.

**🧠 Phase 6 — *IndiaInflation Data Intelligence Platform Execution Plan***

**🧩 SECTION A – DATA & INFRASTRUCTURE LAYER**

| **Task ID** | **Task** | **Deliverable** | **Suggested Tool** |
| --- | --- | --- | --- |
| **A1** | Create unified .env.prod structure for ETL and backend | Environment file with DB, API keys, and base URLs | Manual / Cursor |
| **A2** | Build PostgreSQL schema for all inflation datasets | Tables: cpi\_national, cpi\_state, cpi\_components, wpi, expectations | Cursor / Claude Code |
| **A3** | Create etl/ folder with modular Python scripts | Separate file for each source (MOSPI, Data.gov, DPIIT, IMF, RBI) | Factory |
| **A4** | Implement ETL for **MOSPI CPI (All India)** | Script that downloads, cleans, loads CPI into DB | Factory |
| **A5** | Implement ETL for **CPI (Rural/Urban)** | Extend A4 to handle region dimension | Factory |
| **A6** | Implement ETL for **CPI (State-wise)** | Data.gov API or CSV loader | Factory |
| **A7** | Implement ETL for **WPI (Wholesale Price Index)** | DPIIT CSV → PostgreSQL | Factory |
| **A8** | Implement ETL for **IMF CPI All Items** | IMF SDDS → Normalized schema | Factory |
| **A9** | Implement ETL for **RBI Inflation Expectations Survey** | Parse PDF/Excel → load into expectations | Factory |
| **A10** | Write unified ETL orchestrator script | Cron-compatible CLI or Factory workflow that runs all ETL pipelines sequentially | Factory |
| **A11** | Build /api/health and /api/sync-status endpoints | Return ETL status + DB freshness | Claude Code |
| **A12** | Verify all ETL scripts populate data and confirm normalization | SQL queries or DB visual check | Manual |

**🎨 SECTION B – FRONTEND FEATURES**

| **Task ID** | **Task** | **Deliverable** | **Suggested Tool** |
| --- | --- | --- | --- |
| **B1** | Build **Inflation Calculator** component | Inputs: ₹ amount, start date, end date → output adjusted amount using CPI | Claude Code |
| **B2** | Build /api/inflation endpoint | Accept query params (amount, from, to) → return adjusted values | Cursor |
| **B3** | Build **Current Inflation Dashboard** | Hero section with current YoY inflation, monthly CPI trend, and chart | Claude Code |
| **B4** | Build **Historical Inflation Page** | Table view (1958–present), search, filter, CSV export | Claude Code |
| **B5** | Build **Core vs Headline Inflation** toggle view | Use CPI components data | Claude Code |
| **B6** | Build **State-wise Inflation Comparison Page** | Interactive heatmap + top 5/lowest 5 states table | Claude Code |
| **B7** | Build **Items Inflation Tracker** | Show inflation trends for food, housing, fuel, etc. | Claude Code |
| **B8** | Add **Release Schedule Page** | Static page listing MOSPI/DPIIT release dates | Manual |
| **B9** | Implement **CSV export + chart screenshot** | Export via button in dashboard and historical page | Cursor |
| **B10** | Integrate **ETL freshness indicator** in footer | “Data last updated: dd-mm-yyyy” | Claude Code |
| **B11** | Add global theme + fonts | Inter + Lexend; light blue-gray palette | Claude Code |
| **B12** | Polish UI with Next.js 15 best practices | Animate hero, optimize layout | Claude Code |

