Full-Stack Analytics Platform – Task Breakdown

Project: Customer Spending Analytics Dashboard

Author: Sankeerth Sridhar Narayan
Version: 1.0
Date: June 25, 2025

🔰 Phase 1: Project Initialization

- 📂 Setup Repo & Structure
- - [ ] Create GitHub repo (`analytics-dashboard`)
- - [ ] Create core folders: `/frontend`, `/backend`, `/db`
- - [ ] Add `.gitignore` for Python, VSCode, and Docker
- - [ ] Create `.env` file with secrets and DB config
- - [ ] Initialize `README.md` with project overview
- 🧪 Git Setup
- - [ ] Create branches: `main`, `test`, `dev`
- - [ ] Configure branch protection rules (if needed)
- - [ ] Set up basic commit/PR guidelines (conventional commits)

🐳 Phase 2: Docker Environment Setup

- ⚙️ Docker & Compose
- - [ ] Write `Dockerfile` for backend (FastAPI)
- - [ ] Write `Dockerfile` for frontend (Panel)
- - [ ] Set up PostgreSQL service in Docker
- - [ ] Create `docker-compose.yml` to orchestrate services
- - [ ] Add volume for persistent DB data
- - [ ] Add healthcheck for backend & db
- - [ ] Test `docker-compose up` locally

📃 Phase 3: Database Design & Seeding

- 🧱 Schema Creation
- - [ ] Design SQL schema: `users`, `transactions`
- - [ ] Write `init.sql` to prefill users and mock transaction data
- - [ ] Load `init.sql` via Docker volume
- - [ ] Test DB connection from backend

🔙 Phase 4: Backend (FastAPI) Setup

- 📦 FastAPI Project Structure
- - [ ] Create main FastAPI app (`main.py`)
- - [ ] Set up routing structure: `auth`, `transactions`, `metrics`
- - [ ] Create pydantic models for requests/responses
- - [ ] Use SQLAlchemy for ORM + DB session
- 🔐 Auth Module
- - [ ] Build `/login` endpoint with JWT generation
- - [ ] Create user model and auth logic
- - [ ] Add token validation middleware
- 📈 API Endpoints
- - [ ] `/transactions`: Return filtered data for user
- - [ ] `/metrics`: Return aggregated stats (total, average, top categories)

🎨 Phase 5: Frontend (Panel) Setup

- 🖼️ Dashboard Layout
- - [ ] Set up basic Panel app with `pn.template`
- - [ ] Build login screen → capture username/password
- - [ ] Store token in session state
- 📊 Visual Components
- - [ ] Add summary metrics (total, average, top categories)
- - [ ] Add filters: date range, category
- - [ ] Add charts using Panel + Holoviews/Bokeh (bar, pie, line)
- 🔌 API Integration
- - [ ] Call `/login` on login form submit
- - [ ] Fetch `/transactions` and `/metrics` with token
- - [ ] Re-render UI on filter changes

🧪 Phase 6: Testing

- ✅ Backend Unit Tests
- - [ ] Write test for `/login` (valid/invalid users)
- - [ ] Write test for `/transactions` with filters
- - [ ] Write test for `/metrics` output
- - [ ] Add coverage with `pytest-cov`
- ✅ Frontend Functional Checks
- - [ ] Validate API calls return expected data
- - [ ] Ensure filter selections update charts
- - [ ] Test session handling (token reuse, logout)

🔁 Phase 7: CI/CD with GitHub Actions

- 🧱 CI Pipeline

- - [ ] Create `.github/workflows/main.yml`
- - [ ] Add step for Python setup
- - [ ] Install backend + frontend requirements
- - [ ] Run `flake8` on both folders
- - [ ] Run `pytest` on backend/frontend
- - [ ] Run `black --check .` for formatting
- 🛠️ Lint & Format Tools
- - [ ] Add `flake8` config
- - [ ] Add `black` config
- - [ ] Add `pytest.ini` if needed

📦 Phase 8: QA + Future Extensions

- 🧹 Cleanup & Polish
- - [ ] Add README instructions for local setup
- - [ ] Add sample user credentials
- - [ ] Tag `v1.0` release
- - [ ] Archive and document learnings
- 🌱 Future Ideas
- - [ ] Add new user signup
- - [ ] Add more analytics KPIs (monthly change, transactions over time)
- - [ ] Extend CI to build Docker images
- - [ ] Plan for cloud deployment (Render, EC2, etc.)