

PRODUCTION READY SUBSCRIPTION MANAGEMENT SYSTEM [Project1]

Complete Backend Course | Build and Deploy Your First Production-Ready API

Skip the basic CRUD—this Backend Crash Course is all about building a production-ready Subscription Management System with real users, real money, and real business logic. You'll learn JWT authentication, database modeling, API architecture, security,

<https://youtu.be/rOpEN1JDaD0>



mvc⇒model view controller

models⇒define schema

- User.Model.js
- subscription.model.js

Subscription.model.js

Field	Type	Description / Validation
name	String	Required, trimmed, min 2, max 100 characters
price	Number	Required, must be ≥ 0
currency	String	Enum: 'INR' or 'USD', default 'INR'
frequency	String	Enum: 'daily', 'weekly', 'monthly', 'yearly'
category	String	Enum: sports/news/entertainment/lifestyle/technology/finance/politics/other, required
paymentMethod	String	Required, trimmed
status	String	Enum: 'active', 'cancelled', 'expired', default 'active'
startDate	Date	Required, must be \leq current date
renewalDate	Date	Optional, must be $>$ startDate, auto-calculated if missing
user	ObjectId	References User model, required, indexed
timestamps	N/A	Automatically adds createdAt and updatedAt

Subscription Lifecycle Diagram

Client / API Request

|
| POST /api/v1/subscriptions
V

Subscription Controller (create)

|
| Extract data: name, price, currency, frequency, category, paymentMethod, startDate
V

Subscription Model (Mongoose)

|
| Validate fields:
| - name required, min/max length
| - price ≥ 0
| - category, frequency enums
| - startDate \leq today
| - renewalDate $>$ startDate
V

Pre-validate hook (subscriptionSchema.pre('validate'))

|
|-- Step 1: Auto-calculate renewalDate if missing
| daily \rightarrow +1 day
| weekly \rightarrow +7 days
| monthly \rightarrow +30 days
| yearly \rightarrow +365 days
|
|-- Step 2: Check if renewalDate $<$ today
| \rightarrow set status = 'expired'
V

Save Subscription in MongoDB

|
| Links subscription to User via user ObjectId
| (user field: ref='User')
V

Response to Client

```
{  
  subscription: {name, price, currency, frequency, renewalDate, status, user, ...},  
  success: true  
}
```

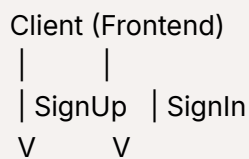
routes

- user.routes
- subscription.routes
- auth

controller.js

- auth
- subscription

Authentication Flow (SignUp & SignIn)



HTTP POST /api/v1/auth/signup or /signin



**Authentication Controller
(signUp / signIn)**



SignUp

1. Extract name, email, pwd
2. Check if user exists
3. Hash password
4. Create user in DB
5. Generate JWT token
6. Commit transaction
7. Send response (token+user)

|

SignIn

1. Extract email, pwd
2. Find user in DB
3. Validate password
4. Generate JWT token
5. Send response (token+user)

|

v

Client

- Stores JWT (localStorage / cookie)
- Uses JWT for protected routes

Middlewares

- auth
- arject
- upstash
- user

Arject Middleware

Client / Incoming Request

|

v

Express Middleware

|

| Arcjet Middleware (aj)

|

|-- Step 1: Shield

```

| - Checks for common attacks (SQLi, XSS, etc.)
| - Mode: LIVE → blocks attacks
|
|-- Step 2: Bot Detection
| - Detects automated bots
| - Allows search engine bots
| - Mode: LIVE → blocks other bots
|
|-- Step 3: Token Bucket Rate Limiting
| - Each request consumes 1 token
| - Max capacity: 10 tokens
| - Refill 5 tokens every 10 seconds
| - Blocks request if tokens exhausted
|
V

```

Protected Route / Controller

```

|
V

```

Response to Client

Subscription Controller Flow Diagram

```

Client / Frontend
|
| HTTP Request
| (POST, GET, PUT, DELETE)
V
Authorize Middleware
(req.user from JWT)
|
V
-----
| Subscription Controller |
-----
|
| 1. createSubscription |
| - req.body + req.user._id |
| - Subscription.create() |
| - Trigger workflowClient |

