web previews, server-side indexing, some collaboration features).
4. **Access controls**
   - Strong password policies, **2FA**, app passwords.
   - Group-based permissions and sharing restrictions.

5. **Auditing and compliance**
   - Logging and retention policies can support regulated environments.
   - Enterprise deployments often integrate with SIEM and identity providers.

Extensibility: Apps and integrations □□□

One of Nextcloud’s most powerful traits is its **modular ecosystem**. You can keep it lightweight or build it into a full internal platform.

Examples of useful extensions

1. **External storage backends**
   - Connect S3-compatible storage, SMB shares, or other backends.
   - Lets you unify multiple storage sources behind one interface.
2. **Identity and SSO**
   - LDAP/Active Directory integration
   - SAML/OIDC via plugins (depending on your setup)
3. **Automation and workflows**
   - File tagging, approval flows, notifications, and policies.
   - Useful for document-heavy organizations.
4. **Security add-ons**
   - Brute-force protection, advanced auditing, device management (varies by edition and configuration).

Hosting options: From home lab to enterprise □□□

You can run Nextcloud in multiple ways, and the “best” choice depends on your time, budget, and reliability needs.

Common deployment paths

1. **Home / personal**
   - Small server, NAS, or mini PC.
   - Great for backups, photos, and personal collaboration.
   - You'll want reliable storage and backups.
2. **VPS / dedicated hosting**
   - Easier uptime and bandwidth than a home connection.
   - A good middle ground for individuals and small teams.
3. **Enterprise / on-prem**
   - Suited to compliance, internal networks, and performance tuning.
   - Often paired with load balancing, object storage, and centralized identity.

> Tip: Many issues people attribute to "Nextcloud being slow" are actually **storage choices**, **database tuning**, **PHP/OPcache settings**, or **insufficient RAM/IOPS** rather than Nextcloud itself.

Performance and reliability: What matters most ⚙□□

For a smooth experience, focus on the fundamentals:

1. **Storage performance**
   - SSDs (or fast network storage) matter a lot for responsiveness.
   - Avoid slow, heavily contended disks for multi-user setups.
2. **Database health**
   - PostgreSQL or MariaDB/MySQL are common choices.
   - Regular maintenance and proper indexing help at scale.
3. **Caching**
   - Memory caching (like Redis) is frequently recommended for better locking and responsiveness.
4. **Background jobs**
   - Cron-based background tasks tend to be more reliable than "AJAX" mode.
5. **Backups**
   - Plan backups for:
     - Files
     - Database
     - Config
   - Test restores—*not just backups* □

Who should consider Nextcloud? □□

Nextcloud is a great fit if you:

1. **Care about privacy and data sovereignty**
   - You decide where data lives and who can access it.
2. **Want customization**
   - Branding, policies, workflows, and integrations can be tailored.
3. **Need collaboration without vendor lock-in**
   - Open standards and flexible storage options reduce lock-in risk.
4. **Operate in regulated environments**
   - Helpful where governance, auditing, and retention policies matter.

It may be less ideal if you need:

1. A fully managed experience with zero admin overhead
2. Guaranteed features identical to Google/Microsoft ecosystems (some capabilities depend on add-ons and tuning)

A practical “getting started” roadmap □□□

If you’re evaluating Nextcloud, this staged approach reduces friction:

1. **Start with a small pilot**
   - 1–5 users, basic file sync, sharing, and mobile access.
   - Verify upload/download speed and reliability.
2. **Add collaboration**
   - Integrate an office suite (Collabora or OnlyOffice).
   - Enable Calendar/Contacts and try cross-device syncing.
3. **Harden security**
   - Enforce HTTPS, enable 2FA, restrict public sharing where appropriate.
   - Establish backups and test restore procedures.
4. **Scale thoughtfully**
   - Add caching (e.g., Redis), optimize database and storage.
   - Consider external object storage for large datasets.

Final thoughts □□

Nextcloud shines as a **flexible, self-controlled cloud platform**: it can be a simple personal file sync service or a robust collaboration suite for an entire organization. The tradeoff is that you—or your provider—must handle the operational side: updates, monitoring, backups, and performance tuning. If you're willing to take on (or outsource) that responsibility, Nextcloud offers an unusually powerful mix of **privacy, extensibility, and ownership** in a world where those qualities are increasingly rare.

If you tell me your intended use—*personal*, *family photos*, *small business*, or *enterprise*—and whether you prefer **home hosting** or a **VPS**, I can suggest a recommended setup (including storage, database choice, and a sensible app bundle) tailored to your needs.

Tools

Typora: A Polished, Distraction-Light Markdown Editor ✍

Typora is a **Markdown editor** that aims to make writing feel *as close as possible to reading*. Instead of showing a split view (raw Markdown on the left, preview on the right), Typora uses a **single, live-rendered writing surface**—so formatting appears *as you type*, without constantly switching mental modes.

It's especially popular among people who write documentation, notes, technical content, and long-form articles—anywhere Markdown shines for its balance of **speed, structure, and portability**.

What Makes Typora Different (and Why People Like It)

Most Markdown editors fall into one of two camps:

1. **Source-first editors**
   - You mostly see Markdown syntax (`#`, `*`, backticks, etc.).
2. **Preview-first editors**
   - You write Markdown in one place and preview in another.

Typora's approach is a third option:

1. **Seamless "What You See Is What You Mean"**
   - You still *write* Markdown.
   - But you *see* the result immediately, inline, on the same page.
   - The goal is fewer interruptions—your eyes and attention stay on the content.

This is the core of Typora's appeal: it feels like a clean writing app, but produces **standard Markdown**.

Core Writing Experience □□

Typora is designed to keep writing friction low while still giving you strong structure.

Inline Formatting (without the fuss)

You can use normal Markdown conventions—bold, italics, links, inline code—while Typora renders the result immediately. It tends to “get out of the way,” so you can focus on the sentence you’re shaping, not the syntax you’re managing.

Headings, Lists, and Tables

Typora makes structured writing feel natural:

1. **Headings**
   - Easy to create and visually obvious, which encourages good document structure.
2. **Lists**
   - Ordered and unordered lists render cleanly and are easy to edit without fighting indentation.
3. **Tables**
   - Tables are a notorious pain in raw Markdown.
   - Typora’s table editing is one of those quality-of-life features that makes Markdown feel “grown up.” □□

Code Blocks and Syntax Highlighting

For technical writing, Typora supports fenced code blocks and typically provides **syntax highlighting** when you specify a language. That makes it a strong choice for:

1. README files
2. Tutorials and guides
3. API notes
4. Internal engineering docs

Navigation and Organization □□

As documents get longer, navigation matters.

1. **Outline / document structure**
   - Typora can show an outline based on headings, letting you jump around quickly.
2. **Readable long-form layout**
   - The editor tends to keep typography comfortable, which helps with long sessions (notes, essays, spec docs).

If your writing style includes frequent reorganizing—moving sections, renaming headings, tightening flow—the outline-centric workflow is a big win.

Themes and Visual Polish □□

Typora supports **themes** that change typography, spacing, and overall look—useful both for personal preference and for matching the tone of the content (e.g., minimal notes vs. formal documentation).

A good theme can improve:

1. **Readability**
   - Better font choices and spacing reduce fatigue.
2. **Scanning**
   - Clear hierarchy makes headings, quotes, and code blocks easier to parse.
3. **Export appearance**
   - Many people like their editor to resemble the final output.

Export and Sharing □□

Markdown is great because it’s *portable*, but people often need to share content as something else. Typora is commonly used in workflows where the same source document needs to become:

1. **PDF**
2. **HTML**
3. **Word-like formats** (depending on setup and available exporters)

This makes Typora useful when you want to keep a single Markdown source of truth while distributing in more “universal” formats.

Who Typora Is For (and Who It Might Not Be For) □□

Typora is a strong fit if you:

1. **Write in Markdown regularly**
   - Documentation, notes, knowledge bases, blogging drafts, coursework.
2. **Prefer a clean, unified editing surface**
   - You don’t want split panes or constant preview toggling.
3. **Care about readability while writing**
   - You want the writing view to feel like the final document.

