

# Universal API Flow Architecture

## Project Structure

```
/project-root
├── /frontend
│   ├── /src
│   │   ├── /components    # Reusable UI components
│   │   ├── /pages         # Route components
│   │   ├── /api           # API integration layer 🔄 Backend Communication
│   │   │   ├── client.ts  # API client configuration
│   │   │   └── types.ts   # API interfaces
│   │   ├── /hooks        # Custom hooks for data fetching
│   │   └── /public
│   └── /public
└── /backend
    ├── /src
    │   ├── /routes        # API route definitions 📥 Request Entry
    │   │   └── index.ts   # Route registration
    │   ├── /controllers   # Request handlers 🧠 Logic Layer
    │   │   └── users.ts   # User-related operations
    │   ├── /services      # Business logic layer 🧑‍💻 Service Layer
    │   │   └── auth.ts    # Authentication service
    │   ├── /models        # Data models/schemas 📦 Data Layer
    │   │   └── User.ts    # User model definition
    │   ├── /middleware    # Custom middleware 🔄 Cross-cutting
    │   └── /config        # Server configuration
```

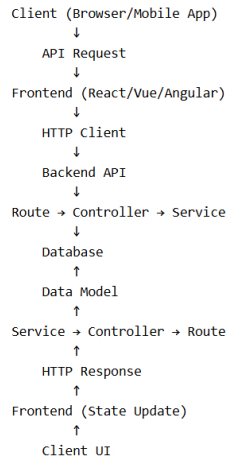
## Request Flow (↓)

```
Client Request
↓
Frontend (/frontend/src/api/client.ts)
↓
HTTP Request (GET /api/users)
↓
Backend Entry (/backend/src/index.ts)
↓
Route Handler (/backend/src/routes/users.ts)
↓
Middleware (/backend/src/middleware/*)
├── Authentication
├── Validation
└── Error Handling
    ↓
    Controller (/backend/src/controllers/users.ts)
    ↓
    Service Layer (/backend/src/services/*)
    ↓
    Data Model (/backend/src/models/*)
    ↓
    Database Query
    ↓
    Database
```

## Response Flow (↑)

```
Database
↑
Query Result
↑
Data Model
↑
Service Layer (Transform/Format)
↑
Controller (Status/Headers)
↑
Response Middleware
├── Error Handling
├── Logging
└── Compression
    ↑
    HTTP Response
    ↑
    Frontend Client
    ↑
    UI Update
```

## 🌐 Complete Request-Response Cycle



## 📦 Data Processing Flow

