Project Title: AI-Powered Menstrual Wellness Companion -ARIVAI

Problem Statement

Most menstrual tracking applications today are limited to logging cycle dates and predicting periods. They lack personalization, context-awareness, and emotional intelligence.
Women experience diverse symptoms — physical (cramps, bloating, fatigue) and emotional (mood swings, anxiety) — that differ across each phase of their menstrual cycle.
However, no existing platform **analyzes these symptoms intelligently** or provides **personalized health, nutrition, and mindfulness guidance** based on biological phases.

Moreover, reproductive health topics like **pregnancy, PMS, and menopause** remain underrepresented and often lack trustworthy digital resources.

There is a need for an **AI-driven menstrual wellness companion** that combines **cycle tracking, symptom understanding, emotional support, and mindful living** — all in one privacy-respecting platform.

Proposed Solution

Arivai is an AI-based menstrual wellness app that goes beyond tracking.
It acts as a **personal wellness agent** that understands user inputs, interprets symptoms, and provides relevant advice such as:

- Healthy snack recipes for cravings 🍫
- Meditation and mindfulness videos (youtube links and preview)🧘
- Educational content on pregnancy, sexual wellness, and menopause 💬

The above mentioned are three separate tabs. One more tab the main page contains user details and calendar for her menstrual phase cycle.

Arivai intelligently adapts to each user's menstrual phase (menstrual, follicular, ovulation, luteal) and provides **phase-based guidance** using both rule-based logic and AI-powered reasoning.

Each user has a **personal agent instance** that remembers their previous symptoms, moods, and responses, ensuring a **continuous personalized experience**.

SDG Alignment

- **Primary:** SDG 3 – Good Health and Well-Being
  → Promotes menstrual, mental, and nutritional wellness through AI support.

- **Secondary:** SDG 5 – Gender Equality
  → Uses technology to empower women with reproductive health awareness.

Tech Stack

Layer	Technology Used	Purpose
Frontend	**React.js** (Your main UI framework) **TailwindCSS** (Fast styling + supports black/grey/burgundy palette easily) **React Router** (Navigation between: Recipes | Meditation | Education | Chat Agent) **Axios** (For API calls to your backend) **React Query** *(optional but recommended)* (Caching + smooth UI updates)	UI for cycle tracking, chatting with agent, mindfulness and recipe tabs
Backend	Python (Flask or FastAPI)	Handles user data, agent logic, and connects APIs

Database	PostgressSQL / MongoDB	Stores user profiles, chat history, cycle info, and symptoms
AI API	Gemini API or OpenAI GPT API	For reasoning, symptom explanation, and generating advice
Recipe API	Edamam API / Spoonacular API	Fetches healthy snack and meal suggestions
YouTube Data API	Google API	Fetches meditation and mindfulness videos
Authentication	Firebase Auth / JWT	For secure user login and personalization
Storage	AWS S3	Saves agent interaction history and preferences

Final Tech Stack Summary

Frontend

- React.js
- TailwindCSS
- Axios
- React Router

Backend

- FastAPI (Python)
- LangChain (Agent)
- Ollama (free LLM backend)

- SQLAlchemy
- Pydantic

Database

- PostgreSQL (RDS)

Content APIs

- Spoonacular (snacks)
- YouTube Data API (meditation videos)
- Internal JSON files for educational content

Authentication

- Firebase Auth or JWT

Deployment

- React → S3 + CloudFront
- Backend (FastAPI + Agent + Ollama) → AWS EC2
- Database → AWS RDS PostgreSQL

Agent System (FREE)

- LangChain Agents
- Memory: PostgresChatMessageHistory
- Reasoning model: LLaMA 3.1 / Phi-3 / Mistral via Ollama

Final Tech Stack for Your Project (Clear-Cut & Minimal) —--- detailed workflow of the tech stack

Frontend

React.js

- Reason: Best for modern UI, lots of prebuilt components.
- Your theme: **Black + Grey + Burgundy** (easy with TailwindCSS)
- Libraries:
    - **TailwindCSS** (fast styling)
    - **React Router** (for navigation between 3 tabs)
    - **Axios** (API calls to backend)

Agent System (NO paid LLMs, ONLY FREE options)

You said: **Agent ≠ LLM** — you want rule-based / open-source reasoning + memory.

✔️ Use: LangChain (FREE) + Open-source model

LangChain is **not an LLM**.
It is a **framework to create agents**, tools, memory, and control logic.

For the model (FREE):

Choose any **open-source small model** such as:

- **Ollama + LLaMA 3.1 (8B)** → Runs locally on a server OR EC2 instance.
- **Phi-3 Mini** (free, optimized for reasoning)
- **Mistral 7B**

🚫 These models are TOTALLY FREE.
🚫 You do NOT pay per token.
✔️ Perfect for making an agent that responds + reasons.

How your agent works:

- LangChain Agent
- Tools:
    - Symptom knowledge base (your own JSON)
    - Cycle history stored in Postgres
    - Recipes API tool
    - YouTube search tool
- Memory:
    - LangChain's **PostgresChatMessageHistory**
      → keeps individual user's conversation history

This means:
Each user = separate agent memory stored in Postgres.

