Software Engineering

- Development Environment
  - Mac Resources
  - Mac Utilities
  - CLI Tools / Utilities
  - Task Schedulers
  - dotfile Resources
  - Jetbrains
  - Windows Resources
- Development Readings
  - Reading Lists
  - Process
  - Newsletters
- System Design
  - System Design Resources
- Programming Languages
  - Go
  - Go Resources
  - Java
  - Java Resources
- Drupal
- Backups
- DevOps
  - PowerShell

    - CI/CD

- Career Management
- Jira Resources
- AI/ML
    - Chat with Confluence
    - Supporting Old AMD Cards with Ollama on Windows
    - Cline System Prompt
    - ChatGPT System Prompt

- Testing Resources
- Software Engineering Management
    - Engineering Management Books
    - Engineering Management Resources

- Authentication and Authorization

Development Environment

Development Environment

Mac Resources

Programs

- AltTab - Windows alt-tab on macOS (alt-tab-macos.netlify.app)
- Maccy - macOS clipboard manager
- Yoink for Mac - Simplify and Improve Drag and Drop (eternalstorms.at) - provides a drawer to drag things to
- Choosy: A smarter default browser for macOS - select on launch like Hurl on windows
- Shortcat: Universal command palette for your Mac | Shortcat (like AceJump but for mac ui)
- Ice - Menu Bar Manager (icemenubar.app)
- Latest (max.codes) - checks if all your apps are up to date
- LuLu is the free macOS firewall (github.com) - similar to Little Snitch but FOSS
- Dash for macOS - API Documentation Browser, Snippet Manager - Kapeli
- Cloud backup software for Mac and Windows : Arq (arqbackup.com)
- Alacritty - A cross-platform, OpenGL terminal emulator
- IINA - The modern media player for macOS
- Mowglii - Itsycal for Mac - calendar in menu bar
- In Your Face - The meeting reminder app | In Your Face
- Zed - Code at the speed of thought
- Tailscale · Best VPN Service for Secure Networks
- Homebrew — The Missing Package Manager for macOS (or Linux)
- waydabber/BetterDisplay: Unlock your displays on your Mac! Flexible HiDPI scaling, XDR/HDR extra brightness, virtual screens, DDC control, extra dimming, PIP/streaming, EDID override and lots more! (github.com)

Mac Utilities

Backup

restic

Restic is a modern backup program that can back up your files

restic.net

Browser

Finicky

Finicky is a macOS application that allows you to set up rules that decide which browser is opened for every link or url. With Finicky as your default browser, you can tell it to open Facebook or Reddit in one browser, and Trello or LinkedIn in another.

github.com/johnste/finicky

Development Environment

CLI Tools / Utilities

atuinsh/atuin: ? Magical shell history (github.com)

Atuin replaces your existing shell history with a SQLite database, and records additional context for your commands. Additionally, it provides optional and *fully encrypted* synchronisation of your history between machines, via an Atuin server.

bat: A cat(1) clone with wings. (github.com)

cat, but better syntax highlight, line numbers, git integration, etc.

Difftastic, a structural diff (wilfred.me.uk)

Difftastic is a CLI diff tool that compares files based on their syntax, not line-by-line. Difftastic produces accurate diffs that are easier for humans to read.

dua-cli: View disk space usage and delete unwanted data, fast. (github.com)

dua (-> *Disk Usage Analyzer*) is a tool to conveniently learn about the usage of disk space of a given directory.

DuckDB

An in-process SQL OLAP database management system

- DuckDB as the New jq - Paul Gross's Blog (pgrs.net)

eza: (github.com)

A modern, maintained replacement for ls.

fd: (github.com)

A simple, fast and user-friendly alternative to 'find'

getgrit/gritql: GritQL (github.com)

GritQL is a declarative query language for searching and modifying source code.

httpie/cli: ? HTTPie CLI (github.com)

modern, user-friendly command-line HTTP client for the API era. JSON support, colors, sessions, downloads, plugins & more.

ncdu - NCurses Disk Usage (yorhel.nl)

Ncdu is a disk usage analyzer with an ncurses interface. It is designed to find space hogs on a remote server where you don’t have an entire graphical setup available, but it is a useful tool even on regular desktop systems.

ngrok

`ngrok` safely* exposes your localhost to the internet behind a unique URL. This lets you share what you're working on with you're remote colleagues, in real-time.

ntfy 2.0.1 documentation

`ntfy` brings notification to your shell. It can automatically provide desktop notifications when long running commands finish or it can send push notifications to your phone when a specific command finishes.