Typora might not be ideal if you:

1. **Want heavy IDE-style features**
   - Deep refactoring tools, project-wide symbol navigation, or language-server-level features are typically better in code editors.
2. **Need advanced knowledge-base features built-in**
   - Some note systems emphasize backlinks, graph views, and database-like organization; Typora is more document-centric.

Practical Ways People Use Typora □

Here are common real-world workflows:

1. **README + docs authoring**
   - Draft in Typora, commit Markdown to Git, export PDF for stakeholders when needed.
2. **Personal notes with structure**
   - Use headings + checklists + code snippets + tables for clean, searchable notes.
3. **Technical blogging**
   - Write Markdown, preview typography with a theme, export HTML or paste into a CMS.
4. **Meeting notes and specs**
   - Fast formatting, consistent structure, easy section rearrangement.

Tips for Getting the Most Out of Typora □□

1. **Adopt a consistent heading hierarchy**
   - Use `#` for title, `##` for main sections, `###` for subsections, etc.
2. **Use checklists for actionable notes**
   - Great for meeting follow-ups and personal task capture.
3. **Lean on tables sparingly**
   - They’re great for comparisons and matrices, but headings + bullet points are often more readable.
4. **Pick a theme that matches your intent**
   - *Minimal* for drafting, *print-like* for publishing.

The Bigger Picture: Why Typora Works □□

Typora succeeds because it treats Markdown as a **writing medium**, not merely a markup language. Its live-rendered interface reduces the friction between *drafting* and *formatting*, which can noticeably improve flow—especially for long documents and structured notes.

If you like Markdown but wish it felt less “code-like” while writing, Typora is one of the most elegant interpretations of that idea.

If you tell me what you want to use Typora for (e.g., *software documentation*, *note-taking*, *blogging*, *academic writing*), I can tailor a recommended setup—theme style, export approach, and a simple folder structure—so it fits your workflow.

BookStack

What is BookStack? How do I install it?

BookStack: A Clear, Structured Wiki for Teams □□

BookStack is an open-source platform for creating and organizing documentation—think *internal knowledge base*, *team wiki*, or *product docs portal*—with a structure that's intentionally familiar: **Books → Chapters → Pages**. That simple model (paired with a clean editor and solid permissions) makes it especially appealing for teams that want documentation to be *easy to write*, *easy to navigate*, and *easy to govern*.

What BookStack is (and why people like it) □

BookStack is designed to reduce the friction that often comes with documentation tools. Instead of treating knowledge as an unstructured pile of pages, it encourages a **hierarchy** that matches how many teams already think:

1. **Books**
   - Top-level collections (e.g., *Engineering Handbook*, *IT Runbooks*, *Customer Support Playbook*).
2. **Chapters**
   - Subsections within a book (e.g., *Onboarding*, *Deployments*, *Incident Response*).
3. **Pages**
   - The actual documentation content (procedures, how-tos, references, policies).

This hierarchy helps readers quickly orient themselves, and it helps authors avoid "where do I put this?" paralysis. □□

Core features that matter in practice □□

1) Editing experience (built for real-world docs)

BookStack provides a modern, approachable authoring workflow:

1. **WYSIWYG editor** (commonly preferred for non-technical contributors)
   - Great for teams where everyone—from engineers to operations to support—needs to contribute.
2. **Markdown support**
   - Useful for technical teams who prefer writing in plain text with predictable formatting.
3. **Rich content tools**
   - Easy inclusion of images, links, tables, and code blocks.
4. **Page revision history**
   - Helps you see what changed and when, which is crucial for controlled documentation.

2) Organization and navigation

BookStack's structure doesn't just help authors—it improves consumption:

1. **Predictable browsing**
   - Readers can move from a *Book* down to *Chapters* and *Pages* without hunting.
2. **Search**
   - Fast searching becomes more valuable when paired with consistent structure and naming.
3. **Cross-linking**
   - Pages can reference other pages to build a connected knowledge graph without losing hierarchy.

3) Permissions and access control □□

Governance is where many wikis struggle. BookStack typically shines here:

1. **Role-based access**
   - Control who can view, create, edit, or delete content.
2. **Granular permissions**
   - Apply restrictions at different levels (e.g., specific books or content areas).
3. **Team-friendly collaboration**
   - Helps keep sensitive runbooks or HR policies restricted while leaving general knowledge open.

4) Authentication options (fits into existing identity stacks)

Depending on your setup and version, BookStack can integrate with common authentication approaches to reduce account sprawl and simplify onboarding/offboarding:

1. **Local authentication**
2. **LDAP/Active Directory**
3. **SSO-style integrations** (commonly via SAML/OAuth-like approaches in organizational environments)

(Exact availability can depend on how you deploy and configure it.)

5) Media and attachment handling □□

Docs rarely live as pure text:

1. **Image management**
   - Useful for diagrams, screenshots, and annotated procedures.
2. **File attachments**
   - Handy for templates, exports, and reference files—though many teams prefer linking to a source of truth (like Git) for certain assets.

Typical use cases (where BookStack fits best) □□

BookStack is broadly useful, but it’s especially strong when your team values clarity and structure.

1. **Internal team wiki**

- Decision logs, meeting notes, standards, and best practices.

2. **IT & Ops runbooks**
   - Incident response steps, on-call procedures, system recovery guides.
3. **Engineering documentation**
   - Architecture overviews, onboarding guides, deployment instructions.
4. **Support & customer success playbooks**
   - Troubleshooting flows, known issues, escalation processes.
5. **Policy and compliance documentation**
   - Controlled edits, auditable changes, restricted sections.

Strengths and trade-offs ⚖

Strengths

1. **Strong information architecture**
   - The Books/Chapters/Pages model makes messy knowledge more navigable.
2. **Low barrier to contribution**
   - Non-technical users often feel comfortable quickly.
3. **Practical permissions**
   - Good for teams that need structure *and* control.
4. **Self-host friendly**
   - Ideal for organizations that prefer keeping data on their own infrastructure.

Trade-offs to consider

1. **Hierarchy can be limiting for some knowledge styles**
   - If your team prefers a purely tag-driven, graph-like, or database-like knowledge system, you may feel constrained.
2. **Not a full doc-as-code pipeline**
   - While Markdown exists, BookStack isn’t primarily designed to be a Git-native docs workflow in the way some static-site generators are.
3. **Customization**
   - You can brand and configure it, but extreme customization may require deeper technical effort and ongoing maintenance.

How teams keep BookStack content high-quality ✍

A tool helps, but process makes it stick. Common patterns that work well:

1. **Documentation templates**
   - For recurring page types (runbooks, how-tos, policies).
2. **Naming conventions**
   - Consistent titles improve search and scanning (e.g., “How to …”, “Runbook: …”, “Policy: …”).
3. **Ownership and review cadence**
   - Assign a “page owner” or “book maintainer” and review quarterly or after major changes.
4. **Link to sources of truth**
   - Reference tickets, diagrams, repos, or monitoring dashboards rather than duplicating volatile data.
5. **Use permissions to reduce accidental edits**
   - Keep “official” procedures protected while allowing contributions in draft areas.

Deployment and operations overview □□

BookStack is commonly deployed in a web-app style environment with a database backend.

1. **Containerized deployment**
   - Many teams run it via Docker for predictable setup and upgrades.
2. **Backups**
   - Plan backups for both:
     1. The **database** (core content, users, settings)
     2. The **uploaded files** (images/attachments)
3. **Upgrades**
   - Regular updates help with security patches and new features; test in staging if possible.
4. **Performance**
   - For most teams, default performance is solid; larger orgs may tune caching and database resources.

Who should choose BookStack? □□

BookStack is a great choice if you want:

1. A **straightforward wiki** that’s easy to navigate
2. A **structured documentation hierarchy**
3. **Permissions** that can match real organizational needs
4. A **self-hostable** solution with a clean UI

If your top priority is a **Git-first** docs workflow with automated builds, PR reviews, and versioned docs tied tightly to code releases, you might instead prefer a doc-as-code toolchain—though many teams still use BookStack effectively alongside those systems (e.g., BookStack for runbooks and onboarding, Git for developer reference docs).

If you tell me your context, I can tailor it □□

If you share a bit about your situation, I can adapt the article into a recommendation or a deployment plan:

1. **Team size** and who will author docs (engineers only vs. cross-functional)
2. Whether you need **SSO/LDAP**
3. Whether this is **internal-only** or partially public
4. Your preferred hosting approach (Docker, VM, Kubernetes, etc.)
5. Your documentation style (runbooks, policies, product docs, onboarding, etc.)

