



We understand your world

Capstone Project Report : Digital - KYC

RIAJUL HOQUE CHOUDHURY
ASSAM DOWN TOWN UNIVERSITY
PROGRAM ROLL NO – DT.32

Introduction

- The Digital KYC (Know Your Customer) system is designed to simplify and modernize the customer onboarding process for financial services such as banking.
- Traditional KYC methods are slow, paper-based, and require physical verification, leading to delays and high drop-off rates.
- This project aims to create a **fully digital, secure, step-by-step KYC workflow** that users can complete from any device.
- The system includes multi-step forms, document uploads, identity verification, and automated progress tracking.
- A responsive and user-friendly React frontend ensures smooth navigation across the entire KYC journey.
- A Node.js + Express backend handles authentication, document processing, storage, and business logic.
- The system improves accuracy, reduces manual effort, and accelerates account creation while maintaining high security.
- The solution uses modern UI/UX principles, automated validation, and seamless PDF confirmation generation.
- The final KYC status and user information are displayed in a secure customer dashboard.
- Overall, this Digital KYC solution increases onboarding efficiency and enhances customer experience.

Objectives

- Reduce customer drop-off during onboarding
- Provide a clean, step-by-step KYC journey
- Enable secure document uploads with validation
- Auto-generate confirmation PDF upon completion
- Provide a professional, bank-like dashboard

Project Structure

```
└── digital-kyc
    ├── backend
    │   ├── config
    │   ├── controllers
    │   ├── middleware
    │   ├── models
    │   ├── node_modules
    │   ├── routes
    │   ├── uploads
    │   ├── utils
    │   └── .env
    │   ├── package-lock.json
    │   ├── package.json
    │   └── server.js
    └── frontend
        ├── node_modules
        ├── public
        ├── src
        ├── package-lock.json
        ├── package.json
        └── tailwind.config.js
```

Tech Stack

Frontend

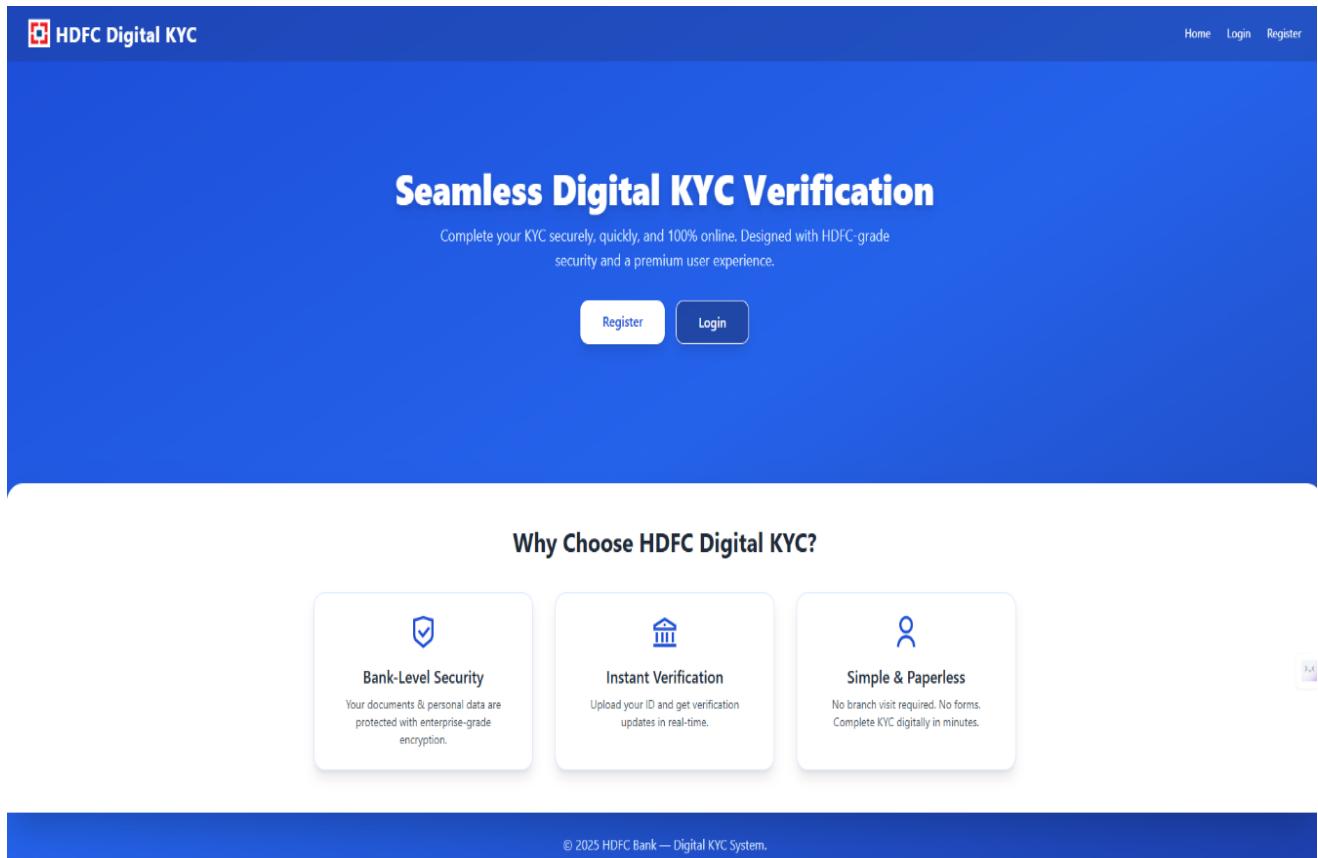
- React.js
- React Router
- TailwindCSS
- Framer Motion
- Axios
- jsPDF + html2canvas