Backend

Python + FastAPI

- Fastest + cleanest for APIs
- Works perfectly with LangChain + open-source models

Backend jobs:

- User authentication
- Chat agent endpoint
- Manage cycle tracking data
- Serve recipe results
- Serve YouTube video links
- Store user preferences + history

Additional libs:

- **LangChain** (agent logic)
- **Ollama Python client** (if running free models)
- **SQLAlchemy** (database ORM)
- **Pydantic** (data validation)

Database

PostgreSQL

- Yes, totally fine
- Stores:
    - User profile

- Cycle info
- Symptoms
- Agent memory history
- Saved preferences (favourites, recipes, videos)

Content APIs (All FREE)

🍫 Healthy Snacks

Use free/cheap recipe APIs:

- **Spoonacular (free tier)**
  OR
- Build your own database of 50–100 snack recipes (no API needed)

🧘‍♀️ Meditation videos

Use:

- **YouTube Data API (FREE)**
  Fetch:
- Title
- Thumbnail
- Channel
- Link

💬 Educational content

- Stored internally as markdown or JSON

- No API required
- Delivered through backend

Authentication

Use one of these (both free):

- **Firebase Auth** (easy)
- **JWT tokens** (backend-generated)

Deployment on AWS (Simple & Cheap)

💡 Recommended Setup:

1️⃣ Frontend

- Deploy React on:
    - **AWS S3 + CloudFront** (super cheap, CDN)

2️⃣ Backend (FastAPI + Agent)

- Deploy on:
    - **AWS EC2 t3.small** (cheap & can run open-source models using Ollama)
    - OR **AWS ECS (Fargate)** if you want containers

3️⃣ Database

- Use:
    - **AWS RDS PostgreSQL** (free tier available)

4️⃣ Model hosting

- Install **Ollama** on the EC2 instance
- Pull a free model:

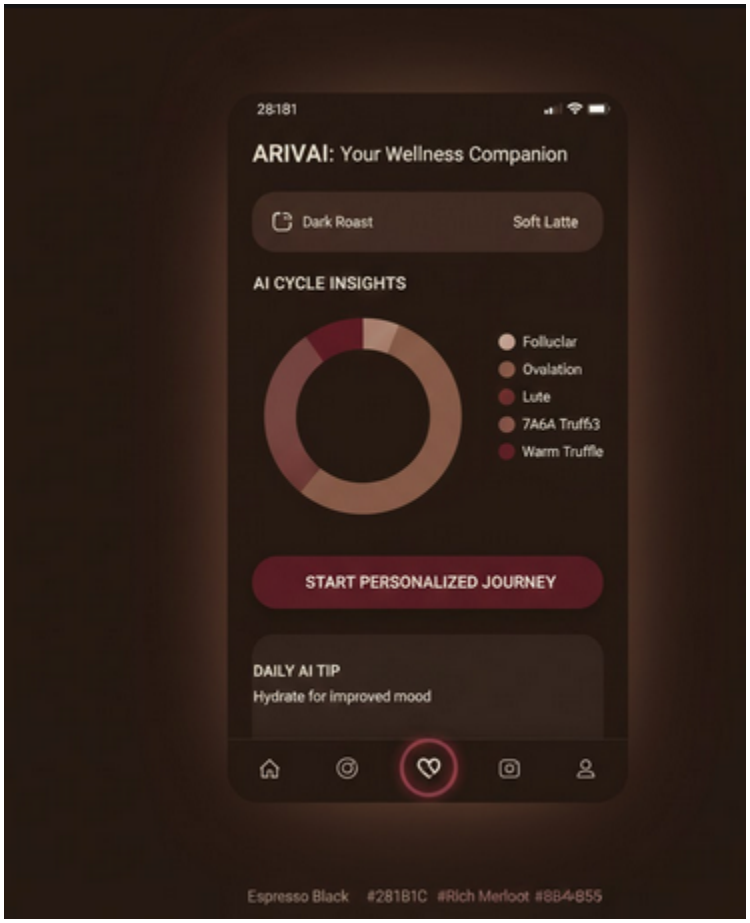
```
ollama pull llama3.1
```

-
- Connect it to LangChain in FastAPI backend

This avoids ALL paid APIs.

Color	Name	Hex Code	Purpose in UI
Deep Background	**Espresso Black**	#281B1C	**Main Screen Background, Navigation Bars.** A very dark, rich, almost black-brown that provides a luxurious and deep foundation.
Card & Container	**Dark Roast**	#4A3C3B	**Container Cards, Panels, Secondary UI Elements.** Offers a slightly lighter, warm brown for depth and separation.
Primary Accent	**Rich Merlot**	#8B4B5B	**Call-to-Action (CTA) Buttons, Key AI Insights, Active State/Toggle.** A deep, sophisticated wine-red that provides a strong, elegant focal point.