Install your own BookStack instance

> This is the way this instance of BookStack is installed.

> This guide starts **from the point** where you already have a **Linux server** and **Docker** is installed. The “steps before that” will follow later — it’s really not hard (especially with AI help).

Overview: What’s being set up here?

In the end you’ll have:

- **BookStack** (wiki/docs system)
- **MariaDB** as the database
- **Caddy** as a reverse proxy with **automatic HTTPS** (Let’s Encrypt) □□

Background: What is Caddy? □□

> **Caddy** is a modern web server (similar to *Nginx*/*Apache*), written in *Go*, that embraces “*security-by-default*” and, above all, makes **HTTPS** extremely convenient.

Core idea: “A web server that just works” □

Key features

1. **Automatic HTTPS (TLS)**
   - Certificates are obtained and renewed automatically (typically via *Let's Encrypt*).
2. **Simple configuration**
   - Via an easy-to-read **Caddyfile**.
3. **Reverse proxy & load balancing**
   - Ideal for forwarding requests to services (e.g., Docker containers).
4. **Good defaults**
   - Many sensible security standards are enabled by default.
5. **Modular extensibility**
   - If you have special requirements, Caddy can be extended.

Typical use cases

- Hosting websites (static/dynamic)
- Reverse proxy in front of an app (e.g., `/api` → backend)
- TLS termination
- Local dev setups with HTTPS

Step 1: Prepare directory & files

Create the directory:

- `/opt/bookstack`

Create **two files** inside it:

- `docker-compose.yml`
- `Caddyfile`

> ✍ **Important:** You'll need to adjust a few values in a moment — you can clearly see *where* in the YAML.

Step 2: Configure Docker Compose (`docker-compose.yml`)

Paste the following content (and adjust it where necessary):

```
services:
  mariadb:
    image: lscr.io/linuxserver/mariadb:latest
    container_name: bookstack-mariadb
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Europe/Berlin
      - MYSQL_ROOT_PASSWORD=PW_OF_MYSQL_ROOT
      - MYSQL_DATABASE=bookstack
      - MYSQL_USER=bookstack
      - MYSQL_PASSWORD=PW_OF_MYSQL_DB
    volumes:
      - ./mariadb:/config
    restart: unless-stopped

  bookstack:
    image: lscr.io/linuxserver/bookstack:latest
    container_name: bookstack
    depends_on:
      - mariadb
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Europe/Berlin

      - APP_URL=https://wiki.fabula.vision
      # Where will BookStack be accessible? (Your URL)
      - APP_KEY=base64:...
      # To generate it, run: docker run -it --rm --entrypoint /bin/bash
lscr.io/linuxserver/bookstack:latest appkey
      - APP_THEME=custom
      #'custom' makes it possible to use hacks; more on that here:
https://www.bookstackapp.com/hacks/applying/

      - DB_HOST=mariadb
      - DB_PORT=3306
      - DB_DATABASE=bookstack
```

```
      - DB_USERNAME=bookstack
      - DB_PASSWORD=PW_OF_MYSQL_DB
      # (As above!)

      - APP_DEFAULT_DARK_MODE=true
      # (Personal preference)

    volumes:
      - ./bookstack:/config
    restart: unless-stopped

  caddy:
    image: caddy:latest
    container_name: caddy
    depends_on:
      - bookstack
    ports:
      - "80:80"
      - "443:443"
    volumes:
      - ./Caddyfile:/etc/caddy/Caddyfile:ro
      - ./caddy/data:/data
      - ./caddy/config:/config
    restart: unless-stopped
```

Step 3: Configure Caddy (`Caddyfile`) □□

Paste the following and replace the domain:

```
wiki.fabula.vision {
□# Your URL, of course...
□encode zstd gzip

□# Reverse proxy to BookStack (container is named "bookstack", internal port 80)
□reverse_proxy bookstack:80
```

```
    # sensible headers (optional)
    header {
        # HSTS (only set this if you're sure HTTPS should remain enabled permanently)
        Strict-Transport-Security "max-age=31536000; includeSubDomains; preload"
        X-Content-Type-Options "nosniff"
        X-Frame-Options "SAMEORIGIN"
        Referrer-Policy "strict-origin-when-cross-origin"
    }
}
```

Step 4: Start the containers ▶

1. Change into the directory:
   - `cd /opt/bookstack/`
2. Start the stack:
   - `docker compose up -d`

   Updating later works like this:
   - `docker compose pull`
   - `docker compose up -d`
3. Done
   Now quickly open the URL and change the default credentials!
   - Default login: `admin@admin.com`
   - Default password: `password`

Step 5: Cron job for BookStack

Create a cron job:

1. Open crontab:
   - `crontab -e`
2. Add the following line (cron every minute):
   - `* * * * * docker exec -i bookstack php /app/www/artisan schedule:run >/dev/null 2>&1`

Limitations & modules/hacks □□

There are three limitations that have stood out so far:

1. **Email sending**
   - (e.g., “Forgot password”) doesn’t work for me, and my experiments haven’t produced a solution yet.
   - I don’t strictly need it — but for example for comments (notifications) you can’t avoid it.
   - More info:

     https://www.bookstackapp.com/docs/admin/email-webhooks/#email-configuration
2. **Extending via “hacks”**
   - BookStack can be extended via so-called **“hacks.”**

     EDIT: I created my own hacks that are better!
   - Two recommendations:
     1. **MathJax/TeX** (math formulas):
        - https://www.bookstackapp.com/hacks/mathjax-tex/
     2. **Mermaid Viewer** (diagrams):
        - https://www.bookstackapp.com/hacks/mermaid-viewer/

   Important note about export □□
   - Mermaid (and formulas as well) **does not export cleanly** for me (e.g., PDF): code blocks appear instead of rendered content.

   How do you install hacks as modules?
   - We already did the first step: In the YML it says `APP_THEME=custom`
     This loads modules from the following path:
     - `/opt/bookstack/bookstack/www/themes/custom/modules`
   - Then via SSH:
     1. Switch into the container:
        - `docker exec -it bookstack /bin/bash`
     2. Go to the web directory:
        - `cd /app/www/`
     3. Run the command that’s listed in the respective hack (example *Mermaid*):
        - On the hack page in the “Install as Module” section

          https://www.bookstackapp.com/hacks/mermaid-viewer/
3. **Shift+Enter — line break (CommonMark)**
   - Due to the decision to use the CommonMark standard, a simple line break is not rendered by default.
   - Instead you need “two spaces at end of line + Enter” — that drove me crazy because many Markdown editors are used to *Shift+Enter*.
   - My fix consists of two parts:

1. In the BookStack settings under
   `https://example.com/settings/customization`
   under “Custom HTML Head Content”, enter the following code:

```
<script>
  window.addEventListener('editor-markdown::setup', event => {
    event.detail.markdownIt.set({breaks: true});
  });
</script>
```

2. Create a `functions.php` under:
   - `/opt/bookstack/bookstack/www/themes/custom/`

   With the following content:

```
<?php

use BookStack\Theming\ThemeEvents;
use BookStack\Facades\Theme;

Theme::listen(ThemeEvents::COMMONMARK_ENVIRONMENT_CONFIGURE, function
($environment) {
  $environment->mergeConfig([
    'renderer' => [
      'soft_break' => "<br>",
    ]
  ]);

  return $environment;
});
```

Additional QoL fixes (under “Customization”) □□

Configure here:

- `https://yoururl.com/settings/customization`

Just paste each one under “Custom HTML Head Content”:

1. **Blockquotes: no thick bottom border / clean spacing**
   - Cause: CSS (among other things, `<p>` with `margin-bottom`)
   - Fix (same spacing top/bottom + no scrollbars):

```
<style>
  /* 1) Last & first direct child element in the blockquote: margin */
  .content-wrap blockquote > :last-child {
    margin-bottom: .3em;
  }

  .content-wrap blockquote > :first-child {
    margin-top: .3em;
  }

  /* 2) Blockquote: no scrollbars */
  .content-wrap blockquote {
    overflow: visible;     /* default “no scrolling” */
    overflow-x: visible;
    overflow-y: visible;
  }
</style>
```

2. **Code blocks: soft wraps (no horizontal scrolling needed)**

```
<script>
window.addEventListener('library-cm6::pre-init', event => {
    const detail = event.detail;
    const config = detail.editorViewConfig;
    const EditorView = detail.libEditorView;

    if (detail.usage === 'content-code-block') {
        config.extensions.push(EditorView.lineWrapping);
    }
});
</script>
```

3. **Dark mode: lighter text (headings 70%, text 90%)**

```
<style>
html.dark-mode .page-content h1, html.dark-mode .page-content h2, html.dark-mode
.page-content h3, html.dark-mode .page-content h4, html.dark-mode .page-content h5,
html.dark-mode .page-content h6 {
```

```
    color: hsl(0 0% 70%);
  }
  html.dark-mode body {
    color: hsl(0 0% 90%);
  }
</style>
```

4. **Text size: main content 110%, headings 90%**

```
<style>
.page-content {
    zoom:1.1;
  }

  .page-content h1, .page-content h2, .page-content h3, .page-content h4, .page-
content h5, .page-content h6 {
    zoom:.9;
  }
</style>
```

Backup + Migration

> Not tested yet!

