📄 Problem Statement

Design and implement a backend microservice for a **peer-to-peer (P2P) wallet transfer system** for a Fintech platform. The system must support:

- Secure user authentication
- Daily and monthly transfer limits
- Caching for performance optimization

The entire application must be containerized using **Docker**, and should use:

- A **relational database** (of your choice)
- A **caching layer** (e.g., Redis, Memcached, etc.)
- A **backend programming language and framework** of your choice

---

✅ Deliverables

Implement the following features and submit the code in a **GitHub repository**, including:

- `README.md` with setup and usage instructions
- A `Dockerfile` and `docker-compose.yml` to spin up the full environment
- All services (backend, database, cache) must run via Docker containers

---

🔐 1. Authentication

Implement secure user authentication using token-based authentication (e.g., JWT or equivalent):

- `POST /auth/signup` – Create a new user
- `POST /auth/login` – Login and receive an access token

All subsequent wallet and transfer endpoints must require valid authentication.

💰 2. Wallet and Transfer APIs

All endpoints below must be protected via authentication:

- `POST /wallets` – Create a wallet for the logged-in user
- `POST /wallets/{wallet_id}/add-funds` – Add funds to the user's wallet
- `POST /wallets/{wallet_id}/transfer` – Transfer funds to another user's wallet
- `GET /wallets/{wallet_id}/transactions` – List transactions (with filters like date range, transaction type, etc.)

🗃️ 3. Relational Database (via Docker Image)

Use any **relational database**, modeled to include at minimum:

- Users
- Wallets
- Transactions
- Transfer limits (daily and monthly)

The database must run using an **official Docker image** via Docker Compose.

⚡ 4. Caching Layer (via Docker Image)

Use a **caching system** (e.g., Redis or Memcached) to:

- Cache wallet balances for faster reads
- Automatically update or invalidate the cache on balance changes

The cache must also run via an **official Docker image**.

🔐 5. Daily and Monthly Transfer Limits

- Each user must have **daily** and **monthly** configurable transfer limits
- Transfers that exceed these limits must be rejected with an appropriate error response

---

🧠 Assumptions

If any part of the requirement is unclear, **assume the most logical or industry-standard approach**. You are encouraged to make reasonable design decisions and clearly document them in the `README.md`.