Text & Icons	**Soft Latte**	#EFE8E5	**Primary Text, Headings, Icons.** Ensures excellent readability and provides a gentle, creamy contrast against the dark backgrounds.
Data Visualization	**Cabernet Stone**	#6D4D4F	**Secondary Data Points, Charts (e.g., Follicular Phase).** A muted, earthy wine-brown for detailed information, blending subtly.
Secondary Data	**Warm Truffle**	#7A6A63	**Mood Tracking, Cravings, or Luteal Phase data.** A calming, medium brown that balances the wine tones.



2. CALCULATE CURRENT CYCLE DAY

Formula:

```
cycleDay = (today - lastPeriodStartDate) + 1
```

Example:
Period started Jan 1.
Today Jan 10 → cycleDay = 10.

---

3. MENSTRUAL PHASE LOGIC (FIXED RULE)

Phase	Day Range	Notes
Menstrual	Day 1–5	bleeding
Follicular	Day 6–12	energy rising
Ovulation	Day 13–15	highest fertility
Luteal	Day 16–28	PMS likely here

Dynamic ovulation calculation:

```
ovulationDay = cycleLength - 14
```

Then:

```
Ovulation window = ovulationDay - 1  to ovulationDay + 1
```

If irregular cycles → widen window to ±3 days.

---

4. PMS CALCULATION

PMS window is:

```
PMS = cycleLength - 7  to cycleLength - 1
```

Example:
28-day cycle → PMS = Days 21–27.

If user reports symptoms (craving, mood swings, fatigue):
→ Shift PMS window earlier by 2 days.

---

5. NEXT PERIOD PREDICTION

```
nextPeriodDate = lastPeriodStartDate + cycleLength
```

If irregular cycles:

```
predictRange = ± 2 days
```

---

6. PREGNANCY DURATION

If user selects "I missed my period" or inputs LMP:

Use standard obstetrics formula:

```
GestationalAgeWeeks = floor( (today - lastPeriodStartDate) / 7 )
GestationalAgeDays = (today - lastPeriodStartDate) % 7
DueDate = lastPeriodStartDate + 280 days
```

ARIVAI must explain trimester:

- Trimester 1 = 0–13 weeks

- Trimester 2 = 14–27 weeks
- Trimester 3 = 28–40 weeks

---

7. MENOPAUSE LOGIC

Perimenopause detection rule:

If user is 40+ AND
`cycleLength` varies by **≥ 7 days month-to-month**, mark as:

```
perimenopause_likely = true
```

Menopause definition:

If **no period for 12 months**:

```
menopause = true
```

---

8. ABNORMAL CYCLE DETECTION

Trigger alerts if:

```
period > 8 days
cycleLength < 21
cycleLength > 35
```

or

```
bleedingBetweenPeriods = true
```

ARIVAI must give educational, safe, non-medical-diagnostic guidance.

---

9. RETURN PAYLOAD FORMAT (VERY IMPORTANT)

Every calculation response must return **this JSON** to frontend:

```
{
  "cycleDay": number,
  "phase": "Menstrual" | "Follicular" | "Ovulation" | "Luteal",
  "pmsWindow": { "startDay": number, "endDay": number },
  "nextPeriodDate": "YYYY-MM-DD",
  "ovulationDay": number,
  "pregnancy": {
      "isPregnant": boolean,
      "weeks": number,
      "days": number,
      "dueDate": "YYYY-MM-DD"
  },
  "menopause": {
      "perimenopauseLikely": boolean,
      "menopause": boolean
  },
  "dailyAdvice": {
      "mood": string,
      "nutrition": string,
      "meditation": string,
      "exercise": string
  }
}
```

---

10. DAILY ADVICE RULES

Menstrual

- Warm foods
- Hydration
- Low-intensity yoga
- Pain-relief lifestyle tips

Follicular

- High-energy recipes
- Cardio or dance
- Light meditation

Ovulation

- High-protein foods
- Strength exercises
- Confidence/mood boosting

Luteal

- Magnesium-rich foods
- PMS support
- Sleep & slow yoga

ARIVAI must generate these based on phase.

---

11. EDUCATIONAL CONTENT (YOUTUBE/RECIPES)

If frontend requests:

```
GET /recipes?phase=Luteal
GET /meditation?phase=Menstrual
```

Return the appropriate content based on phase.

---

12. MEMORY RULES

Agent must remember:

- symptoms
- cravings
- mood patterns
- previous cycle length
- previous period dates

Stored in Postgres using:

```
PostgresChatMessageHistory
```

END OF SYSTEM PROMPT

✅ CALENDAR LOGIC (for React Frontend)

Use this exact UI logic:

1. Highlight period days

```
for i from 0 to periodLength-1:
    mark lastPeriodStartDate + i as "period"
```

2. Highlight fertile window

```
ovulationDay = cycleLength - 14
fertileWindowStart = lastPeriodStartDate + (ovulationDay - 3)
fertileWindowEnd   = lastPeriodStartDate + (ovulationDay + 1)
```

3. Highlight PMS window

```
pmsStart = lastPeriodStartDate + (cycleLength - 7)
pmsEnd   = lastPeriodStartDate + (cycleLength - 1)
```

4. Next period

Mark predicted next date with burgundy dot.