

OpenAPI Contract – Phase 1 (Verbatim)

This document contains the full OpenAPI 3.0.3 contract for Phase 1 pantry and recipe CRUD, including shared error models and schemas.

OpenAPI 3.0.3 YAML

```
openapi: 3.0.3
info:
  title: Meal Planning App API
  version: 1.0.0-phasel
  description: >
    Phase 1 API contract for pantry and recipe CRUD.
    Spring Boot is system of record. MongoDB is canonical store.
servers:
  - url: http://localhost:8080
    description: Local development
tags:
  - name: Health
  - name: Pantry
  - name: Recipes

paths:
  /api/health:
    get:
      tags: [Health]
      summary: Health check
      operationId: getHealth
      responses:
        "200":
          description: Service is healthy
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/HealthResponse"

  /api/pantry:
    get:
      tags: [Pantry]
      summary: List pantry lots
      operationId: listPantryLots
      parameters:
        - in: query
          name: type
          required: false
          schema:
            $ref: "#/components/schemas/PantryLotType"
          description: Optional filter by lot type
        - in: query
          name: name
          required: false
          schema:
            type: string
            description: Optional substring match on name (simple contains)
        - in: query
          name: limit
          required: false
```

```

    schema:
      type: integer
      minimum: 1
      maximum: 200
      default: 50
  - in: query
    name: offset
    required: false
    schema:
      type: integer
      minimum: 0
      default: 0
responses:
  "200":
    description: Pantry lots
    content:
      application/json:
        schema:
          type: object
          properties:
            items:
              type: array
              items:
                $ref: "#/components/schemas/PantryLot"
            total:
              type: integer
          required: [items, total]
  "400":
    $ref: "#/components/responses/BadRequest"
  "500":
    $ref: "#/components/responses/InternalServerError"

post:
  tags: [Pantry]
  summary: Create pantry lot
  operationId: createPantryLot
  requestBody:
    required: true
    content:
      application/json:
        schema:
          $ref: "#/components/schemas/PantryLotCreateRequest"
  responses:
    "201":
      description: Created pantry lot
      content:
        application/json:
          schema:
            $ref: "#/components/schemas/PantryLot"
    "400":
      $ref: "#/components/responses/BadRequest"
    "409":
      $ref: "#/components/responses/Conflict"
    "500":
      $ref: "#/components/responses/InternalServerError"

/api/pantry/{id}:
  get:
    tags: [Pantry]
    summary: Get pantry lot by id
    operationId: getPantryLot
    parameters:
      - $ref: "#/components/parameters/IdPathParam"
    responses:

```

```

    "200":
      description: Pantry lot
      content:
        application/json:
          schema:
            $ref: "#/components/schemas/PantryLot"
    "404":
      $ref: "#/components/responses/NotFound"
    "500":
      $ref: "#/components/responses/InternalServerError"

put:
  tags: [Pantry]
  summary: Update pantry lot
  operationId: updatePantryLot
  parameters:
    - $ref: "#/components/parameters/IdPathParam"
  requestBody:
    required: true
    content:
      application/json:
        schema:
          $ref: "#/components/schemas/PantryLotUpdateRequest"
  responses:
    "200":
      description: Updated pantry lot
      content:
        application/json:
          schema:
            $ref: "#/components/schemas/PantryLot"
    "400":
      $ref: "#/components/responses/BadRequest"
    "404":
      $ref: "#/components/responses/NotFound"
    "409":
      $ref: "#/components/responses/Conflict"
    "500":
      $ref: "#/components/responses/InternalServerError"

delete:
  tags: [Pantry]
  summary: Delete pantry lot (manual only)
  operationId: deletePantryLot
  parameters:
    - $ref: "#/components/parameters/IdPathParam"
  responses:
    "204":
      description: Deleted
    "404":
      $ref: "#/components/responses/NotFound"
    "500":
      $ref: "#/components/responses/InternalServerError"

/api/recipes:
  get:
    tags: [Recipes]
    summary: List recipes
    operationId: listRecipes
    parameters:
      - in: query
        name: tag
        required: false
        schema:
          type: string

```

```

    description: Optional filter by single tag
  - in: query
    name: name
    required: false
    schema:
      type: string
    description: Optional substring match on name (simple contains)
  - in: query
    name: limit
    required: false
    schema:
      type: integer
      minimum: 1
      maximum: 200
      default: 50
  - in: query
    name: offset
    required: false
    schema:
      type: integer
      minimum: 0
      default: 0
responses:
  "200":
    description: Recipes
    content:
      application/json:
        schema:
          type: object
          properties:
            items:
              type: array
              items:
                $ref: "#/components/schemas/Recipe"
            total:
              type: integer
          required: [items, total]
  "400":
    $ref: "#/components/responses/BadRequest"
  "500":
    $ref: "#/components/responses/InternalServerError"

post:
  tags: [Recipes]
  summary: Create recipe
  operationId: createRecipe
  requestBody:
    required: true
    content:
      application/json:
        schema:
          $ref: "#/components/schemas/RecipeCreateRequest"
  responses:
    "201":
      description: Created recipe
      content:
        application/json:
          schema:
            $ref: "#/components/schemas/Recipe"
    "400":
      $ref: "#/components/responses/BadRequest"
    "409":
      $ref: "#/components/responses/Conflict"
    "500":

```

\$ref: "#/components/responses/InternalServerError"