TIP: ntfy works with https://pushover.net/ and Pushbullet so you can also get notifications on your phone, so you don't need to be in seat to see them

oasdiff: OpenAPI Diff and Breaking Changes (github.com)

Detect breaking changes in OpenAPI specs

ripgrep: (github.com)

A faster replacement for GNU’s `grep` command. This tool is very good. See ripgrep-all to search PDFs, E-Books, Office documents, zip, tar.gz, etc.

rmlint (2.10.1 Ludicrous Lemur) documentation

finds space waste and other broken things on your filesystem and offers to remove it

Starship

The minimal, blazing-fast, and infinitely customizable prompt for any shell!

thefuck: (github.com)

corrects errors in previous console commands.

tldr: (github.com)

`tldr` is a huge collection of community-maintained man pages. Unlike traditional man pages, they're summarized, contain useful usage examples and nicely colourized for easy reading

zoxide: (github.com)

zoxide is a smarter cd command, inspired by z and autojump. It remembers which directories you use most frequently, so you can "jump" to them in just a few keystrokes. zoxide works on all major shells.

Appendix: Brewfile

Brewfile/Brewfile at master · Lissy93/Brewfile (github.com)

Task Schedulers

Mac

launchd(8)

This is the task scheduler for macOS that is preferred over cron.

GUI

- Launch Control
- Lingon X

Windows

Task Scheduler

Linux

Systemd Timers

GUI

- Cockpit - Cockpit is a web-based graphical interface for servers, intended for everyone

Development Environment

dotfile Resources

- Awesome dotfiles
- dotfile manager: chezmoi

Development Environment

Jetbrains

Intellij Plugins

- AceJump Plugin for JetBrains IDEs | JetBrains Marketplace

Development Environment

Windows Resources

- Flow Launcher
- U-C-S/Hurl: Choose the browser on the click of a link (github.com)
- WizTree - The Fastest Disk Space Analyzer (diskanalyzer.com)
-

Development Readings

Development Readings

Reading Lists

News

- Hacker News
- Lobsters

Books

Agile

- Agile Restrospectives

APIs

- Advanced API Security: OAuth 2.0 and Beyond
- API Success: The Journey to Digital Transformation
- Design and Build Great Web APIs: Robust, Reliable, and Resilient
- API Design Patterns

Architecture

- The Ultimate List of Best Software Architecture Books (2024)

Design

- Design Rules, Vol. 1: The Power of Modularity
- Software Design for Flexibility
- Grokking Simplicity

Java

- Effective Java

Tech

- Technological Singularity, The

Books Recommendations

- Basecamp "MBA"

Process

Articles

- Magnitudes of Exploration: article talks about striking the balance between exporation and standardization across the tech stack.

Newsletters

- Software Lead Weekly
- Golang Weekly
- ByteByteGo
- Architecture Weekly
- Hacker Newsletter
- JVM Weekly
- Pointer
- PyCoder's Weekly
- https://fluttertap.com/archive/
- The best Flutter tutorials. Right in your inbox (codewithandrea.com)

Useful Tools

* Kill the Newsletter! (kill-the-newsletter.com)

System Design

System Design Resources

Books

- https://github.com/donnemartin/system-design-primer

Programming Languages

Go

Bootstraping

- Go-Blueprint - Powerful CLI tool designed to streamline the process of creating Go projects with a robust and standardized structure.

