# API Setup Guide

Complete guide to setting up the RAS Dashboard API from scratch, including database configuration, environment setup, and initial data seeding.

## 🚀 Quick Setup

# 1. Install dependencies  
cd api  
npm install  
  
# 2. Configure environment  
cp .env.example .env  
# Edit .env with your database credentials  
  
# 3. Test database connection  
npm run db:query tables  
  
# 4. Seed database with default data  
npm run db:seed all --verbose  
  
# 5. Start development server  
npm run dev

## 📋 Prerequisites

### Required Software

* **Node.js** (v16 or higher)
* **PostgreSQL** (v12 or higher)
* **npm** or **yarn**

### Database Requirements

* PostgreSQL database (local or cloud)
* Database user with CREATE, INSERT, UPDATE, DELETE permissions
* SSL support (for cloud databases like AWS RDS)

## 🔧 Detailed Setup

### Step 1: Install Dependencies

cd api  
npm install

**Key Dependencies Installed:** - express - Web framework - drizzle-orm - Database ORM - postgres - PostgreSQL client - bcryptjs - Password hashing - jsonwebtoken - JWT authentication - swagger-jsdoc & swagger-ui-express - API documentation

### Step 2: Environment Configuration

Create your environment file:

cp .env.example .env

Edit .env with your configuration:

# Server Configuration  
PORT=3001  
NODE\_ENV=development  
  
# Database Configuration (Example for AWS RDS)  
DATABASE\_URL=postgresql://username:password@host:5432/database  
PGHOST=your-db-host.amazonaws.com  
PGPORT=5432  
PGUSER=your\_username  
PGPASSWORD=your\_password  
PGDATABASE=your\_database\_name  
  
# JWT Configuration  
JWT\_SECRET=your-super-secret-jwt-key-change-this-in-production  
JWT\_EXPIRES\_IN=24h  
  
# CORS Configuration  
CORS\_ORIGIN=http://localhost:3000  
  
# Rate Limiting  
RATE\_LIMIT\_WINDOW\_MS=900000  
RATE\_LIMIT\_MAX\_REQUESTS=100  
  
# File Upload  
MAX\_FILE\_SIZE=10485760

### Step 3: Database Connection

Test your database connection:

npm run db:query tables

**Expected Output:**

🔍 Running query: tables  
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━  
📊 Database Tables:  
┌─────────┬─────────────────┬────────────┐  
│ (index) │ table\_name │ table\_type │  
├─────────┼─────────────────┼────────────┤  
│ 0 │ 'users' │ 'BASE' │  
│ 1 │ 'roles' │ 'BASE' │  
│ 2 │ 'permissions' │ 'BASE' │  
└─────────┴─────────────────┴────────────┘  
✅ Query completed successfully

### Step 4: Database Schema Verification

Check your database schema matches expectations:

npm run db:query schema

**Required Tables:** - users - User accounts - roles - User roles (admin, user, moderator, etc.) - permissions - System permissions - role\_permissions - Role-permission assignments - user\_roles - User-role assignments - user\_preferences - User preferences (optional)

### Step 5: Seed Default Data

Populate your database with default data:

npm run db:seed all --verbose

**What Gets Created:** - **12 Permissions** across 5 categories (users, roles, admin, system, reports) - **4 Roles** (admin, user, moderator, viewer) - **Role-Permission Assignments** (admin gets all, user gets basic, etc.) - **2 Default Users** (admin and test user)

### Step 6: Verify Setup

Check that everything was created correctly:

# Check users  
npm run db:query users  
  
# Check role assignments  
npm run db:query role-permissions  
  
# Check user-role assignments  
npm run db:query user-roles

### Step 7: Start Development Server

npm run dev

**Expected Output:**

🔄 Testing database connection...  
✅ Database connection established successfully  
🚀 Server is running on port 3001  
📊 Environment: development  
🔗 Health check: http://localhost:3001/health  
📚 API Base URL: http://localhost:3001/api/v1

