**Week 3: Design: Service Layer(s)**

Wasan Yotchai

Maryville University

SWDV 691: Software Development Capstone 1W 25/SP2

Professor Joseph Gradecki

March 29, 2025

****

Design: service Layer(s)

Wasan Yotchai

March 29, 2025

COMMENT: NO CHANGE

**Service Layer Design for MVP Budget App**

**Overview**

The service layer for the Budget App acts as an intermediary between the user interface and the PostgreSQL database, ensuring proper data management and business logic implementation. This document defines the required service endpoints, their purpose, sample requests and responses, and a diagram illustrating UI interaction with these endpoints.

**Service Endpoints**

The following RESTful API endpoints will be used in the MVP Budget App:

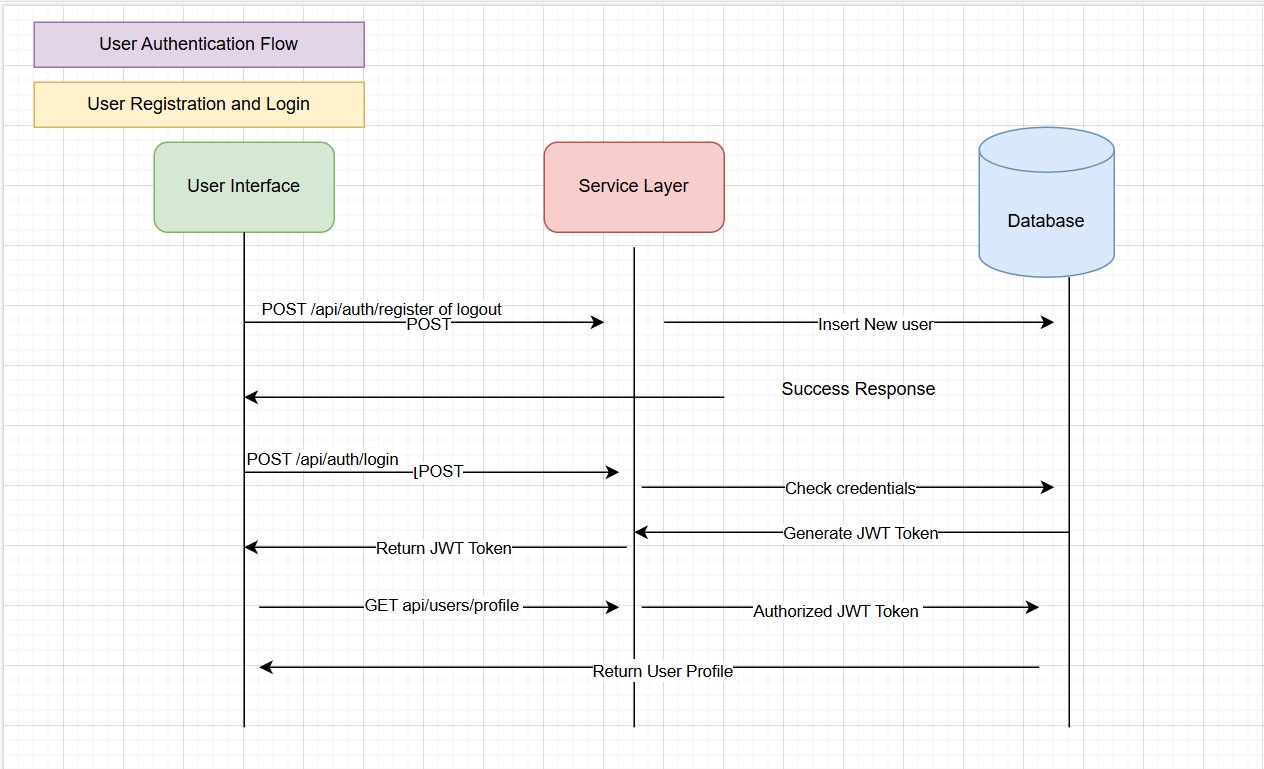
**1. User Authentication & Management**

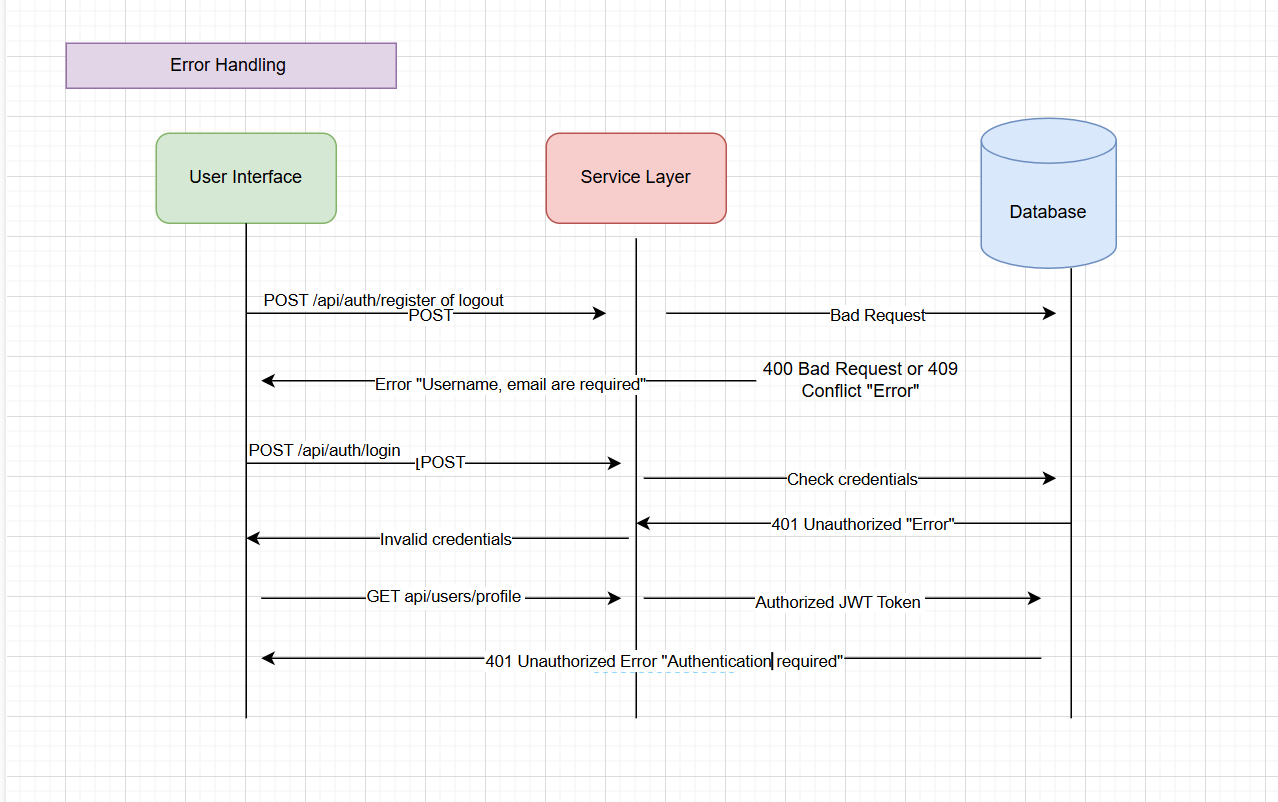
|  |  |  |  |
| --- | --- | --- | --- |
| **Endpoint** | **Method** | **Purpose** | **Request body/ Params** |
| /api/auth/register | POST | Registers a new user | { "name": "John", "email": "john@email.com", "password": "secure123" } |
| /api/auth/login | POST | Authenticates a user and returns a JWT | |  | | --- | |  |  |  | | --- | | { "email": "john@email.com", "password": "secure123" } | |
| /api/auth/logout | POST | Logs out the user | Bearer token |
| /api/users/profile | GET | Get user profile | Authorization: Bearer token |