> “ I installed BookStack via Docker (as in the guide). Could you please explain how to create backups (full content backup) so that I can migrate the website if needed, e.g., to another server/domain? Here is BookStack’s official guide.

Full BookStack backup (Docker) – so that a move to *another server/domain* works

For a **complete, restorable backup**, you essentially need **two components**:

1. **Database** (all pages, books, users, permissions, settings …)
2. **Files/config** (uploads, attachments, `.env`/APP_KEY, possibly themes/modules)

Since you have the setup like in your guide (`/opt/bookstack` with `docker-compose.yml`, volumes `./mariadb` and `./bookstack`), this is pleasantly simple.

What exactly needs to be backed up?

1) Database (MariaDB)

- The content is in the DB container, and in your setup it’s persisted under:
  - `./mariadb` (as volume `./mariadb:/config`)
- **Cleanest approach**: *SQL dump* (portable, tolerant across versions)

2) Files (BookStack container volume)

- Persistent data is located under:
  - `./bookstack` (as volume `./bookstack:/config`)
- In it, especially important:
  - `.env` (including **APP_KEY**!)
  - uploads/attachments
  - possibly `www/themes/custom/...` (your hacks/modules)

3) Compose/Caddy (helpful for migration)

- `docker-compose.yml`
- `Caddyfile`
- Caddy data optional:
  - `./caddy/data` & `./caddy/config` (certificates/state).
    Not strictly required, but sometimes saves re-issuing/rate limits.

□□ Create a backup (recommended approach)

> Assumption: project is located in `/opt/bookstack` and the containers are named as in your YAML:
> `bookstack-mariadb` and `bookstack`.

Step 1: Create a backup folder

```
cd /opt/bookstack
mkdir -p backups
```

Step 2: Dump the database as SQL

```
docker exec -i bookstack-mariadb \
  mysqldump -u bookstack -pbookstack \
  --single-transaction --routines --triggers \
  bookstack > backups/bookstack-db-$(date +%F).sql
```

Important: Replace `-pbookstack` with your real password (from `MYSQL_PASSWORD`).

> Alternative (safer because the password won't end up in shell history):
> Omit `-p...`, then `mysqldump` will prompt interactively for the password:
>
> ```
> docker exec -it bookstack-mariadb mysqldump -u bookstack -p --single-transaction --routines --triggers bookstack > backups/bookstack-db-$(date +%F).sql
> ```

Step 3: Archive files/volumes

```
tar -czvf backups/bookstack-files-$(date +%F).tar.gz \
  docker-compose.yml Caddyfile \
  bookstack mariadb caddy
```

If you **don't** want to back up **Caddy**, you can do it leaner:

```
tar -czvf backups/bookstack-files-$(date +%F).tar.gz \
  docker-compose.yml Caddyfile \
  bookstack mariadb
```

Step 4: (Optional) Integrity check

```
ls -lh backups/
head -n 5 backups/bookstack-db-*.sql
tar -tzf backups/bookstack-files-*.tar.gz | head
```

□□ Restore (on a new server) – incl. domain/URL migration

Step 0: Prerequisites

- Docker + Docker Compose installed
- DNS of the new domain points to the new server (for HTTPS via Caddy)

Step 1: Copy the backup over

Pack on the **old** server or copy directly:

```
scp -r /opt/bookstack/backups user@new-server:/opt/bookstack/
```

Step 2: Restore files (Compose/volumes) □□

On the **new** server:

```
cd /opt/bookstack
tar -xvzf backups/bookstack-files-YYYY-MM-DD.tar.gz
```

(Adjust the date, of course.)

Step 3: Start containers (DB should already be running) □□

```
cd /opt/bookstack
docker compose up -d mariadb
```

Wait a few seconds until MariaDB is ready.

Step 4: Import the database □□

```
cat backups/bookstack-db-YYYY-MM-DD.sql | \
  docker exec -i bookstack-mariadb mysql -u bookstack -pbookstack bookstack
```

Again: adjust the password or use interactive mode.

Step 5: Start BookStack

```
docker compose up -d
```

□□ Domain/URL changed? (very important!)

If you move to a **different URL/domain**, you must:

1) Adjust `APP_URL` in the `.env`

In your volume, the `.env` is typically located at:

- `./bookstack/www/.env` *(common with linuxserver/bookstack)*
- or `./bookstack/.env` *(depending on image/setup)*

Find it like this:

```
find /opt/bookstack/bookstack -maxdepth 3 -name ".env" -print
```

Then set `APP_URL=https://new-domain.tld`.

> **Pay attention:** The `APP_KEY` **must remain the old one** (otherwise encrypted data will break, e.g., MFA secrets).

2) BookStack command: “Update System URL”

So that content/links in the DB are rewritten cleanly:

```
docker exec -it bookstack php /app/www/artisan bookstack:update-url https://old-domain.tld https://new-domain.tld
```

(If the path in the container is different, this often also works: `php artisan ...` *from* `/app/www`*.)*

□□ Practical notes (so it’s truly “migration-safe”)

- Always store **backups offsite** (another system/storage).
- **Database dump + file archive** is the most portable.
- If you restore to a **newer BookStack version**:
  1. Restore DB
  2. Bring containers up
  3. Then run migrations (if needed):

     ```
     docker exec -it bookstack php /app/www/artisan migrate --force
     ```

If you want: I’ll make you a “one-command backup script” □□

Just tell me briefly:

- Is your `.env` located at `bookstack/www/.env`?
- What are your containers actually named (`docker ps` output)?
- Do you want to back up Caddy certificates or preferably not?

Then I’ll give you a small script including rotation (e.g., “keep the last 7 backups”) and optional upload to S3/Hetzner Storage Box.

Formeln & Diagramme

Eine bunte Sammlung *vieler* Formeln (Inline **und** Block)

Unten findest du **ganz viele unterschiedliche Formeln** – von *sehr einfach* bis *ziemlich komplex*, teils **inline** (mit `$...$`) und teils als **Block** (mit `$$...$$`). Viel Spaß beim Kopieren & Testen

---

1) Kurze Inline-Formeln (sehr gemischt)

- Lineare Terme: $3x-7$, $-2a+5b$, $m\cdot x+c$
- Potenzen & Wurzeln: x^2, a^{n+1}, $\sqrt{2}$, $\sqrt{x^2+y^2}$
- Brüche: $\frac{1}{2}$, $\frac{x+1}{x-1}$, $\frac{a^2-b^2}{a-b}$
- Beträge & Intervalle: $|x|$, $|a-b|$, $x\in[0,1)$
- Summen/Produkte: $\sum_{k=1}^{n}k$, $\sum_{k=0}^{\infty}r^k$, $\prod_{i=1}^{n} i$
- Fakultät & Binomial: $n!$, $\binom{n}{k}$
- Logarithmen: $\ln(x)$, $\log_{10}(1000)$, $\log_a(x)$
- Exponentialfunktionen: e^x, 2^{3x-1}
- Trigonometrie: $\sin(x)$, $\cos^2(\theta)$, $\tan(\alpha+\beta)$
- Hyperbelfunktionen: $\sinh(x)$, $\cosh^2(x)-\sinh^2(x)=1$
- Grenzwerte: $\lim_{x\to 0}\frac{\sin x}{x}$
- Ableitungen: $f'(x)$, $\frac{d}{dx}\left(x^3\right)$, $\frac{\partial f}{\partial x}$
- Integrale: $\int_0^1 x^2,dx$, $\int e^x,dx$
- Komplexe Zahlen: $i^2=-1$, $z=a+bi$, $|z|=\sqrt{a^2+b^2}$
- Vektoren (symbolisch): $|v|$, $\langle u,v\rangle$
- Wahrscheinlichkeiten: $P(A)$, $P(A\mid B)$, $E[X]$, $\mathrm{Var}(X)$
- Informatik/Logik: $p\land q$, $p\to q$, $x\in S$