## 🌐 API Endpoints

### Health Check

curl http://localhost:3001/health

### API Documentation

Visit: http://localhost:3001/api-docs

### Authentication Endpoints

# Register new user  
POST /api/v1/auth/register  
  
# Login  
POST /api/v1/auth/login  
  
# Get profile (requires auth)  
GET /api/v1/auth/profile

### User Management

# Get all users (requires auth)  
GET /api/v1/users  
  
# Get user by ID  
GET /api/v1/users/:id  
  
# Create user  
POST /api/v1/users  
  
# Update user  
PUT /api/v1/users/:id  
  
# Delete user  
DELETE /api/v1/users/:id

## 🧪 Testing the Setup

### Test Authentication

# Register a new user  
curl -X POST http://localhost:3001/api/v1/auth/register \  
 -H "Content-Type: application/json" \  
 -d '{  
 "email": "test@example.com",  
 "username": "testuser",  
 "password": "TestPass123!",  
 "firstName": "Test",  
 "lastName": "User"  
 }'  
  
# Login with seeded admin user  
curl -X POST http://localhost:3001/api/v1/auth/login \  
 -H "Content-Type: application/json" \  
 -d '{  
 "email": "admin@rasdash.com",  
 "password": "Admin123!"  
 }'

### Test Protected Endpoints

# Get access token from login response, then:  
curl -X GET http://localhost:3001/api/v1/users \  
 -H "Authorization: Bearer YOUR\_ACCESS\_TOKEN"

## 🔧 Troubleshooting

### Database Connection Issues

**Error:** Connection failed: password authentication failed

# Check your credentials in .env  
npm run db:query tables

**Error:** Connection failed: SSL required

# For cloud databases, ensure SSL is configured  
# Check DATABASE\_URL includes SSL parameters

### Port Already in Use

# Change port in .env  
PORT=3002  
  
# Or kill process using port 3001  
lsof -ti:3001 | xargs kill -9

### Permission Errors

# Ensure database user has proper permissions  
GRANT ALL PRIVILEGES ON DATABASE your\_db TO your\_user;  
GRANT ALL ON ALL TABLES IN SCHEMA public TO your\_user;

### Seeding Failures

# Check for existing data conflicts  
npm run db:remove-duplicates --show  
  
# Force recreate role permissions  
npm run db:seed role-permissions --force

## 🔒 Security Configuration

### JWT Secret

# Generate a secure JWT secret  
node -e "console.log(require('crypto').randomBytes(64).toString('hex'))"

### Password Requirements

Default validation requires: - Minimum 8 characters - At least one uppercase letter - At least one lowercase letter  
- At least one number - At least one special character

### Rate Limiting

Default configuration: - 100 requests per 15 minutes per IP - Configurable via RATE\_LIMIT\_\* environment variables

## 📁 Project Structure

api/  
├── src/  
│ ├── config/ # Configuration files  
│ ├── db/ # Database connection and schemas  
│ ├── controllers/ # Request handlers  
│ ├── routes/ # API route definitions  
│ ├── services/ # Business logic  
│ ├── middleware/ # Express middleware  
│ └── utils/ # Utility functions  
├── scripts/ # Database management scripts  
├── tests/ # Test files  
├── docs/ # Documentation  
├── .env # Environment variables  
├── package.json # Dependencies and scripts  
└── server.js # Application entry point

## 🚀 Production Deployment

### Environment Variables

NODE\_ENV=production  
PORT=3001  
DATABASE\_URL=your\_production\_database\_url  
JWT\_SECRET=your\_production\_jwt\_secret  
CORS\_ORIGIN=https://your-frontend-domain.com

### Database Setup

# Run migrations in production  
npm run db:seed permissions roles role-permissions  
  
# Don't seed test users in production  
# Create admin user manually or through secure process

### Process Management

# Using PM2  
npm install -g pm2  
pm2 start server.js --name "ras-api"  
  
# Using Docker  
docker build -t ras-api .  
docker run -p 3001:3001 ras-api

## 📚 Next Steps

1. **Customize Permissions** - Edit /api/scripts/db-seed.js to add your specific permissions
2. **Add Business Logic** - Create new controllers and services for your domain
3. **Implement Frontend** - Connect your React/Vue/Angular frontend to the API
4. **Add Tests** - Expand the test suite in /api/tests/
5. **Monitor Performance** - Add logging and monitoring tools

## 🔗 Related Documentation

* [Database Query Tool](./database-query-tool.md) - Query and inspect your database
* [Database Seeder](./database-seeder.md) - Manage default data
* [Duplicate Removal](./duplicate-removal.md) - Clean up duplicate records
* [RBAC Implementation](./rbac.md) - Role-based access control details

Your API is now ready for development! 🎉