Mobile Recharge Web Application

Requirement Analysis Document (MERN Stack)

Project: Mobile Recharge Web Application
Technology Stack: MERN (MongoDB, Express.js, React.js, Node.js)

---

1. Executive Summary

The Mobile Recharge Web Application is a digital platform enabling users to recharge their mobile phones with prepaid and postpaid plans from various telecom operators[1]. Built using the MERN stack, it ensures scalability, security, and user-friendliness[2]. This document outlines functional requirements, non-functional requirements, features, technology stack, UI/UX wireframes, and database schema necessary for design and development phases.

---

2. Functional Requirements

2.1 User Authentication

- User registration with email/phone verification
- Secure login using email/phone and password
- Password reset functionality via email/OTP
- Account profile management and updates

2.2 Mobile Number Validation

- Validate and accept 10-digit mobile phone numbers
- Automatically detect telecom operator (Jio, Airtel, Vodafone-Idea, BSNL, MTNL)[1]
- Display operator logo and network type (4G/5G)
- Support multiple circles/regions for accurate identification

2.3 Plan Browsing and Selection

- Display prepaid and postpaid recharge plans
- Filter plans by validity, data limits, and price
- Search functionality for plans
- Show plan details: validity, data, talk time, benefits
- Bookmark/favorite plans for quick access

2.4 Payment Processing

- Support multiple payment methods (Credit/Debit Card, Net Banking, Digital Wallets)[1]
- Secure payment gateway integration with Razorpay/Stripe
- Real-time payment status updates
- Payment receipt generation
- Transaction history with detailed information
- PCI DSS compliance for payment security[1]

2.5 Recharge Execution

- Process recharge requests to telecom operators
- Real-time confirmation of recharge status
- Error handling and automatic retry mechanism
- Confirmation SMS/Email to users

2.6 User Dashboard

- View recharge history with date, operator, amount, plan details
- Display current balance and validity
- Quick recharge options for favorite plans
- Download transaction receipts (PDF format)
- Filter and search recharge history

2.7 Admin Dashboard

- View all users and their recharge history
- Monitor transaction statistics and revenue
- Manage telecom operators and their plans

- Manage promotions and discount codes
- View system performance metrics
- Generate reports and analytics

2.8 Notifications

- Email notifications for recharge confirmation
- SMS alerts for successful/failed transactions
- Push notifications for promotional offers
- In-app notifications for account activities

---

3. Non-Functional Requirements

3.1 Performance

- Page load time: less than 3 seconds
- API response time: less than 500ms for typical operations[2]
- Support 1000+ concurrent users
- Recharge processing completion: 2-3 minutes
- Database query optimization using Redis caching

3.2 Security

- HTTPS/SSL encryption for all data transmission
- Password hashing using bcrypt algorithm
- JWT tokens for session management (15-minute expiry)
- Rate limiting to prevent brute force attacks
- Input validation and sanitization to prevent injection attacks
- SQL injection and XSS attack prevention
- Two-factor authentication for high-value transactions
- PCI DSS compliance for payment processing[1]
- Secure API endpoints with role-based access control

3.3 Usability

- Responsive design for desktop, tablet, and mobile devices
- Intuitive user interface with clear navigation

- Mobile-first design approach[2]
- WCAG 2.1 Level AA accessibility compliance
- Clear error messages and validation feedback
- Multi-language support (English, Hindi)

3.4 Reliability

- 99.5% uptime SLA
- Automatic failover mechanisms
- Database backup and recovery procedures
- Error logging and monitoring
- Graceful error handling with fallback options

3.5 Maintainability and Scalability

- Clean, well-documented code following MERN best practices[2]
- Modular component structure in React
- API documentation using Swagger/OpenAPI
- Version control using Git
- Automated testing (unit, integration, E2E)
- Horizontal scaling capability and load balancing
- Cloud deployment support (AWS, Azure, GCP)

3.6 Compliance

- GDPR compliance for user data privacy
- Local data protection regulations
- GST compliance for payments
- Terms of Service and Privacy Policy pages
- User data deletion and portability options

---

4. Feature List

Core Features (High Priority)

Feature	Description
User Authentication	Email/Phone-based signup/login with password recovery

Mobile Number Validation	Automatic operator detection for 10-digit numbers
Plan Browsing	Display prepaid/postpaid plans with filters
Payment Gateway Integration	Support multiple payment methods (cards, wallets, net banking)
Recharge Processing	Execute recharge and provide confirmation
Transaction History	View past recharges with detailed information
User Dashboard	Quick overview of account and recharge options
Admin Dashboard	Manage users, plans, and view analytics

