Unified Reports (Complete): CS Graduate Portfolios in 2025

This document merges the complete versions of two reports from this chat:

- Crafting a Standout Portfolio for New CS Graduates in 2025
- Portfolio Examples for New CS Graduates: Software Engineering vs. Web Development

Generated by GPT-5 Pro — September 15, 2025

Table of Contents

Report 1 — Crafting a Standout Portfolio for New CS Graduates in 2025

Introduction: The Competitive Landscape for New Graduates

Landing a first job in today's tech industry is highly competitive. Remote-friendly roles can draw hundreds or thousands of applicants. Tech layoffs and AI-driven efficiencies raise the bar. A portfolio is the most credible way to demonstrate skill: unlike a resume line or a grade, it is proof you can build and ship working software.

Portfolio Basics: What Employers Expect

A well-crafted portfolio includes: About Me (2–3 sentences), Projects (3–6 with problem, architecture, stack, screenshots, and links to a live demo and repo), Skills (relevant languages, frameworks, platforms, tools), Contact (email + LinkedIn), a one-page PDF Resume, and optional write-ups or a short blog.

Common Project Sources

Academic (capstone or substantial course projects), personal builds solving your own problems, open-source contributions (even small PRs), and hackathon/internship work. Depth beats quantity—quality projects with clear docs and a deploy link win.

Portfolio Quality Levels

Below-Average Portfolio (Needs Improvement)

- 1–2 tutorial clones (calculator, weather, Netflix) with minimal customization or documentation.
- No deploy link, broken links, inconsistent styling; thin READMEs.
- No real-world context: unclear problem, no ‘why’, no discussion of tradeoffs.

Average Portfolio (Meets Basic Expectations)

- 3–5 projects, at least one non-course project; deployed or easy to run.
- Descriptions include what/why/how, stack list, and screenshots; repos are organized.
- Simple personal site or structured GitHub profile; competent but conventional.

Above-Average Portfolio (Strong Candidate)

- Projects with originality and depth that solve real problems; modern tech (cloud deploy, containers, CI, OAuth, APIs, mobile).

- Polished presentation: consistent branding, clear READMEs, test coverage, deploy links.
- Evidence of collaboration (open-source PRs, hackathons, internships) and clear design tradeoffs.

Elite Portfolio (Truly Exceptional)

- Innovative or high-impact systems with users (SaaS, robotics/IoT, complex game/engine).
- Mastery in at least one area + breadth across the stack; strong personal brand.
- Leadership signals: open-source maintenance, publications, talks, awards.

Showcasing AI Skills In Your Portfolio

AI fluency is valuable even outside pure ML roles. Include at least one AI-driven project if you can ship it responsibly.

Should your portfolio include an agentic system you built?

Yes—if you can make it robust and safe. An agent should use tools/APIs, handle errors, and produce auditable outputs. If a full agent is too much, ship a smaller LLM-with-tools feature (search, retrieval, calculator, code runner) with tests and guardrails.

Agentic AI—What it is

Agentic systems pursue goals with limited supervision by planning, deciding, and taking actions, often via external tools/orchestrators.

Good AI project patterns for new grads

- Tool-using agent: a research assistant that searches, opens pages, and summarizes with citations using an orchestration library (e.g., LangChain/LangGraph).
- LLM-augmented feature inside a conventional app: code-aware helper, structured output validator, AI form-filler.
- Classical ML system: train and evaluate a model, deploy behind an API, integrate into a UI with metrics and known failure cases.

Example AI portfolio entry

Smart Resume Reviewer (GPT-Powered): a web app that critiques resumes. Parses a PDF upload, analyzes sections, suggests improvements. Stack: Python, an agent framework, an LLM API, PDF parsing, React frontend. Outcome: useful feedback; documented limits and privacy.

Layout & Presentation — GitHub and Personal Site

GitHub structure

- Pin 4–6 best repos; each repo has a one-screen README (problem → solution → run steps) and a /docs folder for depth.
- Add minimal tests and CI (GitHub Actions). Include a LICENSE. Use issues and a simple CHANGELOG to show iteration.
- Readable commits and PR descriptions that show collaboration habits.

Personal site structure

- Above the fold: who you are, what you build, and a clear call-to-action (email/LinkedIn).
- Projects: card layout linking to live demos and repos; 2–3 sentences and 1–2 screenshots each.
- Optional: a /now or /blog page with short build notes or lessons learned.

Final Tips & Conclusion

- Aim above average: invest in one or two original, polished builds rather than many shallow clones.
- Talk about value: describe the problem solved, user impact, and what you learned.
- Keep learning: align with cloud, security, and AI to stay relevant.
- Iterate: ask for feedback, file issues, and treat your portfolio like a product.
- Be authentic: choose projects you care about—memorable portfolios come from genuine curiosity.