**Successful and Error Handling example response**

|  |  |  |
| --- | --- | --- |
| **Scenario** | **Response (Success)** | **Response (failure)** |
| Register | 201 Created { "message": "User registered successfully" } | 400 Bad Request { "error": "Username, email, and password are required" }  409 Conflict { "error": "Email is already registered" } |
| Login | 200 OK { "token": "jwt\_token" } | 401 Unauthorized { "error": "Invalid credentials" } |
| Logout | 200 OK { "User logged out successfully" } | 400 Bad Request { "error": "Token is required for logout" } |
| Get user profile | 200 OK { "id": "123", "name": "John", "email": "john@email.com" } | 401 Unauthorized {"error": "Authentication required"} |

**Diagram**



****

**Register User Endpoint Details**

**Endpoint:** POST /api/auth/register

**Request Type:**

* **Method:** POST
* **Request Body:**
  + username (string, required)
  + email (string, required)
  + password (string, required)

**Request Example**

{

"username": "user123",

"email": "user@example.com",

"password": "SecurePass123!"

}

**Successful Response:**

* **Status Code:** 201 Created
* **Response Body:**

{

"message": "User registered successfully",

"userId": 1

}

**Failure Responses:**

**1. Missing Required Fields**

* **Status Code:** 400 Bad Request
* **Response Body:**

{

"error": "Username, email, and password are required"

}

**2. Email Already in Use**

* **Status Code:** 409 Conflict
* **Response Body:**

{

"error": "Email is already registered"

}

**3. Weak Password**

* **Status Code:** 400 Bad Request
* **Response Body:**

{

"error": "Password must be at least 8 characters long, contain a number and a special character"

}

**User Login**

**Endpoint:** POST /api/auth/login

**Request Type:**

* **Method:** POST
* **Request Body:**
  + email (string, required)
  + password (string, required)

**Request Example**

{

"email": "user@example.com",

"password": "SecurePass123!"

}

**Successful Response:**

* **Status Code:** 200 OK
* **Response Body:**

{

"message": "Login successful",

"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",

"user": {

"userId": 1,

"username": "user123",

"email": "user@example.com"

}

}

The token returned is a JWT token that should be used for authentication in protected routes.

**Failure Responses:**

**1. Missing Required Fields**

* **Status Code:** 400 Bad Request
* **Response Body:**

{

"error": "Email and password are required"

}

**2. Invalid Credentials (Incorrect Email or Password)**

* **Status Code:** 401 Unauthorized
* **Response Body:**

{

"error": "Invalid email or password"

}

**3. Account Not Verified (If Email Verification is Required)**

* **Status Code:** 403 Forbidden
* **Response Body:**

{

"error": "Account not verified. Please check your email to verify your account."

}

**Logout Endpoint Details**

**Endpoint:** POST /api/auth/logout

**Request Type:**

* **Method:** POST
* **Request Body:** (Optional) JWT token if required for invalidation
* **Headers:**
  + Authorization: Bearer <JWT\_TOKEN> (if token-based logout is needed)

**Request Example**

If the logout requires the token to be sent in the request body:

{

"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."

}

**Successful Response:**

* **Status Code:** 200 OK
* **Response Body:**

{

"message": "User logged out successfully"

}

**Failure Responses:**

**1. Missing or Invalid Token**

* **Status Code:** 400 Bad Request
* **Response Body:**

{

"error": "Token is required for logout"

}

**2. Unauthorized Token**

* **Status Code:** 401 Unauthorized
* **Response Body:**

{

"error": "Invalid or expired token"

}

**Get User Profile**

**Endpoint:** GET /api/users/me

**Request Type:**

* **Method:** GET
* **Headers:**
  + Authorization: Bearer <JWT\_TOKEN>

**Request Example:**

* **Headers:**

makefile

CopyEdit

Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

**Successful Response:**

* **Status Code:** 200 OK
* **Response Body:**

{

"userId": 1,

"username": "user123",

"email": "user@example.com",

"createdAt": "2025-03-14T12:00:00Z"

}

**Failure Responses:**

**1. Unauthorized Request (No Token Provided)**

* **Status Code:** 401 Unauthorized
* **Response Body:**

{

"error": "Authentication required"

