API Contract v3

Overview

- Base URL (local): `http://localhost:3000`
- Content-Type: `application/json`
- Auth: Bearer JWT is required for `/tasks` routes. Obtain it via `/auth/signup` or `/auth/signin` and send `Authorization: Bearer <token>`.
- Success envelope: `{ "data": ... }`
- Error envelope: `{ "error": { "code": "...", "message": "...", "details": ... } }`
- IDs are UUID strings; timestamps are ISO 8601 strings.

Success Codes

Status	Meaning
200	OK - Request succeeded
201	Created - Resource created
204	No Content - Succeeded, no body

Error Codes

Code	Status	When
VALIDATION_ERROR	400	Invalid request body/params (Zod validation). `details` contains the issues.
UNAUTHORIZED	401	Missing or invalid Bearer token on protected routes.
INVALID_CREDENTIALS	401	Wrong email/password during sign-in.
USER_ALREADY_EXISTS	409	Email already registered during sign-up.
NOT_FOUND	404	Task not found or not owned by the user.
INTERNAL_ERROR	500	Unexpected server error.

Health

GET /health

- Response 200: `{"data":{"ok":true}}`

Auth

POST /auth/signup

- Body:

```
{
  "email": "user@example.com",
  "password": "secret123",
  "name": "John Doe"
}
```

- Validation: email is trimmed and lowercased; password 6-127 chars; name optional, trimmed, blank is dropped (stored as null).
- Fields:

Field	Type	Required	Notes
email	string	yes	Valid email, lowercased.
password	string	yes	6-127 chars.
name	string	no	1-255 chars; blank removed.

- Response 201:

```
{
  "data": {
    "token": "<jwt>",
    "user": {
      "id": "...",
      "email": "user@example.com",
      "name": "John Doe"
    }
  }
}
```

- Errors: 400 VALIDATION_ERROR, 409 USER_ALREADY_EXISTS.

POST /auth/signin

- Body:

```
{
  "email": "user@example.com",
  "password": "secret123"
}
```

- Fields:

Field	Type	Required	Notes
email	string	yes	Valid email, lowercased.
password	string	yes	6-127 chars.

- Response 200: same shape as sign-up (token + user info).

- Errors: 400 VALIDATION_ERROR, 401 INVALID_CREDENTIALS.

GET /auth/me

- Auth: required.
- Response 200:

```
{
  "data": {
    "id": "...",
    "email": "user@example.com",
    "name": "John Doe"
  }
}
```

- Errors: 401 UNAUTHORIZED, 404 NOT_FOUND.

Tasks (requires Authorization header)

Common: send `Authorization: Bearer <token>`. All operations are scoped to the authenticated user.

GET /tasks

- Returns the user's tasks ordered by `createdAt` desc.
- Response 200 example:

```
{
  "data": [
    {
      "id": "...",
      "title": "Test Task",
      "description": "This is my test task",
      "completed": false,
      "userId": "...",
      "createdAt": "2025-12-23T12:34:56.000Z",
      "updatedAt": "2025-12-23T12:34:56.000Z"
    }
  ]
}
```

POST /tasks

- Body:

```
{
  "title": "Buy milk",
  "description": "2% or oat"
}
```

- Fields:

Field	Type	Required	Notes
title	string	yes	Trimmed, 1-255 chars.
description	string	no	Trimmed; empty/whitespace -> null; max 1000 chars.

- Server sets `completed: false` and `userId` from the token.
- Response 201: returns the created task (id, title, description, completed, userId, createdAt, updatedAt).
- Errors: 400 VALIDATION_ERROR.

PATCH /tasks/{id}

- Path: `id` must be a UUID.
- Body (at least one field required):

```
{
  "title": "Buy oat milk",
  "description": "From Whole Foods",
  "completed": true
}
```

- Fields:

Field	Type	Required	Notes
title	string	no	Trimmed, 1-255 chars; omit to keep; null/blank not allowed.
description	string or null	no	Trimmed; empty/whitespace -> null; max 1000 chars.
completed	boolean	no	Completion status.

- Response 200: updated task with all fields.
- Errors: 400 VALIDATION_ERROR, 404 NOT_FOUND.

DELETE /tasks/{id}

- Path: `id` must be a UUID.
- Response 204 No Content.
- Errors: 404 NOT_FOUND.

GET /tasks/{id}

- Path: `id` must be a UUID.
- Response 200 example:.

```
{
  "data":
```

```
    {
      "id": "...",
      "title": "Test Task",
      "description": "This is my test task",
      "completed": false,
      "userId": "...",
      "createdAt": "2025-12-23T12:34:56.000Z",
      "updatedAt": "2025-12-23T12:34:56.000Z"
    }
}
```

- Errors: 404 NOT_FOUND.

Data Model (Prisma)

User

Field	Type	Required	Notes
id	string (UUID)	auto	Primary key.
email	string	yes	Unique, lowercased.
passwordHash	string	yes	Bcrypt hash.
name	string or null	no	Optional display name.
createdAt	string (ISO 8601)	auto	Creation timestamp.
updatedAt	string (ISO 8601)	auto	Last update timestamp.

Task

Field	Type	Required	Notes
id	string (UUID)	auto	Primary key.
title	string	yes	1-255 chars.
description	string or null	no	Up to 1000 chars; nullable.
completed	boolean	auto	Default false.
userId	string (UUID)	yes	Owner user id; indexed.
createdAt	string (ISO 8601)	auto	Creation timestamp.
updatedAt	string (ISO 8601)	auto	Last update timestamp.