---

2) Klassische Block-Formeln (Basics bis Standard)

$$
a^2+b^2=c^2
$$

$$
(x+y)^2=x^2+2xy+y^2
$$

$$
(a-b)(a+b)=a^2-b^2
$$

$$
\frac{d}{dx}\left(x^n\right)=n x^{n-1}
$$

$$
\int x^n,dx=\frac{x^{n+1}}{n+1}+C \quad \text{für } n\neq -1
$$

$$
\int_0^{\infty} e^{-ax},dx=\frac{1}{a}\quad \text{für } a>0
$$

$$
\sum_{k=1}^{n} k=\frac{n(n+1)}{2}
$$

$$
\sum_{k=1}^{n} k^2=\frac{n(n+1)(2n+1)}{6}
$$

$$
\sum_{k=0}^{n} \binom{n}{k}=2^n
$$

3) Algebra, Polynome & Gleichungen (auch etwas „knackiger“)

$$
ax^2+bx+c=0 \quad \Rightarrow \quad x=\frac{-b\pm \sqrt{b^2-4ac}}{2a}
$$

$$
x^3-6x^2+11x-6=(x-1)(x-2)(x-3)
$$

$$
\gcd(a,b)\cdot \mathrm{lcm}(a,b)=|ab|
$$

$$
\frac{1}{1-x}=\sum_{k=0}^{\infty} x^k \quad \text{für } |x|<1
$$

$$
\ln(1+x)=\sum_{k=1}^{\infty}(-1)^{k+1}\frac{x^k}{k}\quad \text{für } |x|<1
$$

---

4) Trigonometrie & Analysis

$$
\sin(\alpha+\beta)=\sin\alpha\cos\beta+\cos\alpha\sin\beta
$$

$$
\cos(\alpha+\beta)=\cos\alpha\cos\beta-\sin\alpha\sin\beta
$$

$$
\sin^2(x)+\cos^2(x)=1
$$

$$
\lim_{x\to 0}\frac{1-\cos x}{x^2}=\frac{1}{2}
$$

$$
\frac{d}{dx}\big(\sin x\big)=\cos x
$$

$$
\frac{d}{dx}\big(\ln x\big)=\frac{1}{x}
$$

$$
\int_0^{2\pi}\sin(nx),dx=0 \quad \text{für } n\in \mathbb{Z}\setminus{0}
$$

---

5) Komplexe Zahlen & Euler

$$
e^{i\theta}=\cos\theta+i\sin\theta
$$

$$
e^{i\pi}+1=0
$$

$$
z=re^{i\theta}\quad \Rightarrow \quad \overline{z}=re^{-i\theta}
$$

$$
|z_1 z_2|=|z_1|\cdot |z_2|
$$

---

6) Lineare Algebra (ohne Matrix-Umgebung)

(Da du „ohne Matrix-Umgebung“ erwähnt hast, nutze ich eher symbolische Schreibweisen.)

$$
|x|_2=\sqrt{\langle x,x\rangle}
$$

$$
\langle x,y\rangle=\sum_{k=1}^{n} x_k y_k
$$

$$
\mathrm{proj}_u(v)=\frac{\langle v,u\rangle}{\langle u,u\rangle}u
$$

$$
\det(A)\neq 0 \quad \Rightarrow \quad A^{-1}\ \text{existiert}
$$

---

7) Differentialgleichungen & Dynamik

$$
\frac{dy}{dx}=ky
$$

$$
y(x)=Ce^{kx}
$$

$$
m\frac{d^2x}{dt^2}+c\frac{dx}{dt}+kx=0
$$

$$
\frac{\partial u}{\partial t}=D\frac{\partial^2 u}{\partial x^2}
$$

---

8) Wahrscheinlichkeit & Statistik □□

$$
P(A\cup B)=P(A)+P(B)-P(A\cap B)
$$

$$
P(A\mid B)=\frac{P(A\cap B)}{P(B)}
$$

$$
P(A\mid B)=\frac{P(B\mid A)P(A)}{P(B)}
$$

$$
E[X]=\sum_x x,P(X=x)
$$

$$
\mathrm{Var}(X)=E[X^2]-E[X]^2
$$

$$
\mathrm{Cov}(X,Y)=E[XY]-E[X]E[Y]
$$

$$
\rho_{X,Y}=\frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}}
$$

$$
\hat{p}=\frac{k}{n}
$$

---

9) Optimierung

$$
\min_{x\in \mathbb{R}^n} ; f(x)
$$

$$
\nabla f(x^*)=0
$$

$$
x_{t+1}=x_t-\eta \nabla f(x_t)
$$

$$
\arg\min_x \left(\frac{1}{2}|Ax-b|_2^2+\lambda |x|_1 \right)
$$

---

10) Diskrete Mathematik & Logik □□

$$
p\to q ;\equiv; \neg p \lor q
$$

$$
p\leftrightarrow q ;\equiv; (p\to q)\land(q\to p)
$$

$$
|A\times B|=|A|\cdot |B|
$$

$$
\sum_{k=0}^{n}\binom{n}{k}x^k y^{n-k}=(x+y)^n
$$

---

11) „Komplexere“ kombinierte Ausdrücke

$$
\int_0^1 \left(x^\alpha (1-x)^\beta
\right),dx=\frac{\Gamma(\alpha+1)\Gamma(\beta+1)}{\Gamma(\alpha+\beta+2)}
$$

$$
\frac{d}{dx}\left(\frac{\ln(1+x)}{x}\right)=\frac{\frac{x}{1+x}-\ln(1+x)}{x^2}
$$

$$
\sum_{n=1}^{\infty}\frac{1}{n^2}=\frac{\pi^2}{6}
$$

$$
\forall \varepsilon>0\ \exists \delta>0:\ |x-a|<\delta \Rightarrow |f(x)-f(a)|<\varepsilon
$$

Wenn du willst, kann ich dir das auch **thematisch sortiert** (z.B. nur Analysis, nur Statistik, nur Diskrete Mathe) oder als **„Stress-Test“** mit besonders vielen verschachtelten Brüchen, Summen, Indizes, Fällen usw. bauen.