}

**2. Invalid or Expired Token**

* **Status Code:** 401 Unauthorized
* **Response Body:**

{

"error": "Invalid or expired token"

}

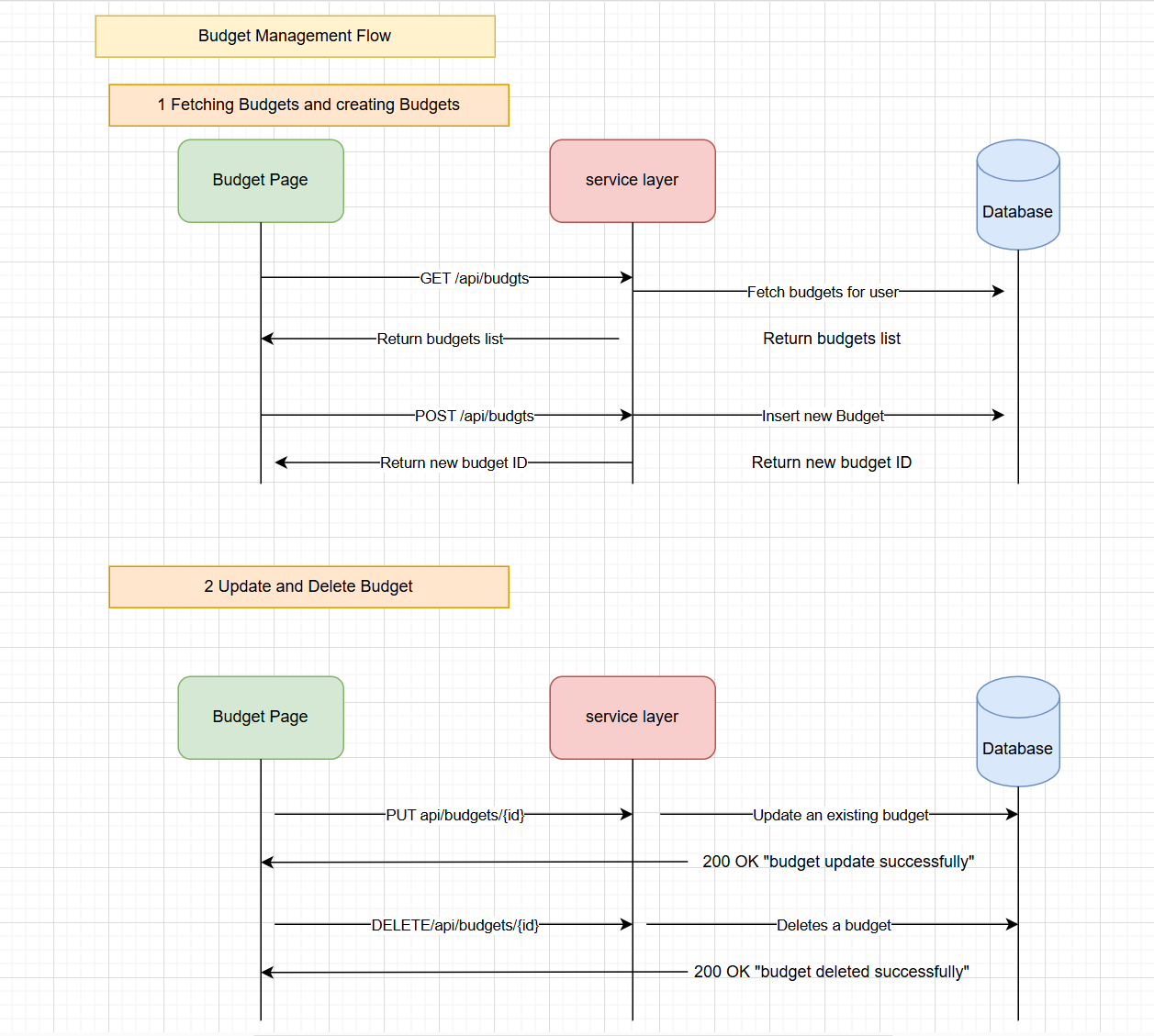
**2. Budget Management**

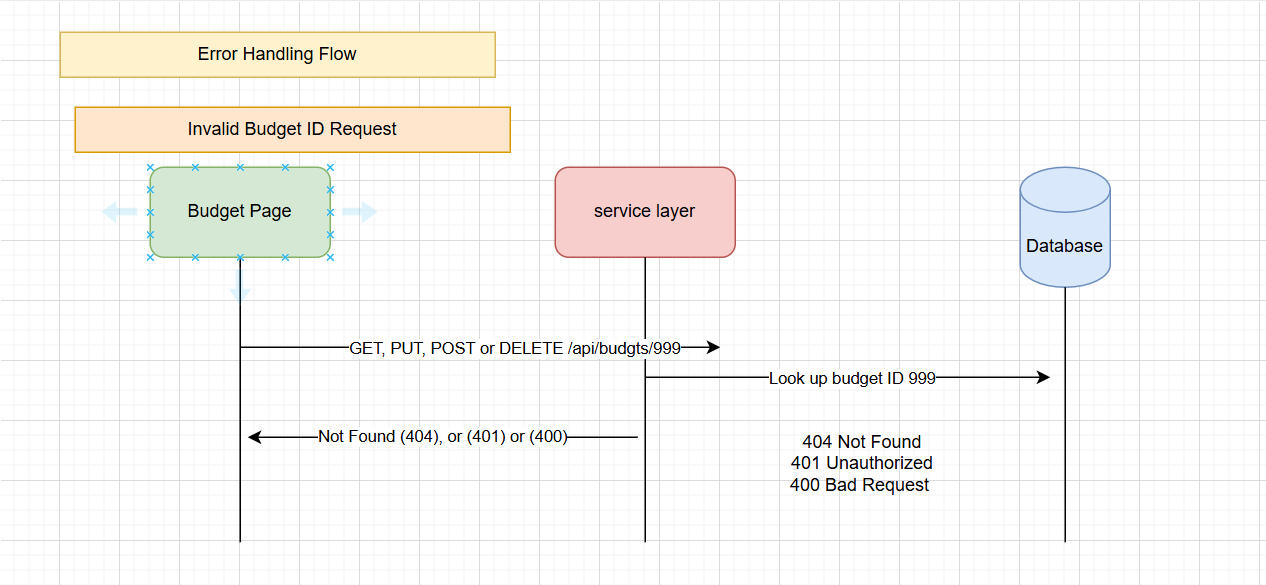
|  |  |  |  |
| --- | --- | --- | --- |
| **Endpoint** | **Method** | **Description** | **Request body/params** |
| /api/budgets | GET | Fetches all budgets for the user | No request body required |
| /api/budgets/{id} | GET | Retrieves a single budget | No request body required |
| /api/budgets | POST | Creates a new budget | {  "name": "Entertainment",  "category": "Leisure",  "amount": 200.00  } |
| /api/budgets/{id} | PUT | Updates an existing budget | {  "name": "Updated Grocery Budget",  "amount": 600.00  } |
| /api/budgets/{id} | DELETE | Deletes a budget | No request body required. |

**Successful and Error Handling example response**

|  |  |  |
| --- | --- | --- |
| **Scenario** | **Response (Success)** | **Response (failure)** |
| Get all Budgets | 200 OK { "token": "show all budgets by id" } | 401 Unauthorized { "error": "Authorization token required" } |
| Get single budget | 200 OK { "token": "show budget by id" } | 404 Not found{ "error": "Budget not found" } |
| Create a new budget | 201 Created {"message" "Budget crated successfully" } | 400 Bad Request { "error": "Name, category, and amount are required" } |
| Update budget | 200 OK { "message ""Budget update successfully" } | 404 Not found{ "error": "Budget not found" }  400 Bad Request {"error": "Amount must be a positive number"} |
| Delete budget | 200 OK { "message ""Budget deleted successfully" } | 404 Not found{ "error": "Budget not found" } |

**Diagram**

****

****

**1. Get All Budgets**

**Endpoint:** GET /api/budgets

**Request Type:**

* **Method:** GET
* **Headers:**
  + Authorization: Bearer <token>

**Request Example:**

No request body required.

