Personal Banking API Extensions

Minimal Changes Walkthrough

Unchanged Architecture

ROUTERS → SERVICES → REPOSITORIES → DATABASE

🔄 Transfer Money API

Move funds between accounts safely with atomic transactions

```
POST /accounts/transfer
```

- Atomic balance updates
- Validation checks
- Returns updated accounts

✅ KYC Compliance API

Check account KYC compliance status

```
GET /accounts/{id}/kyc-status
```

- New boolean column
- Minimal schema change
- Simple compliance check

Personal Banking API Extensions

Minimal Changes Walkthrough

Zero Changes
to existing endpoints

Preserves
layer separation

Minimal
additive changes

Atomic
transactions

API Usage Examples

```
Transfer: curl -X POST /accounts/transfer -d
'{"from_account_id":1,"to_account_id":2,"amount":50.0}'
```

```
KYC Status: curl GET /accounts/1/kyc-status
```

KYC API Enhancement

Adding KYC Status Management

Clean, Backward-Compatible API Implementation

New Endpoint: PUT `/accounts/{account_id}/kyc-status`

KYC API Enhancement

Adding KYC Status Management

Clean, Backward-Compatible API Implementation

New Endpoint: PUT `/accounts/{account_id}/kyc-status`

Enhancement Overview

Objective

Add a small, clean, and backward-compatible API to set an account's KYC compliance flag

Key Features

- **RESTful Design:** PUT endpoint following REST conventions
- **Flexible Input:** Supports JSON body, query parameters, or defaults
- **Ergonomic:** Optional body defaults to marking accounts compliant
- **Idempotent:** Safe to retry operations
- **Minimal Impact:** Surgical changes to existing codebase

