☑ PHASE 0 — FOUNDATION & REPO SETUP

Goal: Create solid infrastructure, skeleton folders, CI, and design baselines.

Epic 0.1 — Create Monorepo & Core Structure

Tasks:

1. Initialize monorepo with pnpm or turborepo (recommended).
2. Create subfolders:
    a. `/backend` (FastAPI)
    b. `/frontend` (React + TS + Tailwind)
    c. `/services/mood` (for classifiers)
    d. `/services/ingestion` (Brightspace adapter)
    e. `/infra` (Docker, deployment, compose files)
3. Add `.gitignore`, `LICENSE`, and top-level `README.md`.
4. Set up basic Dockerfile for backend + frontend.
5. Add Docker Compose that spins up Postgres + backend + frontend.

Epic 0.2 — CI/CD & Quality

Tasks:

1. Add GitHub Actions pipeline:
    a. Lint (ruff for Python, ESLint for TS)
    b. Type checking (MyPy, TypeScript)
    c. Build test (docker compose build)
2. Add pre-commit hooks.
3. Add unit test skeletons (pytest + vitest).

Epic 0.3 — Data Model & API Contracts

Tasks:

1. Create `/backend/schemas` with pydantic models:
    a. `User`, `Task`, `JournalEntry`, `MoodProfile`, `PriorityResult`
2. Write OpenAPI stubs for all endpoints.
3. Define REST contracts for microservices.
4. Add "fake stub implementations" so frontend can start integration early.

🚀 PHASE 1 — BACKEND CORE (User, Tasks, Journal)

Goal: Functional backend API with authentication, task storage, and journaling.

Epic 1.1 — User Authentication

Tasks:

1. Create `/backend/app/main.py` and initialize FastAPI.
2. Add JWT auth (fastapi-users or custom).
3. Build `POST /auth/register`, `POST /auth/login`, `GET /auth/me`.
4. Create `users` table (UUID PK, email, timezone, prefs, consents).

Epic 1.2 — Task Store API

Tasks:

1. Create `tasks` table.
2. Implement CRUD routes:
    a. `GET /tasks`
    b. `POST /tasks`

    c. `PUT /tasks/{id}`
    d. `DELETE /tasks/{id}`
3. Add “source” to tasks (brightspace / calendar / manual).
4. Add migration system (Alembic).

Epic 1.3 — Journal Service (Journa)

Tasks:

1. Create `journal_entries` table.
2. Implement:
    a. `POST /journal` (submit entry)
    b. `GET /journal?since=...`
3. Add optional `mood_label` field (to be filled later).
4. Add rate-limiting (optional).

🔄 PHASE 2 — BRIGHTSPACE / UW LEARN INTEGRATION

Goal: Real scheduling + assignments sync.

Epic 2.1 — OAuth & Token Storage

Tasks:

1. Read D2L Valence API docs.
2. Implement OAuth handshake with required scopes.
3. Store tokens encrypted in DB.

Epic 2.2 — Ingestion Pipeline

Tasks:

1. Write `/services/ingestion/brightspace_client.py`.
2. Implement:
    a. Fetch courses
    b. Fetch content modules
    c. Fetch assignments
3. Map → canonical `Task` model.
4. Implement `POST /sync/brightspace` route.
5. Add a scheduled cron job (`/infra/cron`).

Epic 2.3 — Calendar Integration (Optional Early, Required Later)

Tasks:

1. Add Google OAuth (read/write events).
2. Implement `/sync/calendar`.

🧠 PHASE 3 — MOOD SYSTEM (Text, Behavior, Fusion)

Goal: Emotion inference + fused mood profile.

Epic 3.1 — Mood Analyzer (Text)

Tasks:

1. Create `/services/mood/model.py`.
2. Integrate pretrained model: DistilBERT fine-tuned on GoEmotions.
3. Add LIWC-like lexicon features (light library).