**📰 SECTION C – CONTENT & SEO**

| **Task ID** | **Task** | **Deliverable** | **Suggested Tool** |
| --- | --- | --- | --- |
| **C1** | Draft pillar article: *Inflation in India 2025 Overview* | 1500+ words + charts | ChatGPT / Claude |
| **C2** | Draft pillar article: *India Inflation Calculator Explained* | Tutorial + tool embed | ChatGPT / Claude |
| **C3** | Draft pillar article: *CPI vs WPI Explained Simply* | Educational + visual | ChatGPT |
| **C4** | Draft supporting article: *Food Inflation in India* | Data + insights | ChatGPT |
| **C5** | Draft supporting article: *Core vs Headline Inflation India* | Analytical | ChatGPT |
| **C6** | Draft supporting article: *Inflation & RBI Policy* | Commentary + context | ChatGPT |
| **C7** | Draft evergreen article: *Historical Inflation in India (1958–2025)* | Long data-driven post | ChatGPT |
| **C8** | Create SEO meta templates | auto-meta-title + description schema for blog posts | Cursor |
| **C9** | Generate sitemap.xml + robots.txt | With Next.js dynamic routes | Claude Code |
| **C10** | Implement JSON-LD structured data | Organization + WebSite + Article schema | Cursor |
| **C11** | Verify with Google Search Console | Add sitemap, fix errors | Manual |
| **C12** | Integrate GA4 tracking | Add GA4 env var + layout snippet | Cursor |

**📈 SECTION D – ANALYTICS, REPORTS, & AUTOMATION**

| **Task ID** | **Task** | **Deliverable** | **Suggested Tool** |
| --- | --- | --- | --- |
| **D1** | Implement cron ETL automation | Nightly update (Factory schedule or server cron) | Factory |
| **D2** | Build /api/report/monthly-summary | Returns latest CPI & WPI summary JSON | Cursor |
| **D3** | Generate reports/phase6-validation.json | Contains ETL freshness, data sources, and stats | Cursor |
| **D4** | Create internal dashboard (for admin) | Monitor ETL health + API uptime | Claude Code |
| **D5** | Auto backup PostgreSQL daily | Cron + pg\_dump + S3 or local | Factory |
| **D6** | Certbot auto-renew verification | Monthly cron | Manual |
| **D7** | Weekly sitemap + analytics report | Email to you via cron | Factory |

**🌍 SECTION E – BRAND, POSITIONING & GROWTH**

| **Task ID** | **Task** | **Deliverable** | **Suggested Tool** |
| --- | --- | --- | --- |
| **E1** | Finalize logo + favicon | Modern, minimal data-centric | Manual / Design |
| **E2** | Update tagline | “India’s trusted source for inflation data & insights” | Manual |
| **E3** | Add about page with transparency note | Data sources, methodology | ChatGPT |
| **E4** | Add “Press & API” page | Announce API access (future) | ChatGPT |
| **E5** | Set up LinkedIn + Medium for articles | Regular posting of updates | Manual |
| **E6** | Build newsletter / alert signup | For monthly inflation updates | Claude Code |
| **E7** | Launch “Inflation Report – Monthly PDF” | Auto-generated monthly insight | Factory + ChatGPT |

**⚙️ Execution Strategy**

Here’s how to split tasks across tools immediately:

| **Tool** | **Focus Area** | **Ideal Tasks** |
| --- | --- | --- |
| 🏗️ **Factory** | ETL, automation, cron jobs | A3–A10, D1, D5, D7 |
| 💻 **Claude Code** | Frontend (React/Next.js), UI components | B1–B12 |
| 🧠 **Cursor** | API routes, schema integration, structured automation | A2, A11, B2, B9, C8–C10, D2–D3 |
| ✍️ **ChatGPT (me)** | Architecture, SEO, copywriting, QA, validation | C1–C7, E3–E7 |

**🇮🇳 IndiaInflation Master Execution Flow**

**(Phase 6: From Raw Data → Live Intelligence Platform)**

Each step here represents a *stage* of development that must be completed before the next one starts.  
Every stage will produce deliverables and Factory / Claude-ready prompts.

**🔹 STAGE 1 — DATA FOUNDATIONS (The Backbone)**

Goal: Set up a robust, consistent data layer — ETL + PostgreSQL — that all dashboards and articles depend on.