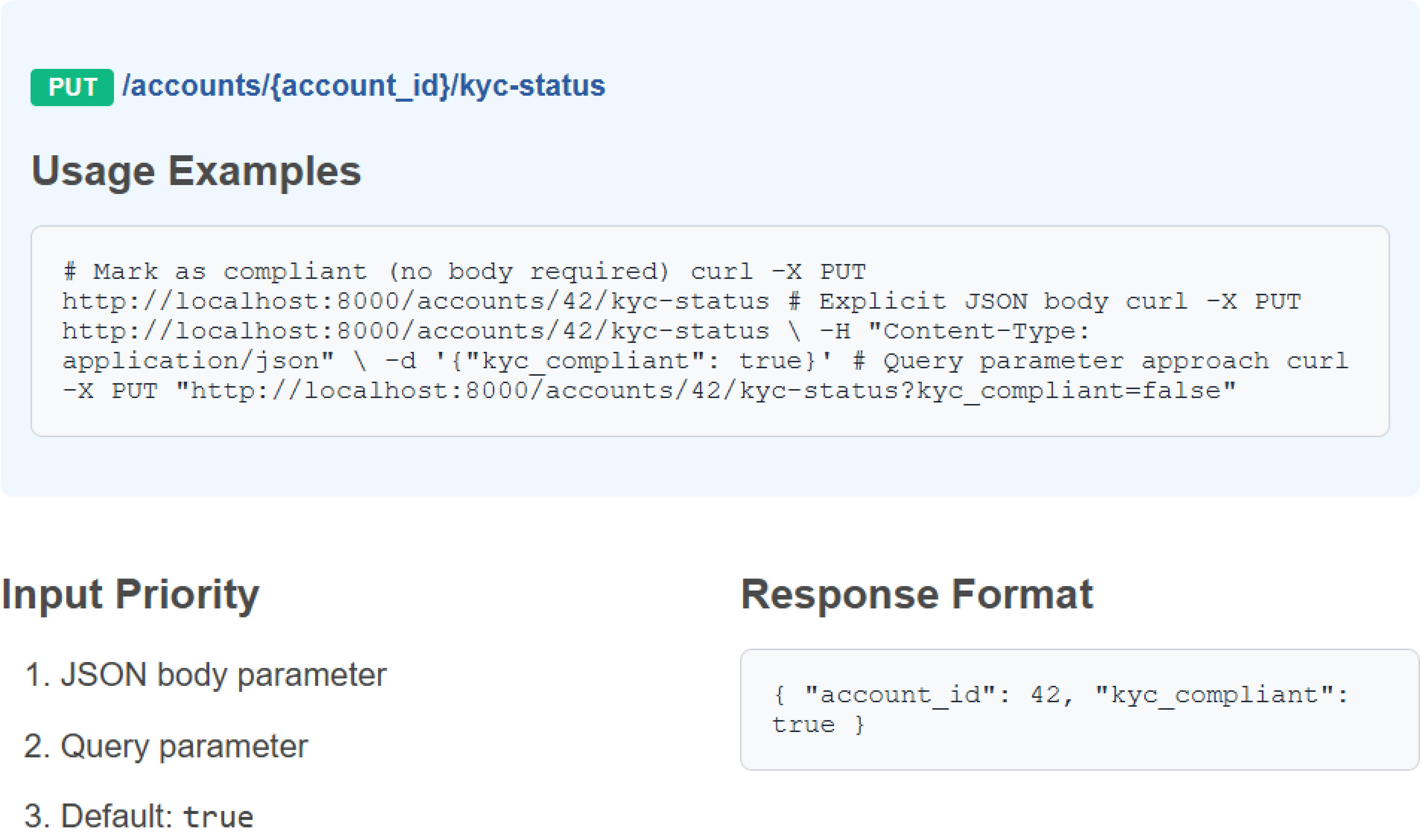
🔧 Minimal Changes

Only 4 files modified with focused additions

🔄 Backward Compatible

No breaking changes to existing functionality

API Design & Usage



PUT /accounts/{account_id}/kyc-status

Usage Examples

```
# Mark as compliant (no body required) curl -X PUT
http://localhost:8000/accounts/42/kyc-status # Explicit JSON body curl -X PUT
http://localhost:8000/accounts/42/kyc-status \ -H "Content-Type:
application/json" \ -d '{"kyc_compliant": true}' # Query parameter approach curl
-X PUT "http://localhost:8000/accounts/42/kyc-status?kyc_compliant=false"
```