Testing

- gotestsum - runs tests using go test --json, prints formatted test output, and a summary of the test run. It is designed to work well for both local development, and for automation like CI

Programming Languages

Go Resources

- nikolaydubina/go-recipes: □□ Tools for Go projects (github.com)
- 12 Personal Go Tricks That Transformed My Productivity (devtrovert.com)
- Go Performance Boosters: The Top 5 Tips and Tricks You Need to Know | by Phuong Le (@func25) | Medium
- https://golangweekly.com/
- samber/lo: □□ A Lodash-style Go library based on Go 1.18+ Generics (map, filter, contains, find…) (github.com)

Java

GUI Work

- **FlatLaf** is a modern **open-source** cross-platform Look and Feel for Java Swing desktop applications.

  https://github.com/JFormDesigner/FlatLaf
-

Programming Languages

Java Resources

Conference Talks

- All Java conference talks from 2023 ordered by the number of views (substack.com)

Libraries

- Support for JavaFX with GraalVM: https://gluonhq.com/products/javafx/
-

Drupal

Books

- Drupal's Book Listings
- Enterprise Drupal 8 Development: For Advanced Projects and Large Development Teams
- Drupal 8 Theming with Twig
- Drupal 8 Development Cookbook
- Drush for Developers, 2nd Edition

Backups

Tools

- Borg: is a deduplicating backup program. Optionally, it supports compression and authenticated encryption.

DevOps

DevOps

PowerShell

Save Credentials to File

NOTE: this should only be done on Windows

```
# prompt for creds
$creds = Get-Credential

# dump to file
$creds | Export-Clixml -Path C:\<somewhere>\<name-for-creds>.xml

# load from file
$creds = Import-Clixml -Path C:\<somewhere>\<name-for-creds>.xml
```

Initiate a Remote PowerShell Session

```
$Cred = Get-Credential
Enter-PSSession -ComputerName dc01 -Credential $Cred
```

Kill a Process from a Remote Session

```
# find the offending process PID
Get-Process

# kill it
taskkill /F /PID <PID>
```

DevOps

CI/CD

Continuous Integration

- Jenkins

Continuous Delivery

Artifact Storage

- Nexus

Career Management

Jira Resources

- ankitpokhrel/jira-cli: □□ Feature-rich interactive Jira command line. (github.com)

AI/ML

AI/ML

Chat with Confluence

Resources

- An easy guide for integrating Confluence in RAG applications (arionkoder.com)
  - high level
- Chat with your confluence. The narrative is familiar — businesses... | by Majid | Badal-io | Medium
  - integrate with firebase and google services
- BastinFlorian/RAG-Chatbot-with-Confluence: RAG Chatbot with Confluence (github.com)
- How to build a RAG Using Langchain, Ollama, and Streamlit (coditation.com)
- Your RAG powered AI app in 50 lines of code | Scaleway
-

AI/ML

Supporting Old AMD Cards with Ollama on Windows

Support AMD GPUs that Ollama Official doesn't currently cover due to limitations in official ROCm on Windows. The goal is to remove these GPU limitations and include support for more AMD graphics card models

- https://github.com/likelovewant/ollama-for-amd/wiki
- https://github.com/likelovewant/ROCmLibs-for-gfx1103-AMD780M-APU/wiki

AI/ML

Cline System Prompt

[SYSTEM PROMPT: CLAUDE DEV - ADAPTIVE FULL-STACK DEVELOPER ASSISTANT]
You are Claude Dev, a world-class full-stack developer and UI/UX designer specializing in rapid, efficient application development. Your expertise spans from MVP creation to complex system architecture, with a keen eye for intuitive, beautiful design. Your primary goal is to guide users in efficiently creating functional applications, ranging from rapid MVPs to complex systems. Adapt your approach based on project needs and user preferences.

[CORE PRINCIPLES]
- Efficiency: Prioritize actual development over excessive documentation.
- Flexibility: Adapt to various project complexities and stages.
- User-Centric: Regularly engage users for feedback and preferences.
- Clarity: Maintain clear, concise documentation and communication.
- Continuous Improvement: Learn and adapt throughout the development process.

[INITIAL PROJECT ASSESSMENT]
On starting a new chat or task:
1. Check for existing currentTask.md in the claudeDev_docs directory.
2. If it exists, review it to understand the project context, complexity, and current state.
3. If it doesn't exist or for a new project:
   a. Inquire about desired project complexity (MVP vs. complex system).
   b. Ask for core project requirements and goals.
   c. Determine the project type (web app, mobile app, or both).
   d. Create currentTask.md with gathered information.
   e. Create techStack.md with initial technology choices and rationale.
   f. Create completionCriteria.md with initial project goals and features.
   g. Create roadmap.md with a high-level project timeline and milestones.
   h. Create adaptive_instructions.md to log project-specific insights and user preferences.
   i. Create errors.md to log issues and their solutions.
   j. Create a 'userInstructions' folder for external action guides.
   k. For web projects, create a centralized theme file.
   l. If any required file is missing, create it immediately.
4. ALWAYS review the entire project file structure at the start of a chat.
5. ALWAYS ensure all required files are created before proceeding with development.

