Docker Setup for PSScript

This document explains how to run the PSScript application using Docker, both for development and production environments.

Prerequisites

- Docker
- Docker Compose
- Git (for cloning the repository)

Project Structure

The application consists of several services:

- **Frontend**: React application running on port 3002
- **Backend**: Node.js API service running on port 4000
- **AI Service**: Python service for AI analysis running on port 8000
- **PostgreSQL**: Database with pgvector extension for vector search
- **Redis**: Caching service

Environment Setup

1. Clone the repository:

`bash git clone https://github.com/YOUR_USERNAME/psscript.git cd psscript`

1. Create a `.env` file based on the example:

`bash cp .env.example .env`

1. Edit the `.env` file to set your environment variables, especially:
2. `OPENAI_API_KEY`: Your OpenAI API key
3. `JWT_SECRET`: A secure secret for JWT authentication
4. `DB_PASSWORD`: Database password

Development Environment

To run the application in development mode:

```
docker-compose up -d
```

This will start all services with hot-reloading enabled.

Production Environment

To run the application in production mode:

```
./docker-deploy.sh
```

This script will:

1. Load environment variables from `.env`
2. Build and start all services in production mode
3. Verify that all services are running correctly

Alternatively, you can run:

```
docker-compose -f docker-compose.prod.yml up -d
```

Accessing the Application

- Frontend: http://localhost:3002
- Backend API: http://localhost:4000
- API Documentation: http://localhost:4000/api-docs

Troubleshooting

If you encounter any issues:

1. Check the logs:

```bash # All services docker-compose logs

# Specific service docker-compose logs frontend ```

1. Restart services:

`bash docker-compose restart`

1. Rebuild services:

`bash docker-compose up -d --build`
```

Notes

- The frontend automatically connects to the backend using the dynamic hostname configuration
- The backend API is available at `http://${hostname}:4000/api`
- Mock mode can be enabled by setting `MOCK_MODE=true` in the `.env` file

Stopping the Application

```
# Development
docker-compose down

# Production
docker-compose -f docker-compose.prod.yml down
```

To remove volumes (this will delete all data):

```bash docker-compose down -v
```