```

```

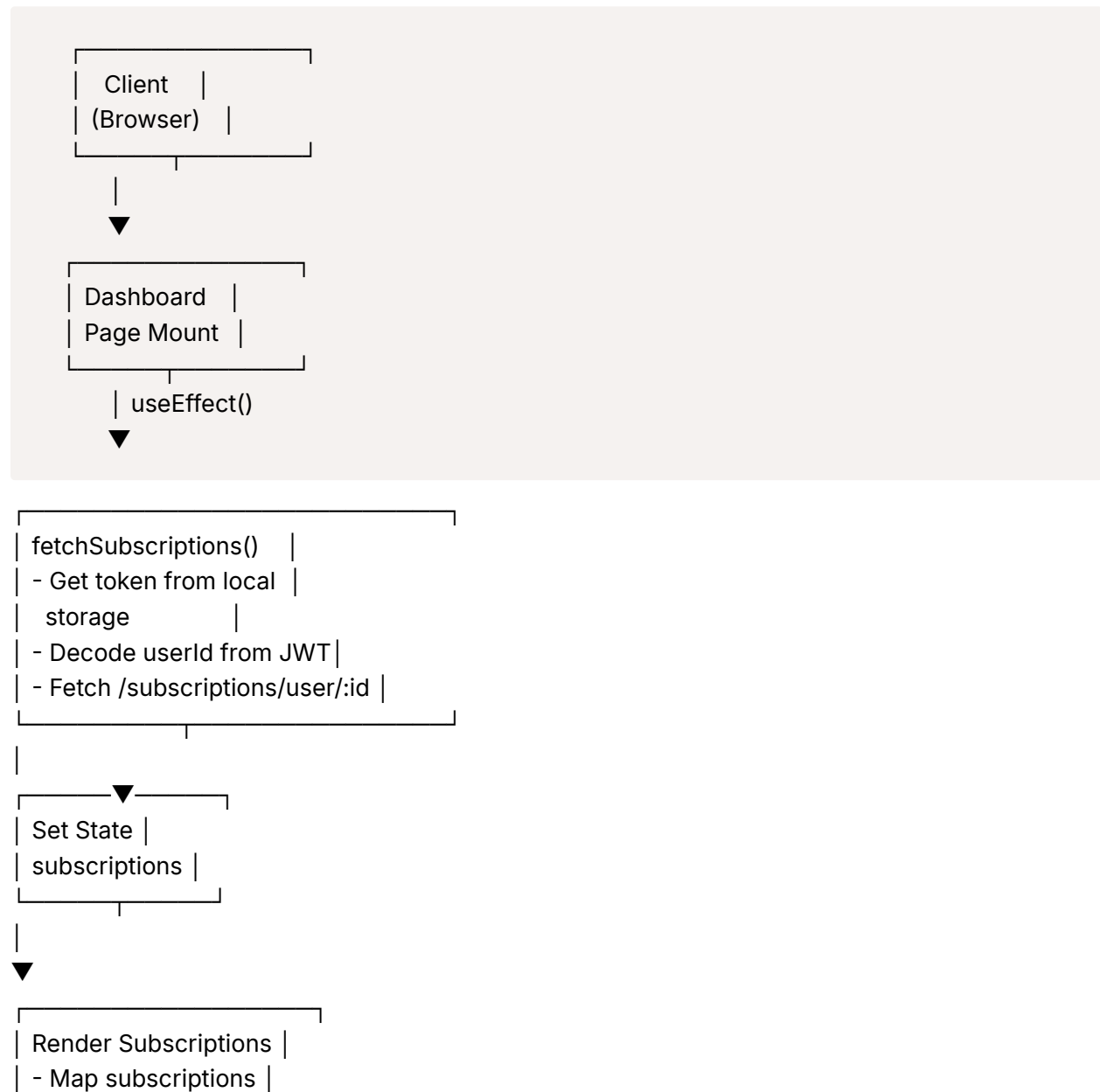
| - Respond with subscription & workflowRunId |
|
|
| 2. getAllSubscriptions |
| - Subscription.find() |
| - Populate user info |
| - Respond with list |
|
|
| 3. getSubscriptionsById |
| - Subscription.findById() |
| - Populate user info |
| - 404 if not found |
|
|
| 4. updateSubscription |
| - Subscription.findByIdAndUpdate() |
| - Return updated doc |
| - 404 if not found |
|
|
| 5. deleteSubscription |
| - Subscription.findByIdAndDelete() |
| - 404 if not found |
|
|
| 6. getUserSubscriptions |
| - Check ownership req.user.id === req.params.id |
