DETAILED REQUIREMENTS DOCUMENT (DRD)

Scheme Assistance Platform

Part 1: System Overview, Users, Authentication, Core Concepts

---

1. System Overview

1.1 What the System Is

This system is a **paid assistance platform** that helps users apply for government and private schemes.

The platform:

- Does **not** apply for schemes automatically
- Does **not** guarantee approvals
- **Does** provide professional assistance, document handling, and application support

Users pay **upfront** to request assistance.
Admins execute the work and upload proof of application.

The platform acts as:

- A **transaction system**
- A **document management system**
- A **single source of truth**

1.2 What the System Is NOT

The system is NOT:

- A government portal
- A free information website
- A chat-based support system

- A marketplace between users and random agents

This distinction is critical for design, messaging, and legal safety.

---

2. User Types and Roles

2.1 End Users

The system currently supports **three user categories**:

1. Students
2. Farmers
3. Loan Candidates

These are **categories**, not permissions.
All end users have the same system capabilities.

Future categories can be added **without database redesign.**

2.2 Admin Users

Admins are internal platform operators.

Admins:

- Manage schemes
- Handle orders
- Verify documents
- Upload proof
- Complete orders

Admins **do not** handle payments manually.

---

3. Authentication & Access

3.1 Authentication Methods (Passwordless Only)

The system supports **only passwordless authentication**:

1. SMS OTP
2. WhatsApp OTP
3. Google OAuth

There is **no password creation**, **no password storage**, and **no password reset.**

3.2 Authentication Rules

- OTPs are time-bound
- OTP attempts are rate-limited
- Sessions are stored server-side
- Redis is used for OTP/session handling

If authentication fails → no access.

3.3 Authorization Rules

- Users can only access:
    - Their own profile
    - Their own orders
    - Their own documents
- Admins can access:
    - All users
    - All orders
    - All documents
    - All schemes

No cross-user visibility is allowed.

---

4. Core Business Concepts (VERY IMPORTANT)

4.1 Scheme

A **Scheme** represents a government or private program users can apply for.

Each scheme contains:

- Name

- Description
- Category (Student / Farmer / Loan)
- Scheme type (Government / Private)
- Eligibility criteria
- Benefits
- Required documents list
- Status (Active / Inactive)

A scheme **defines the document requirements.**

4.2 Order (Central Entity)

An **Order** represents a paid request for assistance.

An order is created **only after successful payment**.

If payment fails → order does not exist.

Each order links:

- User
- Scheme
- Payment
- Documents
- Proof of work

Orders are immutable in terms of payment status.

4.3 Order Lifecycle

Orders move through defined states only:

1. `PAID`
2. `IN_PROGRESS`
3. `PROOF_UPLOADED`
4. `COMPLETED`
5. `CANCELLED` (admin-only, rare)

There is **no unpaid order state.**

---

5. Payment System Requirements

5.1 Payment Model

- **100% upfront payment**
- Mandatory to create an order
- No partial payments
- No pay-later option

Payments are tied one-to-one with orders.

5.2 Payment Flow

1. User submits details + documents
2. User is redirected to payment
3. Payment succeeds
4. Order is created
5. Admin work begins

If payment fails:

- Documents are discarded
- No order is stored

5.3 Payment Data Stored

For every order:

- Amount paid
- Transaction ID
- Payment timestamp
- Payment status (Success only)

Refund handling is **out of scope for now** but data structures must allow it later.

---

6. Document Management (CORE FEATURE)

6.1 Document Upload by Users

Users **must upload all required documents** defined by the scheme.

Rules:

- Documents are uploaded **before payment**
- Missing documents → cannot proceed
- Upload happens inside the platform only

Allowed formats:

- PDF
- JPG / PNG

File limits:

- Size limits per file
- One or more files per document type

6.2 Document Structure

Each uploaded document has:

- Document type (from scheme)
- File URL
- Upload timestamp
- Verification status
- Rejection reason (if any)

6.3 Document Status Lifecycle

Each document can be:

- `UPLOADED`
- `VERIFIED`
- `REJECTED`
- `RESUBMISSION_REQUIRED`

Admins control verification.

Users can **only re-upload rejected documents**.

6.4 Document Access Control

Documents are visible only to:

- The user who uploaded them
- Authorized admins

Documents are never public.

---

7. Admin Proof Upload

7.1 Proof Definition

Proof represents evidence that:

- The application was submitted
- Assistance work was completed

Proof types:

- Application receipt
- Reference ID
- Screenshot
- Confirmation document

7.2 Proof Rules

- Proof upload is **mandatory**
- Order cannot be marked `COMPLETED` without proof
- Multiple proof files are allowed

Proof is visible to users after upload.

---

8. Communication Model

8.1 What the Platform Handles

- Order status updates
- Document status updates
- Proof visibility
- Notifications

8.2 What the Platform Does NOT Handle

- Chat
- Messaging threads
- Call scheduling
- WhatsApp conversations

Calls and WhatsApp are **off-platform.**

The platform remains the **official record.**

---

9. Notifications (Minimum Required)

Users receive notifications when:

- Payment is successful
- Order status changes
- Document is rejected
- Proof is uploaded
- Order is completed

Notification types:

- In-app (mandatory)
- SMS / WhatsApp (optional later)

---

10. User Profile Management

Users can:

- View profile
- Update basic info
- See order history
- See document status

Profile edits do not affect existing orders.

