

# Phase 1: Role-Based Foundation

Web Development Department

## Objective

The goal of Phase 1 is to establish a robust project architecture. Students will implement a full-stack application focusing on project setup, secure authentication, and Role-Based Access Control (RBAC) using a persistent database.

## Recommended Tech Stack

### Frontend Infrastructure

- **Framework:** React.js (Vite) or Next.js
- **Styling:** Tailwind CSS or CSS Modules
- **State Management:** Context API
- **HTTP Client:** Axios or Fetch API

### Backend & Database

- **Runtime:** Node.js with Express.js
- **Database:** MongoDB (Mongoose) or PostgreSQL
- **Auth:** JWT (JSON Web Tokens) & Bcrypt
- **Tools:** Postman (API Testing), Git

## Role-Based Workflows

### 1. Sender Role

- **Authentication:** secure login capability.
- **Metadata Entry:** Ability to create records (simulating file uploads) with:
  - File Name
  - Description
  - Category Tag
- **Dashboard:** View a personal history list of files added by this specific user only.

### 2. Receiver Role

- **Authentication:** Secure login capability.
- **Global View:** Access to a read-only table displaying *all* records created by any Sender.
- **Detail View:** Ability to click on a record to view full metadata details.
- **Restriction:** Cannot add, edit, or delete records.

## Backend Architecture Requirements

- **Authentication:** Implementation of JWT for stateless session management.
- **Middleware:** Custom middleware to verify tokens and enforce role permissions (e.g., `verifyToken, checkRole(['sender'])`).
- **API Security:** Protected routes that reject unauthorized access with appropriate HTTP status codes (401/403).
- **Data Structure:** Relational schemas linking users to their specific roles and data entries.

## Deliverables

1. **Source Code:** A clean, structured GitHub repository with meaningful commit messages.
2. **Documentation:** A `README.md` file containing:
  - Project prerequisites (Node version, etc.)
  - Installation & setup commands (`npm install`, `npm start`)
  - Environment variable examples (`.env`)
3. **Demo:** A fully functional authentication flow with separate dashboards for Senders and Receivers.

### Learning Outcome:

By the end of this phase, students will have mastered the fundamentals of **RBAC**, understanding how to secure backend API routes and conditionally render frontend UI elements based on user permissions.