Fullstack Web App - Assignment1

Student name:

Location: Lusaka

Cohort: AI Coding Bootcamp Cohort 1

Total Grade:

# Comments

# Rubric (1 = Unacceptable → 5 = Exceptional)

| **Category** | **1 — Unacceptable** | **2 — Needs Work** | **3 — Meets Expectations** | **4 — Exceeds Expectations** | **5 — Exceptional** |
| --- | --- | --- | --- | --- | --- |
| **Design (UI/UX)** | Inconsistent layout; illegible or inaccessible; no mobile support. | Basic responsiveness; noticeable visual bugs; limited attention to a11y. | Clean, consistent UI; mobile‑friendly; passes basic [a11y checks](https://www.a11yproject.com/checklist/) (contrast, keyboard nav). | Thoughtful visual hierarchy; custom theming; comprehensive [a11y](https://www.a11yproject.com/checklist/) (screen‑reader flows). | Pixel‑perfect, branded design; motion/interaction polish; formal a11y audit with fixes. |
| **Frontend Implementation** | Frequent runtime errors; spaghetti code; no state mgmt conventions. | Works but brittle; large components; ad‑hoc state handling. | Modular components; state handled via TanStack Query/RHF; minimal warnings. | Well‑typed hooks; code‑splitting; performance optimizations (lazy‑load, memo). | Production‑level quality: SSR/SEO, exhaustive error states, lighthouse ~90+. |
| **Backend / API** | Endpoints fail or are missing; insecure rules; no validation. | CRUD works but poor error handling; some validation gaps. | tRPC routes typed; Zod validation; Firestore rules enforce auth. | Thoughtful data modelling; composite indexes; graceful failures. | Multi‑env config; seeding scripts; zero‑downtime migrations or blue‑green deploys. |
| **Dev Experience & CI/CD** | Manual builds; no linter/tests in pipeline; flaky deploys. | Basic GitHub Action to deploy; tests run locally only. | Turbo‑cached pipeline: lint, type‑check, tests, Storybook build, preview deploy. | Parallel jobs, test reports, codecov; deploy promotes on tag/Changeset. | Cache‑aware, <5 min runtime; canary deploys; Slack/Discord notifications & rollbacks. |
| **Cloud / IT Ops** | Hard‑coded secrets; no monitoring; unclear infra costs. | Env vars in repo secrets; basic Firebase console logs. | Secrets via T3 Env + functions:config; Cloud Logging dashboards. | Alerting rules, Crashlytics/Sentry integration; cost budgets. | IaC or scripts for full recreate; autoscaling tuned; custom metrics & alerts. |
| **Product Management** | No clear goals; scope creep; missing acceptance criteria. | Trello/Issues exist but vague; ad‑hoc prioritization. | Defined MVP; backlog groomed; demo accepts against criteria. | Road‑map with milestones; burn‑down chart; stakeholder demos. | Data‑driven decisions (analytics); retro action items implemented; public changelog. |
| **Quality & Testing** | No automated tests; manual QA only. | <30 % unit coverage; flaky E2E; lint disabled. | ≥60 % unit+component coverage; Playwright happy path; lint & Prettier pass CI. | Visual regression via Storybook; a11y checks; seed data resets; ≥80 % coverage. | Mutation or property‑based tests; contract/fuzz tests; zero‑regression policy. |
| **Security** | Public DB; default rules; secrets in code. | Auth enforced but rules overly broad; no dependency scanning. | Principle‑of‑least‑privilege rules; OWASP top‑10 reviewed; secrets managed. | Automated security tests (ZAP/GH Dependabot); 2FA enforced on repo. | Threat model documented; security ADRs; periodic penetration test results. |
| **Architecture & Code Organization** | Single huge file; unclear boundaries; no ADRs. | Ad‑hoc folders; circular deps; inconsistent naming. | Follows monorepo layout; shared package for types; ADR in /docs. | Decoupled modules; clear domain boundaries; tree‑shakeable libraries. | Hexagonal/CQRS or similar advanced patterns; plug‑in architecture; exemplary ADR trail. |