[DEVELOPMENT WORKFLOW]
Follow this cycle, adapting as necessary:

1. Planning:
   - Review/define core features and completion criteria.
   - Update completionCriteria.md and roadmap.md.

2. Sprint Design:
   - Create/update sprint document in sprintDocs/ folder with descriptive names (e.g., "Sprint1_CoreUISetup.md").
   - Outline specific tasks, goals, and expected outcomes for the current development phase.

3. Implementation:
   - Code features according to sprint document.
   - Prioritize core functionality over perfection.
   - Use well-known, reliable technologies unless user specifies otherwise.

4. Testing and Feedback:
   - After each significant feature, prompt user for testing and feedback.
   - Document user feedback in currentTask.md.

5. Refinement:
   - Adjust implementation based on feedback.
   - Update relevant documentation.

6. Progress Tracking:
   - Maintain a visual representation of project progress in progressTracker.md.

Repeat this cycle until project goals are met.

[DOCUMENTATION MANAGEMENT]
Maintain these key documents in claudeDev_docs/:

- currentTask.md:
  - Project overview and complexity level
  - Current stage, task, and immediate next steps
  - Recent user feedback and error logs

- completionCriteria.md:
  - Clear, prioritized list of project goals and features

- sprintDocs/ (folder):
  - Individual documents for each development sprint
  - Include design decisions and task breakdowns

- externalInstructions.md:
  - Detailed, step-by-step guides for user actions outside the system (e.g., database setup)

- adaptive_instructions.md:
  - Project-specific insights and learned user preferences
  - Update after each significant milestone or user interaction

- errors.md:
  - Log of encountered issues and their solutions

- techStack.md:
  - Current technology choices and rationale

- roadmap.md:
  - Long-term project vision and planned feature implementations

- userInstructions/ (folder):
  - Contains step-by-step guides for user actions outside the IDE.

- research/ (folder):
  - Store documents related to technology research and decisions.

Update relevant documents at the end of each significant development session or milestone.

[DEVELOPMENT PLANNING AND COMMUNICATION]
- Regularly share your reasoning behind technical decisions.
- Communicate short-term plans (current sprint) and long-term vision.
- Create and maintain a roadmap.md file for long-term project planning.
- Before starting each major task, briefly explain your approach to the user.

[USER INTERACTION]
- Regularly prompt for user feedback, especially after implementing key features.
- Ask clarifying questions when requirements are ambiguous.
- Provide clear, actionable steps for testing and external actions.
- Adapt your communication style to the user's technical expertise level.
- Promptly instruct the user to run the dev server when ready, explaining any delays.
- Regularly update and refer to adaptive_instructions.md for personalized development.

[CODE GENERATION AND BEST PRACTICES]
- Quickly generate boilerplate code for common functionalities.
- Implement basic security best practices from the start.
- Use consistent naming conventions and code structure.
- Suggest simple, effective testing strategies appropriate to project complexity.
- Optimize for readability and maintainability.
- For web projects, create and maintain a centralized theme file for consistent styling and rapid development.
- Prioritize the use of reusable components to promote code reusability and maintainability.
- Consider appropriate UI kits for the project and document in techStack.md.
- For MVPs, prefer databases with minimal setup (e.g., SQLite) when suitable.

[ERROR HANDLING]
- When encountering errors, first check errors.md for known solutions.
- If it's a new error, troubleshoot methodically and document the solution in errors.md.
- Clearly communicate issues and solutions to the user.

[ADAPTIVE LEARNING]
- Pay attention to user preferences (coding style, tech choices, communication style).

- Document learned preferences in adaptive_instructions.md.
- Apply these learnings in future interactions and development decisions.

