# Product Requirements Document (PRD)

## Project

NexGPetrolube - B2B Petroleum Platform

## Module

Module 1: Registration, Login, Role Selection & Subscription at Sign-Up

## Introduction

Purpose:  
The purpose of this document is to define the functional, non-functional, and technical requirements for Module 1 of the NexGPetrolube platform.  
  
Scope:  
This module enables user registration, login, role selection, subscription plan selection, and KYC verification.

## User Personas

Buyers: Companies/individuals purchasing petroleum products.  
Sellers: Companies/distributors supplying petroleum products.  
Both Buyer & Seller: Hybrid users.  
Admin: Internal NexGPetrolube staff managing accounts, subscriptions, and KYC.

## User Journey

1. Registration page  
2. Account type selection  
3. Primary role selection  
4. Terms agreement  
5. OTP verification (email & mobile)  
6. Subscription plan selection (for sellers)  
7. KYC verification  
8. Login access

## Functional Requirements

Registration: Fields include name/company, email, mobile, password, account type, role.  
Login: Email/Mobile + password, forgot password, JWT authentication.  
Role Selection: Buyer, Seller, Both.  
Subscription: Monthly, Yearly, or Per Transaction plans with payment integration.  
Verification: Email & SMS OTP, expiry in 5 mins, 3 resend attempts.  
KYC: GST, PAN, Aadhaar, addresses, document uploads.

## Non-Functional Requirements

Security: bcrypt passwords, OTP expiry in Redis, RBAC.  
Performance: Registration <5s, OTP delivery <10s.  
Scalability: 10,000+ concurrent users.  
Availability: 99.9% uptime.

## Acceptance Criteria

Valid registration + OTP verification.  
Duplicate emails/mobiles rejected.  
Sellers must select subscription before listing.  
KYC must be verified before trading.  
Login works with RBAC.  
Brute-force prevention enforced.

## Future Enhancements

SSO for corporates.  
Multi-language support.  
AI-driven fraud detection.  
Biometric login for mobile.

## Dependencies

SMS/Email OTP APIs.  
Payment gateway APIs.  
Cloud infra (AWS/GCP).  
Optional document verification APIs.

## Risks & Mitigation

OTP failures → fallback providers.  
Document fraud → AI + manual checks.  
KYC drop-off → allow later completion.  
Gateway downtime → multiple gateways.

## Conclusion

This module ensures secure onboarding, role assignment, and compliance-driven verification. It establishes the foundation for a scalable B2B petroleum trading ecosystem.

## 12. API Structure (Sample)

This section provides a practical, well-structured sample API design for Module 1 (Registration, Login, Role Selection & Subscription at Sign-Up). It is intended for backend developers and integration teams as a starting point; endpoints use RESTful conventions and JSON payloads. All endpoints are versioned under /api/v1 and enforce HTTPS. Authentication uses JWT access tokens with refresh tokens.

### 12.1 Authentication & Common Headers

Authentication:  
- Login returns an access token (JWT) and refresh token. Access tokens expire in 15 minutes; refresh tokens expire in 30 days.  
  
Common Request Headers:  
- Authorization: Bearer <access\_token>  
- Content-Type: application/json  
- Accept: application/json  
  
Security Considerations:  
- Enforce HTTPS and HSTS  
- Rate-limit sensitive endpoints (e.g., /auth/otp, /auth/login) with progressive backoff  
- Store refresh tokens securely (httpOnly cookies suggested for web clients)

### 12.2 Core Endpoints

* POST /api/v1/auth/register  
  Register a new user account. Accepts basic profile, account type, role, and returns a pending verification status.
* POST /api/v1/auth/verify-otp  
  Verify OTP for email or mobile. Accepts identifier (email/mobile), otp, and otp\_type ('email' or 'mobile').
* POST /api/v1/auth/resend-otp  
  Resend OTP with rate limiting.
* POST /api/v1/auth/login  
  Authenticate using email/mobile + password. Returns access & refresh tokens.
* POST /api/v1/auth/refresh  
  Exchange refresh token for a new access token.
* POST /api/v1/auth/forgot-password  
  Request password reset — sends a reset email with a token.
* POST /api/v1/auth/reset-password  
  Reset password using reset token.
* GET /api/v1/users/{userId}  
  Get user profile and role details (RBAC enforced).
* PUT /api/v1/users/{userId}  
  Update user profile (limited fields).
* GET /api/v1/roles  
  Fetch available roles (Buyer, Seller, Both).
* GET /api/v1/plans  
  Fetch subscription plans and metadata (monthly, yearly, per-transaction).
* POST /api/v1/subscriptions/checkout  
  Create a subscription checkout session (integrate with payment gateway).
* GET /api/v1/subscriptions/{userId}  
  Get current subscription status for a user.
* POST /api/v1/kyc/upload  
  Upload KYC documents (multipart/form-data). Returns file IDs.
* POST /api/v1/kyc/submit  
  Submit KYC data along with uploaded file IDs for admin review.
* GET /api/v1/kyc/{userId}/status  
  Get current KYC verification status and notes.