Report 2 — Portfolio Examples for New CS Graduates: Software Engineering vs. Web Development

Below are complete portfolio sites/GitHub profiles and standout project repos suitable for recent CS graduates targeting U.S. roles. Each example includes a brief note on why it is strong.

Software Engineering Portfolios

• Armaiz Adenwala — Portfolio (https://armaizadenwala.com/) — Clean personal site highlighting full-stack projects (React, Rails, Python) and IoT/hardware builds; strong project write-ups.

• Sage Thomas — Portfolio (https://sage-t.github.io/) — Recent CU Boulder grad; mixes IoT devices, apps, and ML experiments; concise, readable descriptions.

• Jacob Martyniak (ThatGuyJacobee) — GitHub (https://github.com/ThatGuyJacobee) — Uses GitHub as a living portfolio; maintains Elite Music and documents the broader Elite Bot ecosystem (real users, OSS practice).

• Tyler Landtroop — Personal Portfolio (https://www.landtroopt.com/) — Modern React/Next.js site with a terminal-style interface and a Notion-style editor project; crisp, recruiter-friendly layout.

Standout Project Examples (Software Engineering)

• Elite Music (Discord Music Bot) (https://github.com/ThatGuyJacobee/Elite-Music) — Feature-rich, real-time bot (discord.js). Demonstrates API integration, event-driven programming, and open-source practices.

• Elite Bot — Documentation (https://elite-bot.com/docs/) — Polished docs for a multipurpose bot; shows shipping and documenting production-like software.

• PlanSpace — Automated Chore Scheduling (https://armaizadenwala.com/projects/planspace/) — Full-stack mobile/web app that rotates chores among roommates. Clear problem → solution narrative, real-time updates.

• Programmable LED Rave Mask (Arduino/NeoPixel) (https://armaizadenwala.com/blog/how-to-create-a-led-rave-mask-using-arduino/) — Memorable hardware-software project that signals breadth beyond standard web apps.

• DateSync — NFC Meeting Scheduler (https://measeaustin.github.io/) — Android app using NFC to exchange availability and propose meeting times; mobile + hardware APIs.

Web Development Portfolios

• Tyler Landtroop — Personal Portfolio (https://www.landtroopt.com/) — Front-end focused; includes terminal-style site and a Notion-like editor project (Next.js 13/14).

• Sam Cook — Portfolio (https://samcook.vercel.app/) — Recent Texas State CS grad; MERN and Rust/React projects; clear stacks and deploy notes.

• Tamal Sen — Developer Portfolio (https://tamalsen.dev/) — Creative, polished design with strong UX; inspiration for front-end craft and branding.

• Bruno Simon — 3D Portfolio (Inspiration) (https://bruno-simon.com/) — Gamified Three.js/WebGL portfolio; advanced front-end skill and delightful interaction.

Standout Project Examples (Web Development)

• Terminal-Style Portfolio (React/Next.js) — repo (https://github.com/tylerlandtroop/terminal-portfolio) — A CLI-like UX demonstrating command parsing, theming, and custom interactions.

• Jotion — Notion-style Notes App — repo (https://github.com/tylerlandtroop/jotion) — Complex editor features (blocks, drag-and-drop, dark mode) using modern Next.js.

• Blockparty — MERN Social App — repo (https://github.com/sam-cook/blockparty) — Real-time features via Socket.io; end-to-end stack ownership and deployment.

• Madison’s Chat App — write-up (https://careerfoundry.com/en/blog/web-development/software-engineer-portfolio/) — MERN/React Native chat app presented with a short GIF demo; excellent pattern for showcasing UX quickly.

• Interactive 3D Portfolio (Three.js/WebGL) (https://bruno-simon.com/) — Playful, interactive experience that makes front-end skills obvious at a glance.

References & Inspiration

• General Assembly — Portfolio Guide (https://generalassemb.ly/blog/software-engineering-resume-portfolio-guide/)

• CareerFoundry — Software Engineer Portfolio Examples (https://careerfoundry.com/en/blog/web-development/software-engineer-portfolio/)

• CareerFoundry — Web Developer Portfolios (https://careerfoundry.com/en/blog/web-development/web-developer-portfolio/)

• IBM — What is Agentic AI? (https://www.ibm.com/think/topics/agentic-ai)

• AWS — What is Agentic AI? (https://aws.amazon.com/what-is/agentic-ai/)

• LangChain — Agents (Docs) (https://python.langchain.com/api_reference/core/agents.html)

- LangChain — Agent Tutorial (https://python.langchain.com/docs/tutorials/agents/)
- AutoGPT (https://github.com/Significant-Gravitas/AutoGPT)

Quick Hit Checklists & Pro Tips

- Pin 4–6 repos; each has a one-screen README and run steps.
- Every project has a live demo or a quick-start script.
- Ship tests + CI (even minimal).
- Show at least one AI project; if not an agent, add an LLM-with-tools feature with guardrails.
- Use a simple personal domain and fast hosting (Vercel/Netlify).