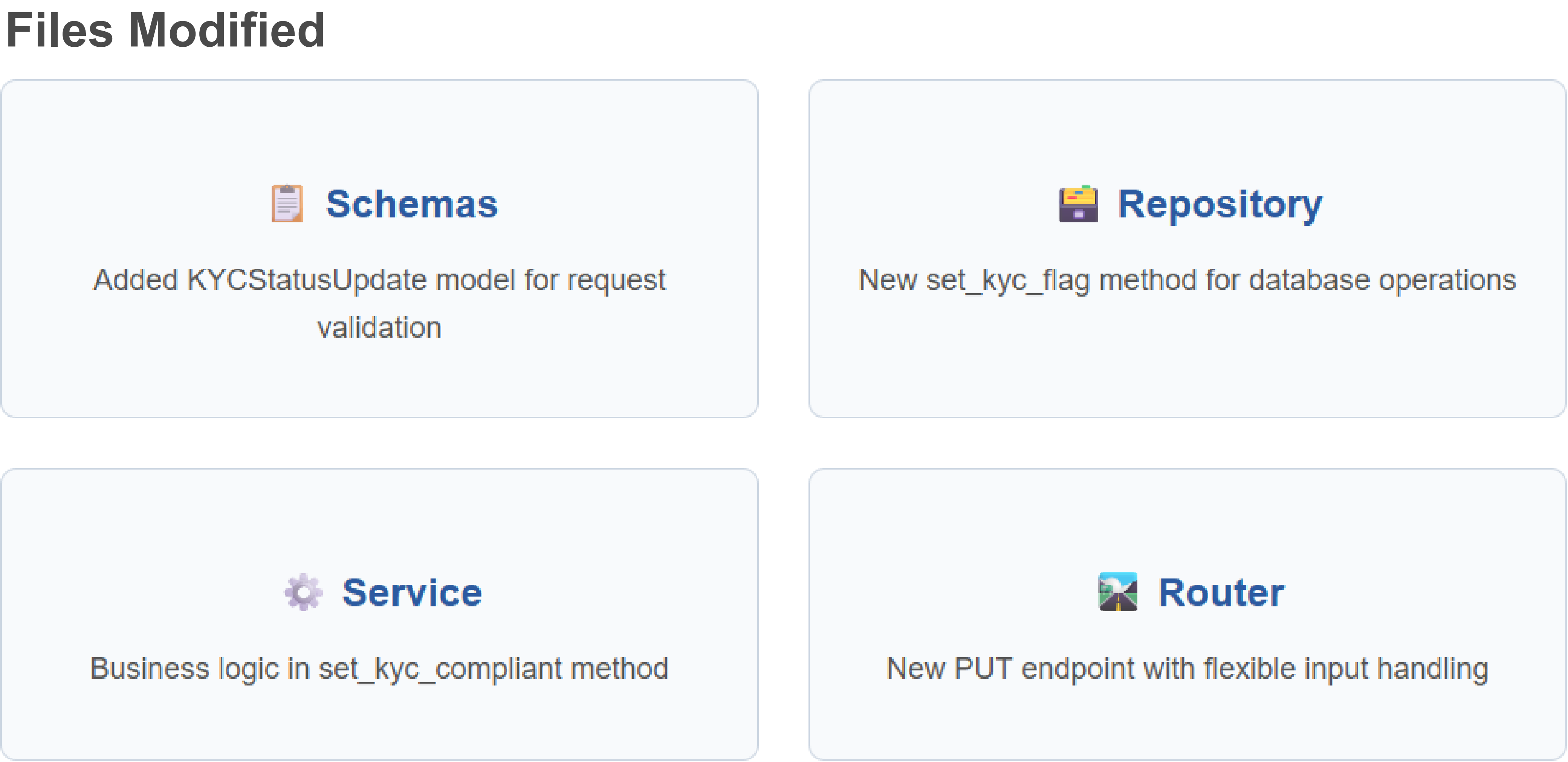
Input Priority

1. JSON body parameter
2. Query parameter
3. Default: `true`

Response Format

```
{ "account_id": 42, "kyc_compliant":
true }
```

Implementation Architecture



Files Modified

📋 Schemas

Added KYCStatusUpdate model for request validation

🗃️ Repository

New set_kyc_flag method for database operations

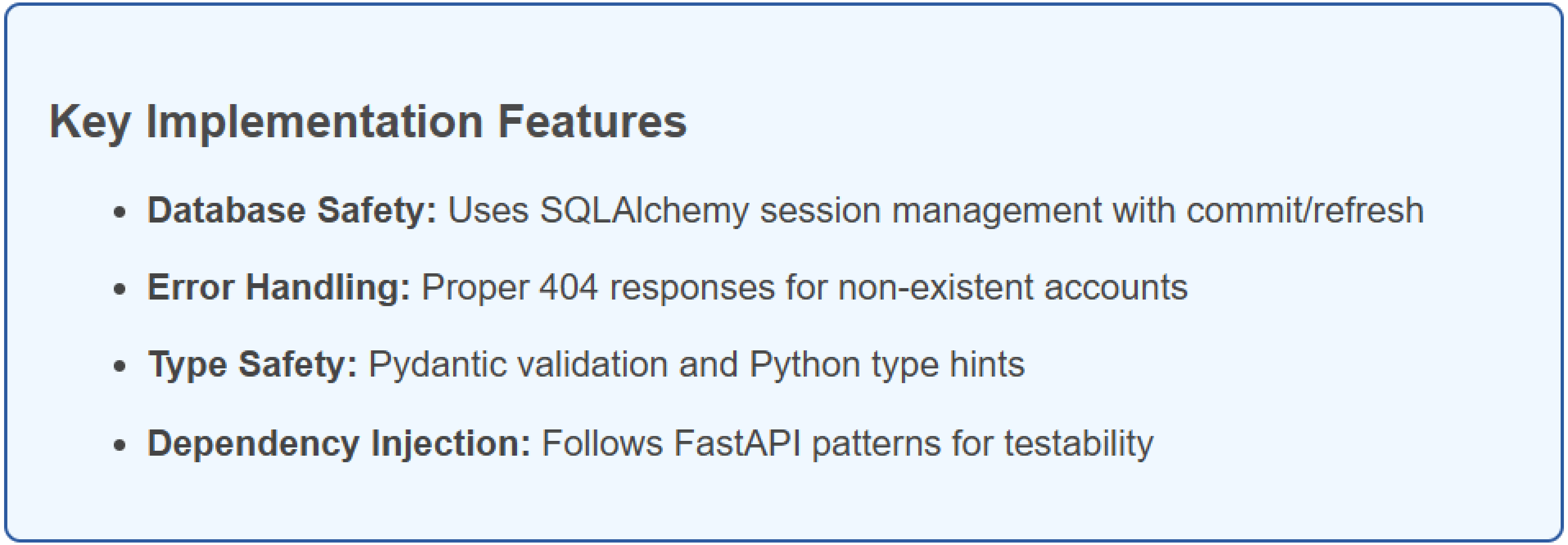
⚙️ Service

Business logic in set_kyc_compliant method

🛣️ Router

New PUT endpoint with flexible input handling

Implementation Architecture



Key Implementation Features

- **Database Safety:** Uses SQLAlchemy session management with commit/refresh
- **Error Handling:** Proper 404 responses for non-existent accounts
- **Type Safety:** Pydantic validation and Python type hints
- **Dependency Injection:** Follows FastAPI patterns for testability