

Security, quality and launch checklist

DMILZE EdTech · Full-Stack Development with AI · Weeks 2, 6, 7, 10 and 11. Tick every line before you call a project done.

Reviewing AI-generated code (every week)

- ☐ I can explain every line it wrote
- ☐ It uses the project's patterns (see AGENTS.md)
- ☐ No invented packages or APIs: I checked each import exists
- ☐ No secrets, keys or real personal data in the code
- ☐ Inputs are validated; errors are handled
- ☐ Tests added or updated, and they pass
- ☐ I ran it, not just read it

Authentication and access (Week 6)

- ☐ Use a proven auth provider (Supabase Auth or Auth.js); never write your own password storage
- ☐ Every server action and API route checks who the user is
- ☐ Every query checks the user may see that record (no reading other users' data by changing an ID)
- ☐ Roles enforced on the server, not only hidden in the UI
- ☐ Sessions expire; logout works everywhere
- ☐ Password reset and email verification work

Common web risks (Week 6)

- ☐ All input validated with Zod on the server
- ☐ Database access only through the ORM or parameterised queries
- ☐ No user content rendered as raw HTML
- ☐ Rate limits on login, sign-up and AI endpoints
- ☐ File uploads: type and size checked, stored outside the app, private by default
- ☐ Security headers set; HTTPS everywhere
- ☐ Dependencies checked (npm audit) and kept up to date

Secrets and data protection (Week 6)

- ☐ .env is in .gitignore; .env.example lists every variable
- ☐ Different keys for development and production
- ☐ Service-role and secret keys only on the server
- ☐ Personal data minimised; you know where it is stored (NDPA)
- ☐ Database backups on, and a restore tested

Testing and quality (Week 7)

- ☐ Unit tests for business logic
- ☐ Integration tests for API routes
- ☐ End-to-end tests for the main user flow (Playwright)
- ☐ Lighthouse: performance and accessibility checked
- ☐ Keyboard and screen reader basics: labels, focus, contrast

Deployment and monitoring (Week 10)

- ☐ CI runs type check, tests and build on every push
- ☐ Production environment variables set in the host (for example Vercel)
- ☐ Database migrations run as part of deploy
- ☐ Health check endpoint monitored (for example UptimeRobot or Better Stack)
- ☐ Errors reported (for example Sentry) without personal data
- ☐ Logs kept and searchable; backups scheduled
- ☐ A rollback plan: how to go back to the last good version

Performance and cost (Week 11)

- ☐ Slow queries found with EXPLAIN ANALYZE; indexes added where needed
- ☐ Lists paginated
- ☐ Caching where data can be slightly stale
- ☐ Slow work moved to background jobs
- ☐ AI and API usage limits per user and per plan
- ☐ Monthly cost estimate written down

Only test the security and load of apps you own or are allowed to test.