Runtime guidance with Resource links

1.1 Prerequisites

Before deploying the system, ensure you have the following installed:

- Node.js (v16 or later)
- MongoDB (local or cloud instance)
- Git
- Postman (for API testing)
- Docker (for containerized deployment)

1.2 Cloning the Repository

Run the following command to clone the project repository:

git clone https://github.com/faisalmc/TCH-PIS.git

cd TCH-PIS

1.3 Setting Up Environment Variables

Each microservice requires environment variables for configuration.

1. Create an .env file in each service directory (auth-service, patient-registration, patient-treatment).

2. Add the following variables inside .env:

MONGO_URI=mongodb+srv://fmc4000:PcMk4CYYjrNQ9LW8@patient-info-system.qj5ku.mongodb.net/patientdb?retryWrites=true&w=majority

JWT_SECRET=e5b8a6d8f92c

1.4 Installing Dependencies

Navigate to the directory of each service and install the required dependencies:

```
cd auth-service
npm install
cd ../patient-registration
npm install
cd ../patient-treatment
npm install
```

1.5 Running Each Microservice

Start the Authentication Service

```
cd auth-service/src
node server.js
```

Start the Patient Registration Service

```
cd ../patient-registration/src
node server.js
```

Start the Patient Treatment Service

```
cd ../patient-treatment/src
node server.js
```

Each service should now be running on its respective port (3000, 3001, 3002).

1.6 Running MongoDB Locally (Optional)

If using a local MongoDB instance, start it with:

```
mongod --dbpath /data/db
```

Or use Docker to run MongoDB:

```
docker run -d --name mongodb -p 27017:27017 -e
MONGO_INITDB_ROOT_USERNAME=admin -e
MONGO_INITDB_ROOT_PASSWORD=admin mongo
```

1.7 API Testing with Postman

- Open Postman.
- Import the provided Postman Collection.
- Test endpoints:
  - Authentication: POST /auth/register, POST /auth/login
  - Patient Registration: POST /patients/register, GET /patients/all
  - Patient Treatment: POST /treatment/diagnosis, GET /treatment/:patientID

1.8 Dockerizing the Services (Further Implementation in Ops Phase)

To run services inside Docker containers, create a Dockerfile for each service:

Example Dockerfile for Auth Service

```
FROM node:16
WORKDIR /app
COPY package.json .
RUN npm install
COPY . .
CMD ["npm", "run", "dev"]
EXPOSE 3000
```

Then, build and run the container:

```
docker build -t auth-service .
docker run -p 3000:3000 auth-service
```

1.9 Stopping the Services

To stop the services, use:

CTRL + C # Stop a running process

Or if running with Docker:

docker stop <container_id>

2.0 Resources Links

• GitHub repository

https://github.com/faisalmc/TCH-PIS

• MongoDB

URI - mongodb+srv://fmc4000:PcMk4CYYjrNQ9LW8@patient-info-system.qj5ku.mongodb.net/patientdb?retryWrites=true&w=majority

JWT secret - e5b8a6d8f92c

• Service APIs – Postman

https://f21ao-group.postman.co/workspace/TCH-PIS~f0968d44-b64a-4c37-b89c-4ac3cccd9974/collection/41575727-3be3f492-99d6-46af-ab3d-8d4422def8fc

• Continuous Testing – Postman

https://f21ao-group.postman.co/workspace/TCH-PIS~f0968d44-b64a-4c37-b89c-4ac3cccd9974/collection/41575727-f92d10b6-d533-485f-be82-30a302bdf3d4