4. Build feature extraction pipeline.
5. Build `POST /mood/analyze-text` route.

Epic 3.2 — Behavioral Mood Predictor

Tasks:

1. Define behavioral features:
    a. time-of-day of work
    b. assignment load
    c. upcoming deadlines
    d. journaling frequency
    e. inactivity streaks

(inspired by MoodScope study)

2. Build simple model:
    a. random forest OR linear regression baseline
3. Add `POST /mood/predict-behavioral`.

Epic 3.3 — Mood Fusion Service

Tasks:

1. Merge text + behavioral signals into a single `MoodProfile`:
    a. Weighted averaging w/ confidence
    b. Softmax normalization
2. Implement `GET /mood/current`
3. Store mood history for insights charts.

🧩 PHASE 4 — PRIORITY ENGINE (Extended Eisenhower)

Goal: Turn tasks + mood → actionable priorities.

Epic 4.1 — Eisenhower Base Logic

Tasks:

1. Create scoring function for urgency = f(due_date).
2. Create scoring function for importance = f(weight, user manual importance).
3. Create matrix buckets: DO / SCHEDULE / DELEGATE / DROP.

Epic 4.2 — Mood-Augmented Priority Modeling

Tasks:

1. Adjust priority with `MoodProfile`:
    a. If low valence → downweight high-cognitive-load tasks
    b. If high arousal → upweight tasks requiring focus
2. Use simple linear/non-linear scoring first.
3. Serve via `GET /priority/today`.

Epic 4.3 — Rationale Generator

Tasks:

1. For each prioritized task, produce textual rationale:
    a. “High urgency (due in 12h)”
    b. “Suitable for mood (low mental load)”
2. Add to API response.

🗓️ PHASE 5 — PLANNER / SCHEDULER

Goal: Generate actionable calendar suggestions.

Epic 5.1 — Time Blocking Engine

Tasks:

1. Parse user's free times from calendar.
2. Implement algorithm to propose 25/50 minute blocks.
3. Add `POST /planner/propose`.

Epic 5.2 — Event Writer

Tasks:

1. Write proposed events to Google calendar.
2. Add `POST /planner/confirm`.

💻 PHASE 6 — FRONTEND (React + TypeScript + Tailwind)

Goal: User interface for journaling, task views, prioritization, and planning.

Epic 6.1 — Global UI Setup

Tasks:

1. Initialize React + TS + Vite.

2. Add TailwindCSS.
3. Add Zustand (or Redux Toolkit) for state.
4. Implement auth pages (login, register).

Epic 6.2 — Dashboard Page

Tasks:

1. Display prioritized tasks with rationale.
2. One-click “schedule for today”.
3. Mood indicator widget (emoji + trend chart).

Epic 6.3 — Journal Page

Tasks:

1. Textbox + quick tags.
2. Submit to backend.
3. Display mood inference result next to each entry.

Epic 6.4 — Planner Page

Tasks:

1. Weekly calendar grid.
2. Drag-and-drop tasks into time blocks.
3. Receive proposed slots from backend.

Epic 6.5 — Settings Page

Tasks:

1. Preferences (work hours, notification style).
2. Privacy & consent toggles.

3. Account deletion.

📊 PHASE 7 — INSIGHTS & ANALYTICS

Goal: Dashboard metrics to evaluate improvement over time.

Epic 7.1 — Mood Trends

Tasks:

1. Plot mood time series (valence, arousal).
2. Weekly averages.

Epic 7.2 — Productivity Insights

Tasks:

1. Completion rate vs. mood.
2. Hours scheduled vs. hours completed.

📏 PHASE 8 — EVALUATION & POLISH

Epic 8.1 — A/B Testing

Tasks:

1. Flags for mood-augmented vs. normal prioritization.
2. Track metrics: on-time submissions, user satisfaction.

Epic 8.2 — App Hardening

Tasks:

1. Add caching for mood inference.
2. Add circuit breakers + retries for Brightspace API.
3. Add Sentry logging.

★ PHASE 9 — DEPLOYMENT

Goal: Real users can test it.

Epic 9.1 — Deploy

Tasks:

1. Deploy backend (Railway or Render).
2. Deploy frontend (Vercel).
3. Run migrations.