**Successful Response:**

* **Status Code:** 200 OK
* **Response Body:**

{

"budgets": [

{

"id": 1,

"userId": 1,

"name": "Monthly Groceries",

"category": "Food",

"amount": 500.00,

"remaining": 250.00

},

{

"id": 2,

"userId": 1,

"name": "Rent",

"category": "Housing",

"amount": 1200.00,

"remaining": 0.00

}

]

}

**Failure Response (Unauthorized Request - Missing Token):**

* **Status Code:** 401 Unauthorized
* **Response Body:**

{

"error": "Authorization token required"

}

**2. Get a Single Budget**

**Endpoint:** GET /api/budgets/{id}

**Request Type:**

* **Method:** GET
* **Headers:**
  + Authorization: Bearer <token>

**Request Example:**

No request body required.

**Successful Response:**

* **Status Code:** 200 OK
* **Response Body:**

{

"id": 1,

"userId": 1,

"name": "Monthly Groceries",

"category": "Food",

"amount": 500.00,

"remaining": 250.00

}

**Failure Responses:**

* **404 Not Found (Budget does not exist)**

{

"error": "Budget not found"

}

* **403 Forbidden (Trying to access another user’s budget)**

{

"error": "Unauthorized access to budget"

}

**3. Create a New Budget**

**Endpoint:** POST /api/budgets

**Request Type:**

* **Method:** POST
* **Headers:**
  + Authorization: Bearer <token>
  + Content-Type: application/json

**Request Example:**

{

"name": "Entertainment",

"category": "Leisure",

"amount": 200.00

}

**Successful Response:**

* **Status Code:** 201 Created
* **Response Body:**

{

"message": "Budget created successfully",

"budget": {

"id": 3,

"userId": 1,

"name": "Entertainment",

"category": "Leisure",

"amount": 200.00,

"remaining": 200.00

}

}

**Failure Responses:**

* **400 Bad Request (Missing required fields)**

{

"error": "Name, category, and amount are required"

}

**4. Update an Existing Budget**

**Endpoint:** PUT /api/budgets/{id}

**Request Type:**

* **Method:** PUT
* **Headers:**
  + Authorization: Bearer <token>
  + Content-Type: application/json

**Request Example:**

{

"name": "Updated Grocery Budget",

"amount": 600.00

}

**Successful Response:**

* **Status Code:** 200 OK
* **Response Body:**

{

"message": "Budget updated successfully",

"budget": {

"id": 1,

"userId": 1,

"name": "Updated Grocery Budget",

"category": "Food",

"amount": 600.00,

"remaining": 350.00

}

}

**Failure Responses:**

* **404 Not Found (Budget does not exist)**

{

"error": "Budget not found"

}

* **403 Forbidden (Trying to update another user’s budget)**

{

"error": "Unauthorized access to update budget"

}

* **400 Bad Request (Invalid amount value)**

{

"error": "Amount must be a positive number"

}

**5. Delete a Budget**

**Endpoint:** DELETE /api/budgets/{id}

**Request Type:**

* **Method:** DELETE
* **Headers:**
  + Authorization: Bearer <token>

**Request Example:**

No request body required.

**Successful Response:**

* **Status Code:** 200 OK
* **Response Body:**

{

"message": "Budget deleted successfully"

}

**Failure Responses:**

* **404 Not Found (Budget does not exist)**

{

"error": "Budget not found"

}

* **403 Forbidden (Trying to delete another user’s budget)**

{

"error": "Unauthorized access to delete budget"

