AGENTS.md

This file has two parts:

- **Part A** — instructions for AI coding agents (Codex, Claude Code, Copilot, Cursor, etc.) working in this repository.
- **Part B** — the catalogue of agents represented inside the ALPHA platform.

Part A — Instructions for AI Coding Agents

Setup & commands

```
cd nextjs_space
yarn install
yarn dev        # dev server on :3000
yarn lint
yarn build
```

Rules

1. Framework is **Next.js 16 + React 19**. `params` / `searchParams` are Promises; `cookies()` / `headers()` must be awaited;
   use `proxy.ts`, never `middleware.ts`; never add a `webpack` key to `next.config.js`.
2. Keep the design system: `#0A0A0F`, `#00F5FF`, `#8B5CF6`, `#EC4899`, `glass-panel` utilities, DM Sans / Plus Jakarta Sans / JetBrains Mono. The app is forced dark mode.
3. Mock data goes in `lib/mock-data.ts` with explicit types. Do not scatter literals across pages.
4. No non-deterministic values in render (`Math.random`, `Date.now`, `new Date()`); use effects or `SafeDate`.
5. Shell and shared-UI files use the `alpha-*` prefix (`alpha-shell.tsx`, `alpha-sidebar.tsx`, `alpha-ui/`); update every import if you rename them.
6. The brand is **ALPHA** (`alpha_platform` where an identifier is needed). Do not use any former product name in code, UI, or docs.
7. Never commit secrets. Never add UI copy asserting an unimplemented security control is active.
8. Do not edit `.abacus.donotdelete` or the Prisma `generator client` block.
9. Update README / ARCHITECTURE / CHANGELOG when behavior or structure changes.

Part B — Platform Agent Catalogue

Agents shown in the Agent Mesh (source: `lib/mock-data.ts → agents`). Budgets are the simulated values.

AI Agents

ID	Name	Role / capabilities	Token budget
`orch-prime`	Orchestrator-Prime	Orchestration, routing, scheduling, meta-reasoning	500,000
`code-alpha`	CodeGen-Alpha	Code generation, refactoring, testing, review	800,000
`data-01`	DataHarvester-01	Web scraping, extraction, ETL, normalization	300,000
`mem-idx`	MemoryIndexer	Indexing, embedding, vector search, dedup	200,000
`qa-lead`	QA-Validator-Lead	Testing, validation, benchmarking, regression	150,000
`research-01`	ResearchAgent-01	Research, synthesis, citation, summarization	600,000
`sentinel-01`	Sentinel-Security	Threat detection, anomaly analysis, zero-trust enforcement	400,000
`planner-01`	StrategicPlanner	Planning, goal decomposition, resource allocation, causal reasoning	250,000

Bots

ID	Name	Capabilities	Token budget
`bot-slack`	SlackBot-Notifier	Notifications, alerts, commands	100,000
`bot-monitor`	HealthMonitor-Bot	Health checks, metrics, alerting	80,000
`bot-deploy`	DeployBot	Deployment, rollback, CI/CD	60,000

Swarms, MCP servers, APIs, and databases are also mesh nodes; see `swarms` and `mcpServers` in the mock data.

Agent Card Template

Use this format when proposing a new agent:

```yaml
id: invoice-reconciler
name: InvoiceReconciler
role: Match invoices to purchase orders and flag discrepancies
model: { primary: claude-sonnet, fallback: gpt-4o }
tools:            # deny-by-default allow-list
  - mcp-postgres:query_readonly
  - mcp-email:draft          # draft only; sending requires HIL
memory:
  read:  [ns:finance/invoices]
  write: [ns:finance/reconciliation]
budgets: { tokenBudget: 150000, maxIterations: 8, timeoutMs: 120000 }
escalation:
  humanApprovalRequired: [payments, outbound_email, data_deletion]
  onBudgetExceeded: halt_and_raise_p2
```