| - Subscription.find({user: id}) |
| - Respond with list |
|
|
| 7. cancelSubscription |
| - Find subscription |
| - Check ownership |
| - status = 'canceled', canceledAt = new Date() |
| - Save & respond |
|
|
| 8. upcomingRenewals |
| - Filter renewalDate between today & next 7 days |
| - Only active subscriptions |
| - Respond with list |
|
|-----
|
|
| V
| MongoDB (Subscriptions Collection)
|

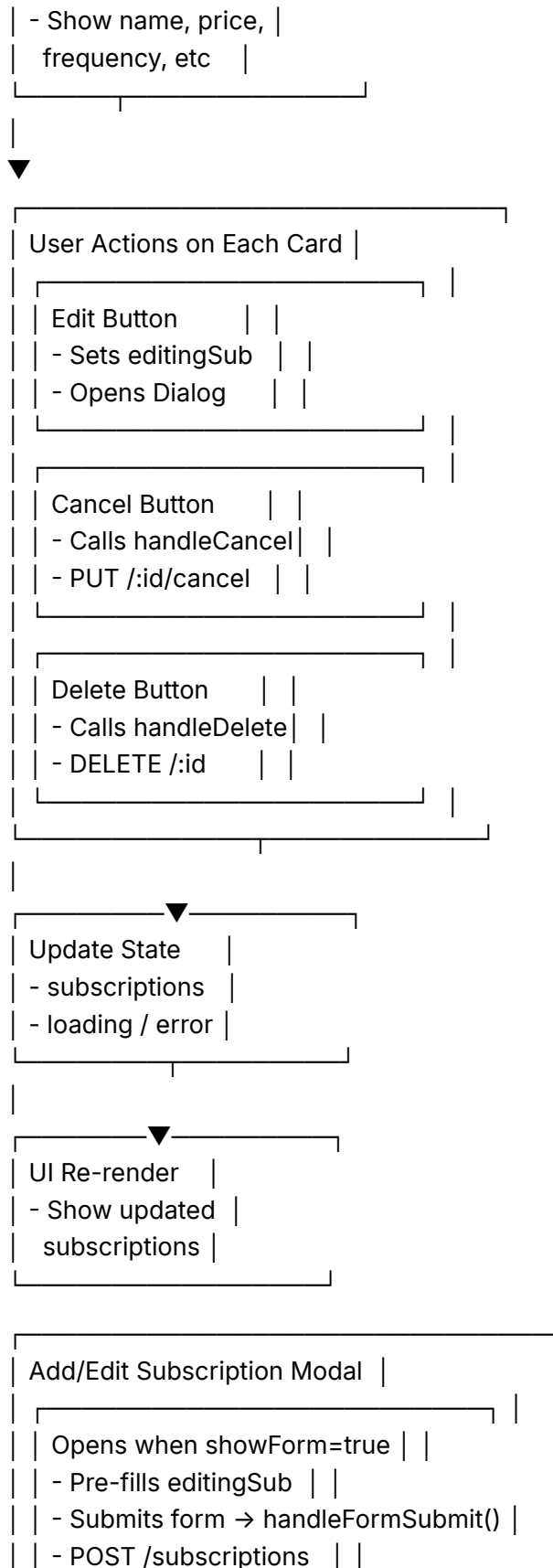
```

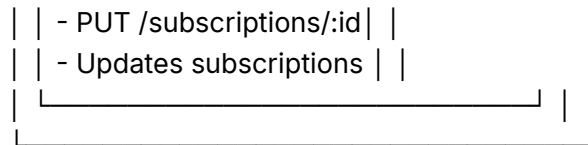
V
Response to Client

FRONTEND

DashboardPage Flow Chart







FRNTEND+BACKEND FLOWCHART