}

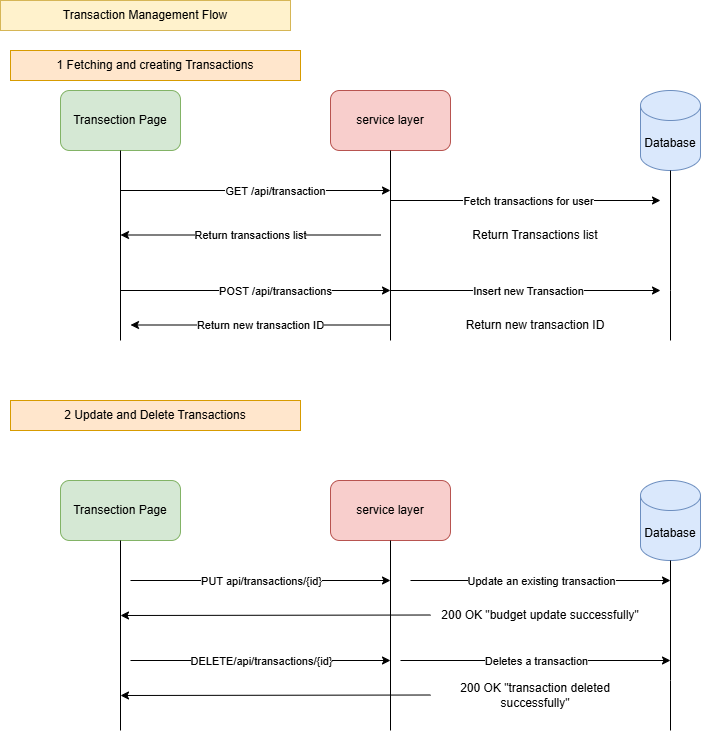
**Transaction Management**

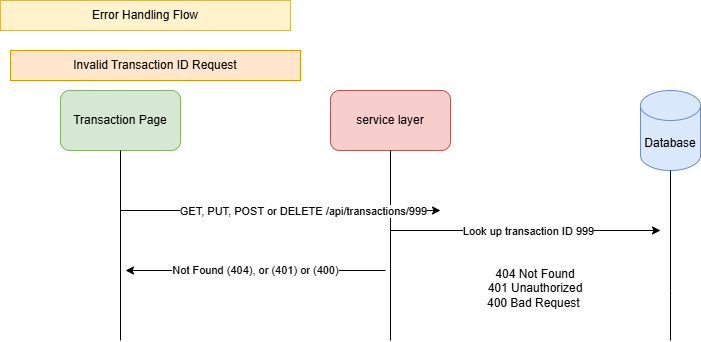
|  |  |  |  |
| --- | --- | --- | --- |
| **Endpoint** | **Method** | **Description** | **Request body/Params** |
| /api/transactions | GET | Fetches all transactions for the user | No request body required. |
| /api/transactions/{id} | GET | Retrieves a specific transaction | No request body required. |
| /api/transactions | POST | Creates a new transaction | {  "budgetId": 2,  "description": "Dinner at a restaurant",  "category": "Food",  "amount": 45.99,  "date": "2025-03-14T19:00:00Z"  } |
| /api/transactions/{id} | PUT | Updates a transaction | {  "description": "Updated Grocery Shopping",  "amount": 80.00  } |
| /api/transactions/{id} | DELETE | Deletes a transaction | No request body required. |

**Successful and Error Handling example response**

|  |  |  |
| --- | --- | --- |
| **Scenario** | **Response (Success)** | **Response (failure)** |
| Get all transactions | 200 OK { "token": "show all transaction by id" } | 401 Unauthorized { "error": "Authorization token required" } |
| Get single transaction | 200 OK { "token": "show budget by id" } | 404 Not found{ "error": "Budget not found" } |
| Create a new budget | 201 Created {"message" "transaction crated successfully" } | 400 Bad Request { "error": "Budget ID, category, and amount are required" } |
| Update budget | 200 OK { "message " "Transaction update successfully" } | 404 Not found{ "error": "Transaction not found" }  400 Bad Request {"error": "Amount must be a positive number"} |
| Delete budget | 200 OK { "message " "Transaction deleted successfully" } | 404 Not found{ "error": "Transaction not found" } |

**Diagram:**





**1. Get All Transactions**

**Endpoint:** GET /api/transactions

**Request Type:**

* **Method:** GET
* **Headers:**
  + Authorization: Bearer <token>

**Request Example:**

No request body required.