art10m's BookStack Hacks

<head>

```
<!-- ============================================================================
     BOOKSTACK CUSTOM HEAD CONFIGURATION

     This file contains custom styles, scripts, and integrations for BookStack:
     - Typography and layout adjustments (zoom, colors, spacing)
     - CodeMirror 6 line wrapping for code blocks
     - Markdown-it configuration (soft line breaks)
     - MathJax integration for LaTeX math rendering
     - Mermaid diagram rendering with theme synchronization
     - Light/Dark mode toggle button (works for both guests and logged-in users)
     ============================================================================ -->

<!-- ============================================================================
     CONSOLIDATED STYLES
     ============================================================================ -->
<style>
  /* ----------------------------------------------------------------------------
     DETAILS ELEMENT SPACING
     Ensures proper bottom margin for the last child inside <details> elements
     ---------------------------------------------------------------------------- */
  .page-content details > *:last-child {
    margin-bottom: .2em;
  }

  /* ----------------------------------------------------------------------------
     BLOCKQUOTE ADJUSTMENTS
     - Adds consistent vertical spacing for first/last children
     - Disables overflow scrolling to prevent unwanted scrollbars
     ---------------------------------------------------------------------------- */
  .content-wrap blockquote > :last-child {
```

```
  margin-bottom: .3em;
}

.content-wrap blockquote > :first-child {
  margin-top: .3em;
}

.content-wrap blockquote {
  overflow: visible;
  overflow-x: visible;
  overflow-y: visible;
}

/* ------------------------------------------------------------------------
   PAGE CONTENT ZOOM AND TYPOGRAPHY
   - Applies 1.15x zoom to page content for better readability
   - Compensates heading size with 0.9x zoom to maintain visual hierarchy
   ------------------------------------------------------------------------ */
.page-content {
  zoom: 1.15;
}

.page-content h1,
.page-content h2,
.page-content h3,
.page-content h4,
.page-content h5,
.page-content h6 {
  zoom: .9;
}

/* ------------------------------------------------------------------------
   COLOR SCHEME: LIGHT MODE
   - Dark text on light background for optimal contrast
   - Slightly muted headings for visual hierarchy
   ------------------------------------------------------------------------ */
html .page-content {
  color: hsl(0 0% 10%);
}
```

```
html .page-content h1,
html .page-content h2,
html .page-content h3,
html .page-content h4,
html .page-content h5,
html .page-content h6 {
  color: hsl(0 0% 30%);
}

/* ------------------------------------------------------------------------
   COLOR SCHEME: DARK MODE
   - Light text on dark background
   - Adjusted heading colors for dark theme
   - Muted gutter colors for CodeMirror editor
   ------------------------------------------------------------------------ */
html.dark-mode .page-content {
  color: hsl(0 0% 90%);
}

html.dark-mode .page-content h1,
html.dark-mode .page-content h2,
html.dark-mode .page-content h3,
html.dark-mode .page-content h4,
html.dark-mode .page-content h5,
html.dark-mode .page-content h6 {
  color: hsl(0 0% 70%);
}

html.dark-mode .co .cm-gutters {
  color: hsl(0 0% 33%);
}

/* ------------------------------------------------------------------------
   CODEMIRROR EDITOR STYLING
   Removes border-radius for a cleaner, squared appearance
   ------------------------------------------------------------------------ */
.page-content .cm-editor {
  border-radius: 0;
```

```
}

/* ------------------------------------------------------------------------
   MERMAID DIAGRAM CONTAINER
   - Dashed border for visual identification during development/editing
   - Centered layout with horizontal scroll for large diagrams
   ------------------------------------------------------------------------ */
.mermaid-container {
  border: 1px dashed #4238ff !important;
}

.mermaid {
  margin: 1em auto;
  overflow-x: auto;
  text-align: center;
}

.mermaid svg {
  max-width: 100%;
  height: auto;
  display: inline-block;
}

/* ------------------------------------------------------------------------
   THEME TOGGLE BUTTON (FIXED POSITION)
   - Positioned at bottom-left corner for easy access
   - Circular button with hover/active states
   - Semi-transparent by default, full opacity on hover
   ------------------------------------------------------------------------ */
.theme-toggle-fixed {
  position: fixed;
  bottom: 20px;
  left: 20px;
  z-index: 9999;
  background: var(--color-primary, #206ea7);
  border: none;
  border-radius: 50%;
  width: 44px;
  height: 44px;
```

```
    cursor: pointer;
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.25);
    display: flex;
    align-items: center;
    justify-content: center;
    transition: transform 0.2s ease, box-shadow 0.2s ease, opacity 0.2s ease;
    opacity: 0.7;
  }

  .theme-toggle-fixed:hover {
    transform: scale(1.1);
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.35);
    opacity: 1;
  }

  .theme-toggle-fixed:active {
    transform: scale(0.95);
  }

  .theme-toggle-fixed svg {
    width: 22px;
    height: 22px;
    fill: #ffffff;
  }
</style>

<!-- ============================================================================
     MATHJAX CONFIGURATION
     Enables LaTeX math rendering with $ for inline and $$ for display math
     ============================================================================ -->
<script>
  window.MathJax = {
    tex: {
      inlineMath: [['$', '$']],
      displayMath: [['$$', '$$']]
    }
  };
</script>
<script id="MathJax-script" async src="https://cdn.jsdelivr.net/npm/mathjax@4/tex-mml-
```

```
chtml.js"></script>

<!-- ============================================================================
     BOOKSTACK EVENT LISTENERS AND THEME TOGGLE
     - CodeMirror 6 line wrapping configuration
     - Markdown-it soft line break configuration
     - Light/Dark mode toggle for guests and logged-in users
     ============================================================================ -->
<script>
  (function () {
    'use strict';

    /* ============================================================================
       SHARED THEME STORAGE KEY
       Used by both the toggle button and Mermaid for consistent theme state
       ============================================================================ */
    var THEME_STORAGE_KEY = 'bookstack-guest-dark-mode';

    // Expose storage key globally for the Mermaid module to access
    window.BOOKSTACK_THEME_STORAGE_KEY = THEME_STORAGE_KEY;

    /* ============================================================================
       CODEMIRROR 6: LINE WRAPPING FOR CODE BLOCKS
       Listens for the CM6 pre-init event and enables line wrapping
       for content code blocks to prevent horizontal scrolling
       ============================================================================ */
    window.addEventListener('library-cm6::pre-init', function (event) {
      var detail = event.detail;
      var config = detail.editorViewConfig;
      var EditorView = detail.libEditorView;

      // Only apply line wrapping to content code blocks (not the main editor)
      if (detail.usage === 'content-code-block') {
        config.extensions.push(EditorView.lineWrapping);
      }
    });

    /* ============================================================================
       MARKDOWN-IT: SOFT LINE BREAKS
```

```
   Configures the Markdown editor to convert single newlines to <br> tags
   (GFM-style line breaks)
   ============================================================================ */
window.addEventListener('editor-markdown::setup', function (event) {
  event.detail.markdownIt.set({breaks: true});
});

/* ============================================================================
   LIGHT/DARK MODE TOGGLE BUTTON

   Features:
   - Works for both guests (localStorage) and logged-in users (server-side)
   - Applies saved preference immediately to prevent flash of wrong theme
   - Detects system color scheme preference as fallback for guests
   ============================================================================ */

/**
 * Checks if the guest has dark mode enabled based on localStorage
 * Falls back to system preference if no stored value exists
 * @returns {boolean} True if dark mode should be enabled
 */
function isGuestDarkModeEnabled() {
  var stored = localStorage.getItem(THEME_STORAGE_KEY);
  if (stored !== null) {
    return stored === 'true';
  }
  // Fallback: Check system color scheme preference
  return window.matchMedia && window.matchMedia('(prefers-color-scheme: dark)').matches;
}

/**
 * Determines if the current user is logged in
 * Checks for CSRF token AND user menu elements (both required)
 * @returns {boolean} True if user is logged in
 */
function isUserLoggedIn() {
  var csrfMeta = document.querySelector('meta[name="csrf-token"]');
  var hasToken = csrfMeta && csrfMeta.getAttribute('content');
  var hasUserMenu = document.querySelector('.dropdown-container [data-
```

```
shortcut="favourites_view"]')
        || document.querySelector('[href*="/logout"]');
      return !!(hasToken && hasUserMenu);
    }

    /**
     * Applies the theme by toggling the 'dark-mode' class on <html>
     * @param {boolean} isDark - Whether to enable dark mode
     */
    function applyTheme(isDark) {
      if (isDark) {
        document.documentElement.classList.add('dark-mode');
      } else {
        document.documentElement.classList.remove('dark-mode');
      }
    }

    // ---------------------------------------------------------------------------
    // IMMEDIATE THEME APPLICATION (before DOMContentLoaded)
    // Prevents flash of wrong theme by applying saved preference early