Backend

- Node.js
- Express.js
- Sequelize ORM
- MySQL
- Multer (file uploads)
- JWT authentication

Project Workflow

1. Opening Page



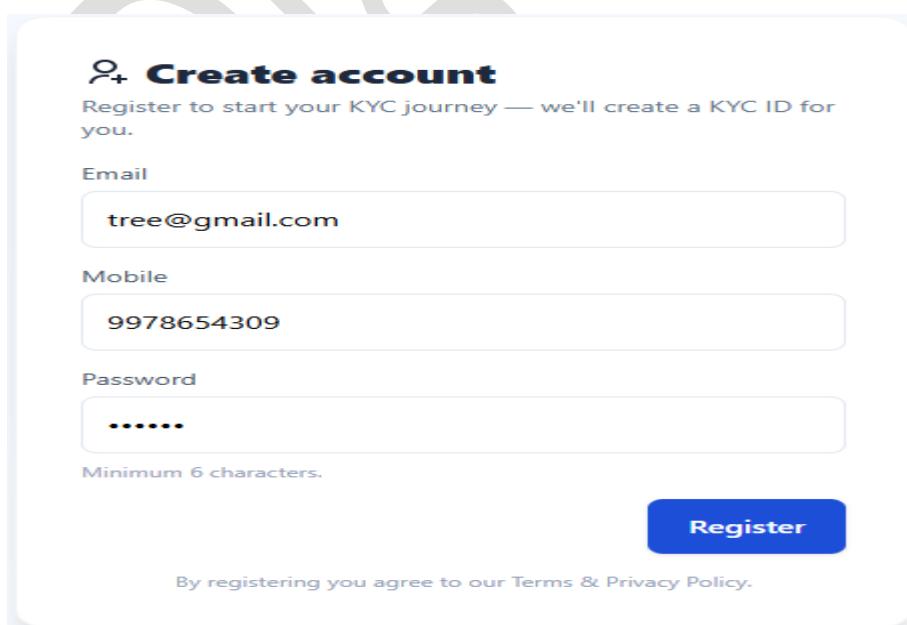
The screenshot shows the homepage of the HDFC Digital KYC system. At the top, there is a navigation bar with the HDFC logo, "HDFC Digital KYC", and links for "Home", "Login", and "Register". The main title "Seamless Digital KYC Verification" is centered, followed by a subtitle: "Complete your KYC securely, quickly, and 100% online. Designed with HDFC-grade security and a premium user experience." Below the subtitle are two buttons: "Register" and "Login".

Why Choose HDFC Digital KYC?

- Bank-Level Security**
Your documents & personal data are protected with enterprise-grade encryption.
- Instant Verification**
Upload your ID and get verification updates in real-time.
- Simple & Paperless**
No branch visit required. No forms. Complete KYC digitally in minutes.

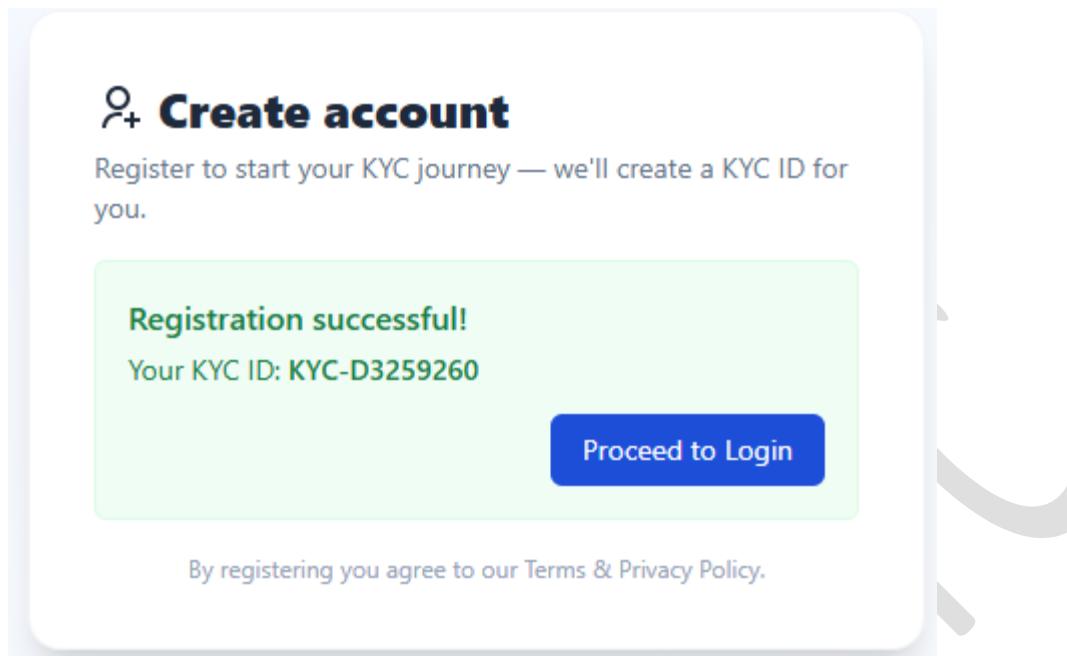
© 2025 HDFC Bank — Digital KYC System.