---

11. Admin Capabilities (Operational)

Admins can:

- Manage schemes
- View users
- View orders
- Change order states
- Verify documents
- Upload proof
- View payment info
- Add internal notes

Admins **cannot**:

- Modify payment records
- Access unrelated user accounts casually

---

12. Technical Foundations (Locked)

- Backend: Service-based architecture
- Redis:
    - OTP/session handling
    - Rate limiting
    - Caching scheme lists
- Client caching:
    - Scheme data
    - Orders
    - User profile

---

13. End-to-End User Workflows

13.1 User Onboarding & Login Flow

1. User lands on the platform
2. User chooses login method:
    - SMS OTP
    - WhatsApp OTP
    - Google OAuth
3. System verifies authentication
4. If first-time user:
    - Profile is created automatically
5. User is redirected to scheme exploration

There is **no manual signup form** and **no password handling.**

13.2 Scheme Discovery Flow

1. User browses schemes
2. User can filter by:
    - Category (Student / Farmer / Loan)
    - Scheme type (Govt / Private)
3. User opens scheme detail page
4. User reviews:
    - Eligibility
    - Benefits
    - Required documents
    - Service fee
5. User proceeds only if eligible (self-declared)

The system does **not auto-reject** based on eligibility**.**

13.3 Order Creation Flow (Critical Path)

This is the **most important workflow.**

1. User clicks **Request Assistance**
2. System shows:
    - Required documents checklist

    - Upload interface
3. User uploads all required documents
4. System validates:
    - File type
    - File size
    - Required count
5. User confirms consent & disclaimers
6. User proceeds to payment
7. Payment succeeds
8. **Order is created**
9. Order state = `PAID`

If **any step fails**, order creation is aborted.

13.4 Order Tracking (User Side)

Users can view:

- Order status
- Scheme details
- Uploaded documents
- Document verification status
- Proof uploaded by admin
- Order timeline (timestamps)

Users cannot:

- Modify paid orders
- Change scheme
- Delete orders

---

14. Admin Workflows

14.1 Admin Order Processing Flow

1. Admin views new orders (`PAID`)
2. Admin reviews uploaded documents
3. Admin verifies documents:
    - Marks `VERIFIED` or `REJECTED`

4. If rejected:
    - Admin adds rejection reason
    - Order remains `IN_PROGRESS`
5. User re-uploads rejected documents
6. Admin proceeds with application work
7. Admin uploads proof
8. Order state changes to `PROOF_UPLOADED`
9. Admin marks order `COMPLETED`

Admin cannot complete an order without proof.

14.2 Admin Scheme Management Flow

Admins can:

- Create new schemes
- Define required documents
- Set service fee
- Activate / deactivate schemes

Deactivated schemes:

- Are hidden from users
- Do not affect existing orders

14.3 Admin User Management Flow

Admins can:

- View user list
- View user profile
- View user orders

Admins cannot:

- Edit user identity data arbitrarily
- Access documents without a valid order context

---

15. Failure Handling & Edge Cases

15.1 Payment Failures

- Failed payment → no order created
- Uploaded documents are discarded
- User must restart flow

No “pending payment” state exists.

15.2 Document Rejection Loop

- Rejected documents must include a reason
- Only rejected documents are re-uploadable
- Order cannot proceed until documents are verified

Infinite loops are prevented by admin discretion.

15.3 Admin Inaction

If admin does not act:

- Order remains `PAID` or `IN_PROGRESS`
- System does not auto-complete or auto-cancel

SLA handling is a **business concern**, not a system default.

---

16. Security Requirements

16.1 Data Security

- All documents stored in private object storage
- Access via short-lived signed URLs
- No public document URLs

16.2 Access Control

- Users access only their data
- Admin access is role-verified
- Every document access is authenticated

16.3 Audit Logging (Mandatory)

System logs:

- Order state changes
- Document verification actions
- Proof uploads
- Admin actions

Logs are immutable and timestamped.

---

17. Compliance & Legal Safety (System-Level)

The platform must display:

- Service disclaimer
- No-guarantee clause
- Consent checkbox before payment

System stores:

- Consent timestamp
- Accepted terms version

This protects the platform in disputes.

---

18. Performance & Scalability Requirements

18.1 Caching Strategy

Redis

- OTPs
- Sessions
- Rate limiting
- Scheme list cache

Client-side caching

- Scheme data
- User profile
- Order lists

Cache invalidation occurs on:

- Scheme updates
- Order state changes

18.2 File Storage

- Object storage (S3/R2 equivalent)
- Folder structure:
  - `/orders/{orderId}/documents/`
  - `/orders/{orderId}/proof/`

No documents stored in database.

18.3 Horizontal Scalability

- Stateless backend services
- Shared Redis
- Shared object storage

System scales by adding instances.

---

19. Non-Functional Requirements

19.1 Availability

- Platform should tolerate partial service failures

- Authentication and payment failures must not corrupt data

19.2 Observability

- Error logging
- Request tracing
- Payment failure monitoring

19.3 Maintainability

- Clear entity boundaries
- No hardcoded business rules
- Extensible user categories

---

20. Explicit Out-of-Scope (Locked)

The system will NOT include:

- In-app chat
- User-to-user communication
- AI eligibility checks
- Automatic scheme approval
- Offline document handling

These are intentionally excluded.

---

21. Final System Philosophy (Locked)

- **Upfront payment only**
- **Documents are mandatory**
- **Proof is mandatory**
- **Platform is the source of truth**
- **Trust is earned through process, not promises**

Final Note (Important)

This document now defines:

- A **real production system**
- With **clear boundaries**
- And **zero ambiguous states**

You can now:

- Design DB schemas
- Write APIs
- Build frontend & backend in parallel
- Hand this to another engineer without explanations