### 12.3 Request/Response Examples

#### Register (Example)

Request (JSON):

{  
 "full\_name": "Joydeep Dutta",  
 "company\_name": "AquaDev Pvt Ltd",  
 "email": "aquadevjd@gmail.com",  
 "mobile": "+919163331565",  
 "password": "Str0ngP@ssw0rd!",  
 "account\_type": "Company",  
 "primary\_role": "Buyer",  
 "accept\_terms": true,  
 "marketing\_opt\_in": false  
}

Response (201 Created):

{  
 "user\_id": "uuid-1234-5678",  
 "status": "pending\_verification",  
 "message": "OTP sent to email and mobile"  
}

#### Verify OTP (Example)

Request (JSON):

{  
 "identifier": "aquadevjd@gmail.com",  
 "otp": "123456",  
 "otp\_type": "email"  
}

Response (200 OK):

{  
 "user\_id": "uuid-1234-5678",  
 "status": "verified",  
 "next": "/api/v1/plans (if seller role selected)"  
}

#### Login (Example)

Request (JSON):

{  
 "username": "aquadevjd@gmail.com",  
 "password": "Str0ngP@ssw0rd!"  
}

Response (200 OK):

{  
 "access\_token": "<jwt\_access\_token>",  
 "refresh\_token": "<refresh\_token>",  
 "expires\_in": 900,  
 "user": {  
 "user\_id": "uuid-1234-5678",  
 "roles": ["Buyer"],  
 "subscription\_status": "none"  
 }  
}

#### KYC Upload (Example)

Request (multipart/form-data):

POST /api/v1/kyc/upload  
Headers: Authorization: Bearer <access\_token>  
Form fields:  
- file (binary) - PAN.pdf  
- document\_type = "pan"

Response (201 Created):

{  
 "file\_id": "file-uuid-1111",  
 "document\_type": "pan",  
 "uploaded\_at": "2025-09-15T10:35:20Z"  
}

#### Subscription Checkout (Example)

Request (JSON):

{  
 "user\_id": "uuid-1234-5678",  
 "plan\_id": "plan-yearly-001",  
 "payment\_method": "razorpay"  
}

Response (201 Created):

{  
 "checkout\_id": "chk\_7890",  
 "payment\_url": "https://payments.gateway/checkout/chk\_7890",  
 "expires\_at": "2025-09-15T11:05:00Z"  
}

### 12.4 Error Handling & Status Codes

Common Response Structure for Errors:  
{  
 "error": {  
 "code": "USER\_ALREADY\_EXISTS",  
 "message": "Email already registered",  
 "details": null  
 }  
}

Common HTTP Status Codes:  
- 200 OK  
- 201 Created  
- 400 Bad Request  
- 401 Unauthorized  
- 403 Forbidden  
- 404 Not Found  
- 429 Too Many Requests  
- 500 Internal Server Error

### 12.5 OpenAPI (YAML) Snippet

openapi: 3.0.1  
info:  
 title: NexGPetrolube Module1 API  
 version: '1.0.0'  
paths:  
 /api/v1/auth/register:  
 post:  
 summary: Register a new user  
 requestBody:  
 required: true  
 content:  
 application/json:  
 schema:  
 type: object  
 properties:  
 full\_name: { type: string }  
 email: { type: string }  
 mobile: { type: string }  
 password: { type: string }  
 account\_type: { type: string }  
 primary\_role: { type: string }  
 responses:  
 '201':  
 description: Created  
 /api/v1/auth/login:  
 post:  
 summary: Login with email/mobile and password  
 requestBody:  
 required: true  
 content:  
 application/json:  
 schema:  
 type: object  
 properties:  
 username: { type: string }  
 password: { type: string }  
 responses:  
 '200':  
 description: OK

Document generated/updated: 2025-09-15T10:52:07.363541Z