Secondary Features (Medium Priority)

Feature	Description
Plan Favorites	Save favorite plans for quick access
Promotional Codes	Apply discount coupons to transactions
Referral Program	Earn rewards by inviting friends
Recharge Reminders	Notify users when validity expires
Multi-language Support	Support Hindi and English
Email Notifications	Receive transaction confirmations

Future Enhancements

- DTH Recharge support
- Bill payment services
- Cashback loyalty program
- Native mobile app (iOS/Android)
- Advanced analytics dashboard

---

5. Technology Stack

Frontend (React.js)

- React Hooks for state management
- React Router for navigation

- Axios for API communication
- Tailwind CSS/Material-UI for styling
- React Query for server state management
- Form validation (Formik + Yup)
- Charts library (Recharts) for analytics

Backend (Node.js + Express.js)

- Express.js for REST API development[2]
- JWT for authentication
- Helmet.js for security headers
- CORS middleware for cross-origin requests
- Joi for input validation

Database (MongoDB)

- MongoDB Atlas for cloud database
- Mongoose ODM for schema definition
- Database indexing for performance
- TTL indexes for session management

Additional Tools

- Redis for caching and session storage
- Stripe/Razorpay for payment gateway integration
- Nodemailer for email notifications
- AWS S3 for file storage
- Twilio for SMS notifications
- Jest for unit testing

---

6. UI/UX Wireframes Overview

Key Screens Description

Landing Page / Home

- Logo, navigation (Home, Recharge, Login)
- Mobile number input field with "Recharge Now" button

- Quick features highlight section
- Footer with links

Login / Sign Up

- Centered card with email/phone, password fields
- Create Account button
- Social login options (Google, Facebook)
- Forgot Password link

Mobile Number & Operator Selection

- Mobile number input (10 digits)
- Auto-detected operator display
- Circle/Region selector dropdown
- Prepaid/Postpaid toggle buttons
- Continue button

Plan Selection Screen

- Mobile number and operator summary
- Search and filter options
- Plan cards showing amount, validity, benefits
- Add to Favorites option
- Continue to Payment button

Payment Page

- Plan details and total amount display
- Payment method selection (Card, UPI, Net Banking)
- Card/payment details input
- Promo code application field
- Secure payment button with SSL badge

Recharge Confirmation

- Success/failure icon and message
- Transaction details (ID, amount, date, plan)
- New validity display
- Download receipt and share options
- New Recharge and Dashboard buttons

User Dashboard

- Quick action buttons (Recharge, Favorites, History)
- Recent recharges table
- Account settings options
- Promotional offers banner
- Profile and logout section

Admin Dashboard

- Navigation menu (Dashboard, Users, Plans, Reports, Offers)
- Overview cards (Total Users, Revenue, Transactions, Cashback)
- Recent transactions table with filters
- Quick actions for plan management
- Analytics section

---

7. Database Schema (MongoDB)

Collections Overview

The application will use three primary collections in MongoDB: users, transactions, and plans.

User Collection

Field	Type	Description
_id	ObjectId	Unique user identifier
email	String	User email (unique)
phone	String	User phone number
password	String	Hashed password
firstName	String	User first name
lastName	String	User last name
createdAt	Date	Account creation timestamp
updatedAt	Date	Last update timestamp
lastLogin	Date	Last login timestamp
isVerified	Boolean	Email/phone verification status
status	String	Active/inactive status

Transaction Collection

Field	Type	Description
_id	ObjectId	Unique transaction identifier
userId	ObjectId	Reference to user
mobileNumber	String	Mobile number recharged
operator	String	Telecom operator
planId	ObjectId	Reference to plan
amount	Number	Recharge amount
paymentMethod	String	Payment method used
transactionId	String	Payment gateway transaction ID
status	String	Success/pending/failed
createdAt	Date	Transaction creation timestamp
completedAt	Date	Completion timestamp
discountCode	String	Applied discount code

Plan Collection

Field	Type	Description
_id	ObjectId	Unique plan identifier
operator	String	Telecom operator name
planType	String	Prepaid/postpaid
amount	Number	Plan cost
validity	String	Plan validity period
dataLimit	String	Data allowance
talkTime	String	Talktime allowance
sms	String	SMS allowance
benefits	Array	List of benefits
description	String	Plan description
isActive	Boolean	Active status
createdAt	Date	Creation timestamp