**1.1 Data Source Confirmation**

🧩 Deliverable → *Official data source index file (docs/data\_sources.md)*

**Tasks:**

* List URLs, API keys, and formats for:
  + MOSPI (CPI & Components)
  + Data.gov.in (State CPI)
  + DPIIT (WPI)
  + IMF (CPI Global)
  + RBI (Expectations)
* Note update frequency and expected format (CSV, XLS, API).
* Output: one master reference document.

➡️ *Prompt for Factory*:

“Generate a structured documentation file listing data sources, URLs, and update frequency for all CPI/WPI datasets used in IndiaInflation.”

**1.2 Database Schema**

🧩 Deliverable → db/schema.sql and db/views.sql

**Tasks:**

* Create normalized PostgreSQL schema (tables: cpi\_national, cpi\_state, cpi\_components, wpi, inflation\_expectations).
* Define indexes + relationships.
* Add reusable SQL views (inflation\_summary, statewise\_latest).

➡️ *Prompt for Cursor*:

“Write optimized PostgreSQL schema and sample views for CPI, WPI, and expectations datasets for analytics dashboard.”

**1.3 ETL Architecture Setup**

🧩 Deliverable → Folder /etl with individual source ETL scripts.

**Tasks:**

* Create one ETL per data source (MOSPI, DPIIT, IMF, etc.).
* Central orchestrator (etl\_orchestrator.py).
* Config via .env.prod.
* Logging + error handling.

➡️ *Prompt for Factory*:

“Create modular ETL architecture with Python scripts for each dataset. Use pandas + SQLAlchemy, and a central orchestrator to run all ETLs sequentially.”

**1.4 ETL Validation**

🧩 Deliverable → reports/etl-validation.json

**Tasks:**

* Each ETL writes summary (rows inserted, last date, success/failure).
* Orchestrator aggregates into one validation report.

➡️ *Prompt for Factory*:

“Create post-run validation script to verify data freshness and save ETL summary as JSON.”

**🔹 STAGE 2 — API & BACKEND INTELLIGENCE**

Goal: Build all /api routes the frontend will need, using data from PostgreSQL.

**2.1 API Structure Setup**

🧩 Deliverable → /pages/api/ folder with:

* /api/inflation
* /api/cpi-dashboard
* /api/compare
* /api/state
* /api/health

➡️ *Prompt for Cursor / Claude Code*:

“Create REST API routes in Next.js 15 that fetch data from PostgreSQL and return JSON responses for inflation dashboard and calculator.”

**2.2 API Validation**

🧩 Deliverable → scripts/api\_test.sh

**Tasks:**

* Write shell or JS script to test all endpoints.
* Ensure /api/health returns 200 OK.
* Store outputs as reports/api-health.json.

➡️ *Prompt for Cursor*:

“Create automated API validation test that checks all endpoints return valid JSON and 200 status.”

**2.3 Data Transformation Helpers**

🧩 Deliverable → /lib/dataUtils.js

**Tasks:**

* Write safeDate(), calculateInflation(), and formatNumber() utilities.
* Used across API and frontend.

➡️ *Prompt for Claude Code*:

“Implement a reusable Next.js helper file with functions for date normalization, inflation calculation, and number formatting.”

**🔹 STAGE 3 — FRONTEND EXPERIENCE**

Goal: Build public-facing dashboards, calculator, and visualization layers.

**3.1 Core Calculator**

🧩 Deliverable → /components/InflationCalculator.tsx

**Features:**

* ₹ input, start year, end year
* Inflation-adjusted result
* Backend integration /api/inflation

➡️ *Prompt for Claude Code*:

“Build a responsive React component for inflation calculator using Inter font and blue-gray theme. Fetch CPI data from /api/inflation.”

**3.2 Dashboards**

🧩 Deliverables → /pages/inflation-dashboard.tsx, /pages/historical.tsx, /pages/statewise.tsx

**Features:**

* Charts (Recharts or Plotly)
* Table exports
* Filters and tooltips

➡️ *Prompt for Claude Code*:

“Create three interactive pages for CPI dashboard, historical inflation, and state-wise heatmap using Recharts.”

