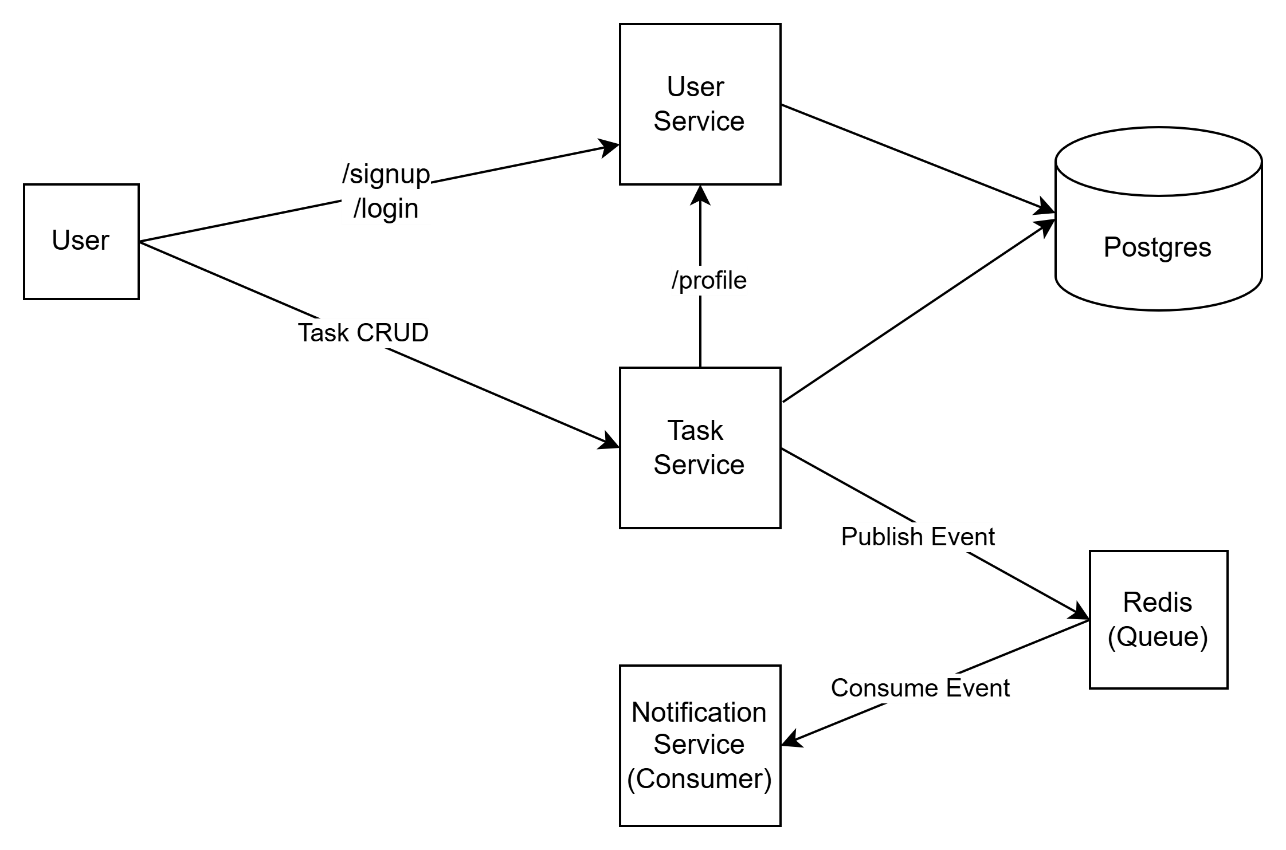
**High-Level Design and Documentation for Task Management Microservices Application**

1. **Architecture Diagram:**



**2. Database Schema:**

**Users Table (in public schema)**

users (

id SERIAL PRIMARY KEY,

username VARCHAR(50) UNIQUE NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

password TEXT NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

)

**Tasks Table (in taskservice schema)**

taskservice.tasks (

id SERIAL PRIMARY KEY,

title VARCHAR(100) NOT NULL,

description TEXT,

status VARCHAR(20) DEFAULT 'pending',

due\_date DATE,

user\_id INTEGER NOT NULL REFERENCES users(id),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

)

**3. Sequence Diagram**

**A black screen with white text

AI-generated content may be incorrect.**

**4. Deployment Steps:**

1. Provision a PostgreSQL database server instance.
2. Provision a Redis cache server instance.
3. Clone the repository onto the deployment server using Git.
4. Update the environment variables in all .env files located in the env directory as follows:
   1. REDIS\_URL = redis://<username>:<password>@<host>:<port>

Replace <username>, <password>, <host>, and <port> with the credentials of the Redis instance created in step 2.

* 1. DATABASE\_URL = postgresql://<username>:<password>@<host>:<port>/<db\_name>

*Replace <username>, <password>, <host>, <port>, and <db\_name> with the connection details of the PostgreSQL instance created in step 1.*

1. Run command “docker compose up” to build the docker images and run the containers.

**5. Design Choices:**

**Scalability:**

* Stateless services allow horizontal scaling using Docker Swarm or Kubernetes.
* Redis Pub/Sub ensures decoupled communication.

**Resiliency:**

* Connection pooling in DB for load handling.
* Retry and error handling in DB and Redis ops.
* Graceful degradation in Notification service.

**Security:**

* JWT-based authentication ensures secure access.
* Passwords are hashed using bcrypt.

**Best Practices:**

* Redis decouples task and notification services.
* Keeping service configurable using environment configs (.env files)

**Optimized DB Querying:**

* Indexing via primary keys.
* Queries scoped by user\_id to reduce load.