/api/recipes/{id}:

get:

tags: [Recipes]
summary: Get recipe by id
operationId: getRecipe
parameters:
- \$ref: "#/components/parameters/IdPathParam"
responses:
"200":
description: Recipe
content:
application/json:
schema:
\$ref: "#/components/schemas/Recipe"
"404":
\$ref: "#/components/responses/NotFound"
"500":
\$ref: "#/components/responses/InternalServerError"

put:

tags: [Recipes]
summary: Update recipe
operationId: updateRecipe
parameters:
- \$ref: "#/components/parameters/IdPathParam"
requestBody:
required: true
content:
application/json:
schema:
\$ref: "#/components/schemas/RecipeUpdateRequest"
responses:
"200":
description: Updated recipe
content:
application/json:
schema:
\$ref: "#/components/schemas/Recipe"
"400":
\$ref: "#/components/responses/BadRequest"
"404":
\$ref: "#/components/responses/NotFound"
"409":
\$ref: "#/components/responses/Conflict"
"500":
\$ref: "#/components/responses/InternalServerError"

delete:

tags: [Recipes]
summary: Delete recipe
operationId: deleteRecipe
parameters:
- \$ref: "#/components/parameters/IdPathParam"
responses:
"204":
description: Deleted
"404":
\$ref: "#/components/responses/NotFound"
"500":
\$ref: "#/components/responses/InternalServerError"

components:

```

parameters:
  IdPathParam:
    in: path
    name: id
    required: true
    schema:
      type: string
    description: MongoDB ObjectId string

responses:
  BadRequest:
    description: Bad request
    content:
      application/json:
        schema:
          $ref: "#/components/schemas/ErrorResponse"
  NotFound:
    description: Not found
    content:
      application/json:
        schema:
          $ref: "#/components/schemas/ErrorResponse"
  Conflict:
    description: Conflict
    content:
      application/json:
        schema:
          $ref: "#/components/schemas/ErrorResponse"
  InternalError:
    description: Internal server error
    content:
      application/json:
        schema:
          $ref: "#/components/schemas/ErrorResponse"

schemas:
  HealthResponse:
    type: object
    properties:
      status:
        type: string
        enum: ["ok"]
      version:
        type: string
      time:
        type: string
        format: date-time
      required: [status, time]

  ErrorResponse:
    type: object
    properties:
      code:
        type: string
        description: Machine-readable error code
      message:
        type: string
        description: Human-readable error message
      details:
        type: object
        additionalProperties: true
        description: Optional structured details (field errors, etc.)
      traceId:
        type: string

```

description: Correlation ID for logs
required: [code, message]

PantryLotType:
type: string
enum: ["INGREDIENT", "PACKAGED"]

PantryLot:
type: object
properties:
 id:
 type: string
 name:
 type: string
 type:
 \$ref: "#/components/schemas/PantryLotType"
 quantity:
 type: number
 format: double
 minimum: 0
 unit:
 type: string
 description: User-entered unit (e.g., g, oz, lb, ml, count)
 metadata:
 type: object
 additionalProperties: true
 createdAt:
 type: string
 format: date-time
 updatedAt:
 type: string
 format: date-time
required: [id, name, type, quantity, unit, createdAt, updatedAt]

PantryLotCreateRequest:
type: object
properties:
 name:
 type: string
 minLength: 1
 type:
 \$ref: "#/components/schemas/PantryLotType"
 quantity:
 type: number
 format: double
 minimum: 0
 unit:
 type: string
 minLength: 1
 metadata:
 type: object
 additionalProperties: true
required: [name, type, quantity, unit]

PantryLotUpdateRequest:
type: object
properties:
 name:
 type: string
 minLength: 1
 type:
 \$ref: "#/components/schemas/PantryLotType"
 quantity:
 type: number

```
    format: double
    minimum: 0
  unit:
    type: string
    minLength: 1
  metadata:
    type: object
    additionalProperties: true
  required: [name, type, quantity, unit]
```

```
RecipeIngredient:
  type: object
  properties:
    name:
      type: string
      minLength: 1
    quantity:
      type: number
      format: double
      minimum: 0
    unit:
      type: string
      minLength: 1
  required: [name, quantity, unit]
```

```
Recipe:
  type: object
  properties:
    id:
      type: string
    name:
      type: string
    ingredients:
      type: array
      items:
        $ref: "#/components/schemas/RecipeIngredient"
    instructions:
      type: array
      items:
        type: string
    servings:
      type: integer
      minimum: 1
    tags:
      type: array
      items:
        type: string
    embedding:
      type: object
      description: Optional back-reference for vector indexing
      properties:
        qdrantCollection:
          type: string
        vectorId:
          type: string
        embeddedAt:
          type: string
          format: date-time
      additionalProperties: false
    createdAt:
      type: string
      format: date-time
    updatedAt:
      type: string
```



```
    format: date-time
  required: [id, name, ingredients, instructions, servings, tags, createdAt, updatedAt]
```

```
RecipeCreateRequest:
  type: object
  properties:
    name:
      type: string
      minLength: 1
    ingredients:
      type: array
      minItems: 1
      items:
        $ref: "#/components/schemas/RecipeIngredient"
    instructions:
      type: array
      minItems: 1
      items:
        type: string
    servings:
      type: integer
      minimum: 1
    tags:
      type: array
      items:
        type: string
  required: [name, ingredients, instructions, servings]
```

```
RecipeUpdateRequest:
  type: object
  properties:
    name:
      type: string
      minLength: 1
    ingredients:
      type: array
      minItems: 1
      items:
        $ref: "#/components/schemas/RecipeIngredient"
    instructions:
      type: array
      minItems: 1
      items:
        type: string
    servings:
      type: integer
      minimum: 1
    tags:
      type: array
      items:
        type: string
  required: [name, ingredients, instructions, servings]
```