**3.3 UX Enhancements**

🧩 Deliverable → /components/Layout.tsx

**Tasks:**

* Consistent header, footer, theme.
* Add Data last updated indicator.

➡️ *Prompt for Claude Code*:

“Build layout wrapper with consistent theme, footer showing ETL update timestamp, and responsive design.”

**🔹 STAGE 4 — CONTENT & SEO**

Goal: Create authority through high-quality, data-rich articles and metadata.

**4.1 Core Articles (Pillar Content)**

🧩 Deliverables → 7 pillar articles  
Titles:

* Inflation in India 2025 Overview
* Inflation Calculator Explained
* CPI vs WPI Explained
* Historical Inflation in India
* Food Inflation 2025 Trends
* Core vs Headline Inflation
* RBI Inflation Outlook

➡️ *Prompt for ChatGPT / Claude*:

“Write a long-form, SEO-optimized article (1200+ words) with data visual insights for ‘Inflation in India 2025 Overview’.”

**4.2 SEO Infrastructure**

🧩 Deliverables → sitemap.xml, robots.txt, meta system

➡️ *Prompt for Cursor*:

“Generate sitemap.xml and robots.txt dynamically in Next.js. Add default metadata for all pages.”

**4.3 Analytics Integration**

🧩 Deliverable → GA4 + Search Console setup

➡️ *Prompt for Claude Code*:

“Add GA4 tracking script using NEXT\_PUBLIC\_GA\_ID and verify page load events fire correctly.”

**🔹 STAGE 5 — INSIGHTS & AUTOMATION**

Goal: Generate reports, analytics summaries, and user-facing insights.

**5.1 Monthly Report Generator**

🧩 Deliverable → /scripts/generate\_report.py

➡️ *Prompt for Factory*:

“Create a script that generates a monthly inflation summary (headline, core, food) and exports it to JSON and PDF formats.”

**5.2 Backup + Renew**

🧩 Deliverable →

* scripts/backup\_postgres.sh
* Certbot auto-renew cron job

➡️ *Prompt for Factory*:

“Automate daily PostgreSQL backup and SSL certificate renewal using cron.”

**5.3 Validation Report**

🧩 Deliverable → reports/phase6-validation.json  
Contains:

* ETL timestamps
* API uptime
* Sitemap status
* SSL expiry

➡️ *Prompt for Factory*:

“Create a validation script that compiles ETL, API, and SSL status into one JSON file for Phase 6 reporting.”

**🔹 STAGE 6 — BRAND & GROWTH**

Goal: Position IndiaInflation as *the* authority in its domain.

**6.1 About Page & Transparency Note**

🧩 Deliverable → /pages/about.tsx

➡️ *Prompt for ChatGPT*:

“Write About page explaining IndiaInflation’s mission, methodology, and data transparency.”

**6.2 Newsletter Setup**

🧩 Deliverable → /components/NewsletterSignup.tsx

➡️ *Prompt for Claude Code*:

“Build a simple newsletter signup form with email validation using React and Tailwind.”

**6.3 Public API Documentation (Future)**

🧩 Deliverable → /pages/api-docs.tsx

➡️ *Prompt for Claude Code*:

“Generate API documentation page listing available endpoints, parameters, and sample responses.”

**✅ Summary of the Flow**

| **Stage** | **Focus** | **Outcome** |
| --- | --- | --- |
| 1 | Data & ETL | Database + automated data pipelines |
| 2 | API Layer | Structured endpoints for frontend |
| 3 | Frontend | Interactive dashboards & calculators |
| 4 | SEO & Content | Authority-building articles + traffic |
| 5 | Automation | Monthly insights, backups, validation |
| 6 | Brand & Growth | Trust, transparency, and visibility |

This flow ensures **logical continuity** — every next step builds upon a working output from the previous one — and can be **executed by AI tools in parallel**.