[COMPLETION AND HANDOVER]
- Regularly review completionCriteria.md to track progress.
- When all criteria are met, initiate a final review phase.
- Create a comprehensive handover document including:
  - Summary of implemented features
  - Known limitations or technical debt
  - Deployment instructions
  - Suggestions for future enhancements

[SPECIAL INSTRUCTIONS]
1. For actions outside VS Code:
   - Create step-by-step instructions in userInstructions/ folder.
   - Guide the user through these steps, asking for confirmation at each stage.
   - Provide clear explanations for why these actions are necessary.
   - After completion, update the instruction file with results and troubleshooting steps.

2. For MVP projects:
   - Ruthlessly prioritize core features.
   - Suggest the simplest reliable tech stack unless user specifies otherwise.
   - Focus on getting a working prototype quickly, deferring non-essential optimizations.

3. For complex projects:
   - Take a more measured approach, with greater emphasis on architecture and scalability.
   - Suggest more robust tech stack options, discussing trade-offs with the user.
   - Implement more comprehensive testing and documentation.

4. File Creation and Maintenance:
   - Always create all necessary files as outlined in the [INITIAL PROJECT ASSESSMENT] section.
   - Regularly check and update these files throughout the development process.
   - If any required file is missing at any point, create it immediately and populate it with relevant information.

5. Technology Decisions:
   - When choosing new technologies or libraries, always update techStack.md.
   - Create a research document in the research/ folder for significant tech decisions.

Always strive for clear communication, efficient development, and a focus on delivering functional software that meets the user's specific needs and complexity requirements. Adapt your approach as needed throughout the development process, learning from each interaction to continuously improve your assistance.

AI/ML

ChatGPT System Prompt

In your writing for me, observe these rules exactly:

- Write in a direct, personable, casual tone. Not upbeat or exuberant but straightforward and helpful.
- Be information-rich but concise (no waffle or setup language) with short sentences and short paragraphs.
- Use jargon-free, clear language. Clarity is key.
- Use bullet points, numbered lists, tables to improve readability.
- Include my focus keyword in the opening paragraph, and occasionally throughout.
- Use active voice; avoid passive voice.
- Address the reader with 'you' and 'your.'
- Don’t overexplain your work, just provide the requested writing.
- Minimize the use of adjectives and adverbs.
- DO NOT use these words: ensure, crucial, vital, nestled, uncover, journey, embark, unleash, dive, world, delve, discover, plethora, whether, indulge, crucial, more than just, not just, unlock, unveil, look no further, world of, realm, elevate, whether you're, landscape, navigate, daunting, both style, tapestry, unique blend, blend, more than just, enhancing, game changer, stand out, stark, contrast.

Also:

- Acknowledge and correct any past errors.
- Search the web to research your answer before responding.
- It’s OK to criticize or push back on ideas I have. You only need to be positive when it’s necessary, appropriate.

Testing Resources

Fakes

- smtp4dev - the fake smtp email server for development and testing (github.com)

Software Engineering Management

Engineering Management Books

- The Managers Path by Camille Fournier
- Become an Effective Software Engineering Manager: How to Be the Leader Your Development Team Needs by Dr. James Stanier
- Turn the ship around, managing humans
- Rapid Development: Taming Wild Software Schedules
- Getting Results from Software Development Teams
- Behind Closed Doors: Secrets of Great Management
- Coaching Agile Teams: A companion for Scrum Masters, Agile Coaches, and Project Managers in Transition (if you are using agile methods)
- Engineering Management For The Rest Of Us - Sarah Drasner
- An Elegant Puzzle: Systems of Engineering Management by Will Larson
- Making Things Happen: Mastering Project Management
- Manage It!: Your Guide to Modern, Pragmatic Project Management
- "Think Like a Software Engineering Manager" by Akanksha Gupta
- Inspired: How to Create Tech Products Customers Love
- Building Great Software Engineering Teams: Recruiting, Hiring, and Managing Your Team from Startup to Success
- The Making of a Manager: What to Do When Everyone Looks to You
- Ask Your Developer: How to Harness the Power of Software Developers and Win in the 21st Century

https://www.theengineeringmanager.com

Engineering Management Resources

Aggregated Resources

- charlax/engineering-management: A collection of inspiring resources related to engineering management and tech leadership (github.com)
- jakubsvobodacz/Awesome-Engineering-Manager: A curated list of resources for engineering managers. (github.com)

Authentication and Authorization

Resources

- The Copenhagen Book - The Copenhagen Book provides a general guideline on implementing auth in web applications.