    // ---------------------------------------------------------------------------
    var currentlyDark = document.documentElement.classList.contains('dark-mode');
    var guestPref = localStorage.getItem(THEME_STORAGE_KEY);

    if (guestPref !== null) {
      if (!currentlyDark && guestPref === 'true') {
        document.documentElement.classList.add('dark-mode');
      } else if (guestPref === 'false' && currentlyDark) {
        document.documentElement.classList.remove('dark-mode');
      }
    }

    // ---------------------------------------------------------------------------
    // CREATE TOGGLE BUTTON (after DOM is ready)
    // ---------------------------------------------------------------------------
    document.addEventListener('DOMContentLoaded', function () {
      var isDarkMode = document.documentElement.classList.contains('dark-mode');
      var loggedIn = isUserLoggedIn();
```

```
      // SVG icons for sun (light mode) and moon (dark mode)
      var sunIcon = '<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24"><path d="M20 15.31 23.31 12 20 8.69V4h-4.69L12 .69 8.69 4H4v4.69L.69 12 4 15.31V20h4.69L12 23.31 15.31 20H20zM12 18c-3.31 0-6-2.69-6-6s2.69-6 6-6 6 2.69 6 6-2.69 6-6 6"/></svg>';
      var moonIcon = '<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24"><path d="M12 3a9 9 0 1 0 9 9c0-.46-.04-.92-.1-1.36a5.389 5.389 0 0 1-4.4 2.26 5.403 5.403 0 0 1-3.14-9.8c-.44-.06-.9-.1-1.36-.1z"/></svg>';

      // Create the toggle button element
      var button = document.createElement('button');
      button.className = 'theme-toggle-fixed';
      button.type = 'button';
      button.title = isDarkMode ? 'Activate Light Mode' : 'Activate Dark Mode';
      button.innerHTML = isDarkMode ? sunIcon : moonIcon;

      // Handle click: server-side for logged-in users, client-side for guests
      button.addEventListener('click', function () {
        if (loggedIn) {
          // Logged-in users: Submit form to BookStack's preference endpoint
          var csrfToken = '';
          var csrfMeta = document.querySelector('meta[name="csrf-token"]');
          if (csrfMeta) {
            csrfToken = csrfMeta.getAttribute('content') || '';
          }

          var form = document.createElement('form');
          form.method = 'POST';
          form.action = '/preferences/toggle-dark-mode';
          form.innerHTML =
            '<input type="hidden" name="_token" value="' + csrfToken + '">' +
            '<input type="hidden" name="_method" value="PATCH">' +
            '<input type="hidden" name="_return" value="' + window.location.href + '">';
          document.body.appendChild(form);
          form.submit();
        } else {
          // Guests: Toggle theme client-side and save to localStorage
          var nowDark = document.documentElement.classList.contains('dark-mode');
          var newMode = !nowDark;
```

```
          applyTheme(newMode);
          localStorage.setItem(THEME_STORAGE_KEY, newMode.toString());

          // Update button appearance
          button.innerHTML = newMode ? sunIcon : moonIcon;
          button.title = newMode ? 'Activate Light Mode' : 'Activate Dark Mode';
        }
      });

      document.body.appendChild(button);
    });
  })();
</script>

<!-- ============================================================================
     MERMAID DIAGRAM RENDERING (ES MODULE)

     Features:
     - Automatic detection of mermaid code blocks (multiple selector formats)
     - Theme synchronization with BookStack's dark/light mode
     - Uses shared localStorage key with theme toggle button
     - Theme changes apply on next page load (no live re-rendering)
     ============================================================================ -->
<script type="module">
  import mermaid from "https://cdn.jsdelivr.net/npm/mermaid@11/dist/mermaid.esm.min.mjs";

  // Use the shared storage key from the main script
  const STORAGE_KEY = window.BOOKSTACK_THEME_STORAGE_KEY || "bookstack-guest-dark-mode";

  /* ============================================================================
     THEME DETECTION UTILITIES
     ============================================================================ */

  /**
   * Checks if BookStack is currently in dark mode
   * @returns {boolean} True if 'dark-mode' class is present on <html>
   */
  function isBookStackDarkMode() {
    return document.documentElement.classList.contains("dark-mode");
```

```
}

/**
 * Retrieves the stored theme preference from localStorage
 * @returns {boolean|null} true = dark, false = light, null = not set
 */
function getStoredThemeIsDark() {
  const stored = localStorage.getItem(STORAGE_KEY);
  if (stored === null) return null;
  return stored === "true";
}

/**
 * Saves the theme preference to localStorage
 * @param {boolean} isDark - Whether dark mode is enabled
 */
function storeThemeIsDark(isDark) {
  localStorage.setItem(STORAGE_KEY, isDark ? "true" : "false");
}

/**
 * Determines the Mermaid theme based on current BookStack mode
 * Uses actual DOM state (dark-mode class) for accurate theme selection
 * @returns {string} Mermaid theme name ("dark" or "default")
 */
function getMermaidTheme() {
  return isBookStackDarkMode() ? "dark" : "default";
}

/* ============================================================================
   MERMAID INITIALIZATION
   Disabled auto-start to allow manual control over rendering
   ============================================================================ */
mermaid.initialize({
  startOnLoad: false,
  securityLevel: "strict",
  theme: getMermaidTheme()
});
```

```
/* ================================================================================
   DOM UTILITIES FOR MERMAID CODE BLOCKS
   ================================================================================ */

/**
 * Finds all elements containing Mermaid diagram source code
 * Supports multiple code block formats used by different editors
 * @returns {HTMLElement[]} Array of source elements
 */
function findMermaidSources() {
  const selectors = [
    "pre.mermaid",
    "pre > code.language-mermaid",
    "pre > code.lang-mermaid",
    "code.language-mermaid"
  ];

  const nodes = Array.from(document.querySelectorAll(selectors.join(",")));

  // Normalize to parent <pre> element when applicable
  return nodes.map(n =>
    (n.tagName.toLowerCase() === "code" && n.parentElement?.tagName.toLowerCase() === "pre")
      ? n.parentElement
      : n
  );
}

/**
 * Extracts the text content (Mermaid code) from a source element
 * @param {HTMLElement} node - The source element
 * @returns {string} The trimmed Mermaid code
 */
function extractCodeFromNode(node) {
  const codeEl = node.tagName.toLowerCase() === "pre"
    ? (node.querySelector("code") || node)
    : node;

  return (codeEl.textContent || "").trim();
}
```

```
/**
 * Replaces the original code block with a Mermaid container div
 * Initially hidden to prevent flash of unstyled content
 * @param {HTMLElement} originalNode - The original code block element
 * @param {string} code - The Mermaid diagram code
 * @returns {HTMLElement} The new container element
 */
function replaceWithContainer(originalNode, code) {
  const container = document.createElement("div");
  container.className = "mermaid";
  container.textContent = code;
  container.style.visibility = "hidden";

  originalNode.replaceWith(container);
  return container;
}

/* ============================================================================
   PAGE READY DETECTION
   ============================================================================ */

/**
 * Waits for the page to be fully loaded and rendered
 * Uses multiple techniques to ensure DOM is stable:
 * 1. Wait for 'load' event (or skip if already complete)
 * 2. Wait for two animation frames (ensures layout is calculated)
 * 3. Additional timeout for any async CSS/font loading
 */
async function waitForPageReady() {
  // Wait for window load event
  await new Promise(resolve => {
    if (document.readyState === "complete") return resolve();
    window.addEventListener("load", () => resolve(), {once: true});
  });

  // Wait for layout stabilization (two animation frames)
  await new Promise(r => requestAnimationFrame(() => requestAnimationFrame(r)));
```

```
  // Additional buffer for async resources
  await new Promise(r => setTimeout(r, 150));
}

/* ========================================================================
   MAIN RENDERING FUNCTION
   ======================================================================== */

/**
 * Main entry point: finds, transforms, and renders all Mermaid diagrams
 * Theme is determined once at render time and applied consistently
 * Theme changes will take effect on next page load
 */
async function renderAllMermaid() {
  const sources = findMermaidSources();

  // Even without diagrams, ensure theme state is stored for consistency
  if (!sources.length) {
    await waitForPageReady();
    storeThemeIsDark(isBookStackDarkMode());
    return;
  }

  // Transform code blocks into Mermaid containers
  const containers = [];
  for (const src of sources) {
    const code = extractCodeFromNode(src);
    if (!code) continue;

    const container = replaceWithContainer(src, code);
    // Store original code for potential future use
    container.setAttribute("data-original-code", code);
    containers.push(container);
  }

  await waitForPageReady();

  // Render all diagrams with the current theme
  for (const el of containers) {
```

```
      try {
        await mermaid.run({nodes: [el]});
        el.style.visibility = "visible";
      } catch (e) {
        el.style.visibility = "visible";
        console.error("Mermaid render failed for element:", el, e);
      }
    }

    // Store current theme state for next page load
    storeThemeIsDark(isBookStackDarkMode());
  }

  // Start rendering process
  renderAllMermaid();
</script>
```