**Successful Response:**

* **Status Code:** 200 OK
* **Response Body:**

{

"transactions": [

{

"id": 1,

"userId": 1,

"budgetId": 2,

"description": "Grocery Shopping",

"category": "Food",

"amount": 75.00,

"date": "2025-03-14T10:30:00Z"

},

{

"id": 2,

"userId": 1,

"budgetId": 3,

"description": "Gym Membership",

"category": "Fitness",

"amount": 50.00,

"date": "2025-03-13T15:45:00Z"

}

]

}

**Failure Response (Unauthorized Request - Missing Token):**

* **Status Code:** 401 Unauthorized
* **Response Body:**

{

"error": "Authorization token required"

}

**2. Get a Specific Transaction**

**Endpoint:** GET /api/transactions/{id}

**Request Type:**

* **Method:** GET
* **Headers:**
  + Authorization: Bearer <token>

**Request Example:**

No request body required.

**Successful Response:**

* **Status Code:** 200 OK
* **Response Body:**

{

"id": 1,

"userId": 1,

"budgetId": 2,

"description": "Grocery Shopping",

"category": "Food",

"amount": 75.00,

"date": "2025-03-14T10:30:00Z"

}

**Failure Responses:**

* **404 Not Found (Transaction does not exist)**

{

"error": "Transaction not found"

}

* **403 Forbidden (Trying to access another user’s transaction)**

{

"error": "Unauthorized access to transaction"

}

**3. Create a New Transaction**

**Endpoint:** POST /api/transactions

**Request Type:**

* **Method:** POST
* **Headers:**
  + Authorization: Bearer <token>
  + Content-Type: application/json

**Request Example:**

{

"budgetId": 2,

"description": "Dinner at a restaurant",

"category": "Food",

"amount": 45.99,

"date": "2025-03-14T19:00:00Z"

}

**Successful Response:**

* **Status Code:** 201 Created
* **Response Body:**

{

"message": "Transaction created successfully",

"transaction": {

"id": 3,

"userId": 1,

"budgetId": 2,

"description": "Dinner at a restaurant",

"category": "Food",

"amount": 45.99,

"date": "2025-03-14T19:00:00Z"

}

}

**Failure Responses:**

* **400 Bad Request (Missing required fields)**

{

"error": "Budget ID, description, category, amount, and date are required"

}

* **404 Not Found (Budget ID does not exist)**

{

"error": "Budget not found"

}

**4. Update an Existing Transaction**

**Endpoint:** PUT /api/transactions/{id}

**Request Type:**

* **Method:** PUT
* **Headers:**
  + Authorization: Bearer <token>
  + Content-Type: application/json

**Request Example:**

{

"description": "Updated Grocery Shopping",

"amount": 80.00

}

**Successful Response:**

* **Status Code:** 200 OK
* **Response Body:**

{

"message": "Transaction updated successfully",

"transaction": {

"id": 1,

"userId": 1,

"budgetId": 2,

"description": "Updated Grocery Shopping",

"category": "Food",

"amount": 80.00,

"date": "2025-03-14T10:30:00Z"

}

}

**Failure Responses:**

* **404 Not Found (Transaction does not exist)**

{

"error": "Transaction not found"

}

* **403 Forbidden (Trying to update another user’s transaction)**

{

"error": "Unauthorized access to update transaction"

}

* **400 Bad Request (Invalid amount value)**

{

"error": "Amount must be a positive number"

}

**5. Delete a Transaction**

**Endpoint:** DELETE /api/transactions/{id}

**Request Type:**

* **Method:** DELETE
* **Headers:**
  + Authorization: Bearer <token>

**Request Example:**

No request body required.

**Successful Response:**

* **Status Code:** 200 OK
* **Response Body:**

{

"message": "Transaction deleted successfully"

}

**Failure Responses:**

* **404 Not Found (Transaction does not exist)**

{

"error": "Transaction not found"

}

* **403 Forbidden (Trying to delete another user’s transaction)**

{

"error": "Unauthorized access to delete transaction"

}