functions.php

`/opt/bookstack/bookstack/www/themes/custom/functions.php`

```
<?php

// Import the ThemeEvents class which contains event constants for the theming system
use BookStack\Theming\ThemeEvents;
// Import the Theme facade to register event listeners
use BookStack\Facades\Theme;

/**
 * Register a listener for the CommonMark environment configuration event.
 * This hook is triggered when BookStack sets up the Markdown parser.
 *
 * CommonMark is the Markdown parsing library used by BookStack.
 */
Theme::listen(ThemeEvents::COMMONMARK_ENVIRONMENT_CONFIGURE, function ($environment) {

  // Merge custom configuration into the CommonMark environment
  $environment->mergeConfig([
```

```
    'renderer' => [
      // Convert soft line breaks (single newlines) into <br> HTML tags
      // By default, CommonMark ignores single newlines in Markdown.
      // This setting makes single line breaks visible in the rendered output,
      // which is useful for preserving line formatting in user content.
      'soft_break' => "<br>",
    ]
  ]);

  // Return the modified environment so other listeners can further customize it
  return $environment;
});
```

oEmbeds in `<head>`

```
<style>
  .auto-embed {
    position: relative;
    width: 100%;
    max-width: 100%;
    margin: 1rem 0;
    overflow: hidden;
    background: #000;
  }

  .auto-embed.video {
    aspect-ratio: 16/9;
  }

  .auto-embed.audio {
    aspect-ratio: 16/5;
  }

  .auto-embed.code {
    aspect-ratio: 4/3;
    min-height: 400px;
  }
```

```
  .auto-embed.square {
    aspect-ratio: 1/1;
  }

  .auto-embed iframe {
    position: absolute;
    inset: 0;
    width: 100%;
    height: 100%;
    border: 0;
  }
</style>

<script>
  document.addEventListener('DOMContentLoaded', () => {
    const providers = [
      // □□Video
      {
        regex: /youtu(?:be\.com\/watch\?v=|\.be\/)([\w-]+)/i,
        embed: id => `https://www.youtube-nocookie.com/embed/${id}`, type: 'video'
      },
      {
        regex: /vimeo\.com\/(\d+)/i,
        embed: id => `https://player.vimeo.com/video/${id}`, type: 'video'
      },
      {
        regex: /dailymotion\.com\/video\/([\w-]+)/i,
        embed: id => `https://www.dailymotion.com/embed/video/${id}`, type: 'video'
      },
      {
        regex: /twitch\.tv\/videos\/(\d+)/i,
        embed: id => `https://player.twitch.tv/?video=${id}&parent=${location.hostname}`,
type: 'video'
      },
      {
        regex: /twitch\.tv\/([\w-]+)$/i,
        embed: id => `https://player.twitch.tv/?channel=${id}&parent=${location.hostname}`,
type: 'video'
```

```
      },
      {
        regex: /loom\.com\/share\/([\w-]+)/i,
        embed: id => `https://www.loom.com/embed/${id}`, type: 'video'
      },
      {
        regex: /streamable\.com\/([\w-]+)/i,
        embed: id => `https://streamable.com/e/${id}`, type: 'video'
      },

      // □□Audio
      {
        regex: /open\.spotify\.com\/(track|album|playlist|episode)\/([\w-]+)/i,
        embed: (t, id) => `https://open.spotify.com/embed/${t}/${id}`, type: 'audio'
      },
      {
        regex: /soundcloud\.com\/([\w-]+\/[\w-]+)/i,
        embed: path =>
`https://w.soundcloud.com/player/?url=https://soundcloud.com/${path}&color=%23ff5500&auto_play
=false&hide_related=true&show_comments=false&show_user=true&show_reposts=false&show_teaser=fal
se`,
        type: 'audio'
      },

      // □□Code - CodePen

      // □ CodePen v2.0 Editor-Format (DIREKTES IFRAME!)
      {
        regex: /codepen\.io\/editor\/([\w-]+)\/pen\/([\w-]+)/i,
        embed: (user, penId) => `https://codepen.io/editor/${user}/embed/${penId}?default-
tab=html%2Cresult&theme-id=dark&editable=true`,
        type: 'code'
      },

      // □ Klassisches CodePen-Format
      {
        regex: /codepen\.io\/([\w-]+)\/(?:pen|full|details)\/([\w-]+)/i,
        embed: (user, penId) => `https://codepen.io/${user}/embed/${penId}?default-
tab=html%2Cresult&theme-id=dark&editable=true`,
```

```
        type: 'code'
      },

      // Andere Code-Plattformen
      {
        regex: /codesandbox\.io\/s\/([\w-]+)/i,
        embed: id => `https://codesandbox.io/embed/${id}`, type: 'code'
      },
      {
        regex: /stackblitz\.com\/edit\/([\w-]+)/i,
        embed: id => `https://stackblitz.com/edit/${id}?embed=1`, type: 'code'
      },
      {
        regex: /jsfiddle\.net\/([\w-]+\/[\w-]+)/i,
        embed: path => `https://jsfiddle.net/${path}/embedded/result,js,html,css/`, type:
'code'
      },
      {
        regex: /gist\.github\.com\/([\w-]+)\/([\w-]+)/i,
        embed: (u, id) => `data:text/html,<script
src="https://gist.github.com/${u}/${id}.js"><\/script>`,
        type: 'code'
      },

      // □□□Maps & Design
      {
        regex: /figma\.com\/(file|design)\/([\w-]+)/i,
        embed: (t, id) =>
`https://www.figma.com/embed?embed_host=bookstack&url=https://www.figma.com/${t}/${id}`,
        type: 'code'
      },
      {
        regex: /google\.com\/maps\/embed\?pb=([^&\s]+)/i,
        embed: pb => `https://www.google.com/maps/embed?pb=${pb}`, type: 'video'
      },
    ];

    document.querySelectorAll('p').forEach(p => {
      const text = p.textContent.trim();
```

```
      if (!/^https?:\/\/\S+$/i.test(text)) return;

      for (const {regex, embed, type} of providers) {
        const match = text.match(regex);
        if (!match) continue;

        const src = embed(...match.slice(1));
        p.outerHTML = `<div class="auto-embed ${type}"><iframe src="${src}" allowfullscreen
loading="lazy" allowtransparency="true"></iframe></div>`;
        return;
      }
    });
  });
</script>
```

And this must be added to `docker-compose.yml`

Under `environment`:

```
- ALLOWED_IFRAME_SOURCES=https://*.codepen.io https://codepen.io https://*.jsfiddle.net
https://jsfiddle.net https://*.codesandbox.io https://codesandbox.io https://open.spotify.com
https://gist.github.com https://*.youtube.com https://youtube.com https://www.youtube-
nocookie.com https://*.vimeo.com https://player.vimeo.com https://*.dailymotion.com
https://*.twitch.tv https://player.twitch.tv https://clips.twitch.tv https://*.tiktok.com
https://*.loom.com https://*.wistia.com https://fast.wistia.net https://streamable.com
https://*.vidyard.com https://rumble.com https://*.soundcloud.com https://w.soundcloud.com
https://embed.music.apple.com https://*.deezer.com https://widget.deezer.com
https://*.bandcamp.com https://*.mixcloud.com https://*.audiomack.com https://anchor.fm
https://*.transistor.fm https://*.stackblitz.com https://*.replit.com https://*.glitch.com
https://jsbin.com https://*.observablehq.com https://carbon.now.sh https://asciinema.org
https://platform.twitter.com https://*.instagram.com https://*.reddit.com
https://*.linkedin.com https://assets.pinterest.com https://*.figma.com https://*.canva.com
https://docs.google.com https://*.pitch.com https://prezi.com https://*.slideshare.net
https://speakerdeck.com https://maps.google.com https://*.openstreetmap.org
https://*.notion.so https://*.notion.site https://*.airtable.com https://*.typeform.com
https://form.typeform.com https://calendly.com https://*.miro.com https://excalidraw.com
https://*.diagrams.net https://whimsical.com https://*.lottiefiles.com https://lottie.host
```

```
https://*.sketchfab.com
```

---

https://www.youtube.com/watch?v=UclrVWafRAI