2. Registration Page after clicking on Register button

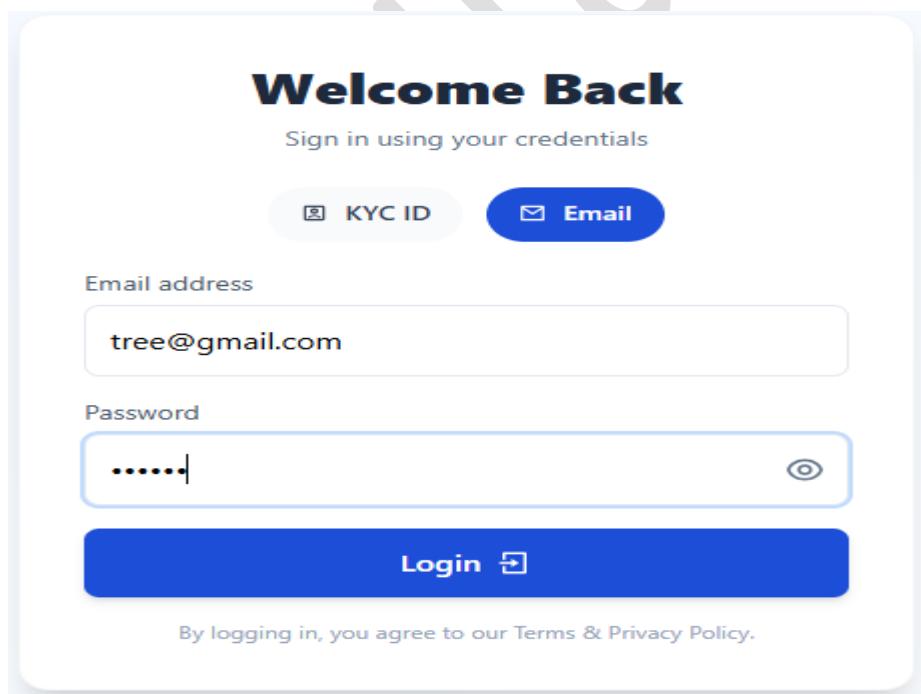


The screenshot shows the registration form page. The title is "Create account" with a user icon. A sub-instruction says "Register to start your KYC journey — we'll create a KYC ID for you." The form fields are: "Email" (input: tree@gmail.com), "Mobile" (input: 9978654309), "Password" (input: masked), and "Minimum 6 characters." A note below the password field says "Minimum 6 characters." A large blue "Register" button is at the bottom. A footer note says "By registering you agree to our Terms & Privacy Policy."

3. Registration successful.



4. Login page after clicking on the **Proceed to Login** button.



5. After successful login, the window of first step of the KYC opens. Here, we need to fill the personal Information.

Step 1 of 5

Personal Information

RAJ CHOUDHURY	10-06-1996
ANIL CHOUDHURY	Male
Guwahati	Assam
781026	
Nraengi_Guwahati	

Passport-size Photo*



Next →

6. KYC step 2 – Document Selection.

Step 2 of 5

Enter document details

Provide one Address Proof and one Identity Proof. Aadhaar + PAN are required for digital accounts.

Address proof

Aadhaar Card 123456765432

Identity proof

PAN Card ABCSE6068A

Back **Next →**

7. KYC Step 3 – Document Upload

Step 3 of 5

Upload your documents

Upload Address Proof and Identity Proof. Max 5.00 MB each.

Progress: 0%

Address Proof

Document type	Document number
Aadhaar	123456765432

File selected
Accepted: JPG, PNG, PDF - Max 5.00 MB
Aadhaar.png
89.4 KB

Identity Proof

Document type	Document number
Aadhaar	ABCSE6068A

File selected
Accepted: JPG, PNG, PDF - Max 5.00 MB
hdfc.png
3.6 KB

[← Back](#) [Upload & Next →](#)

Tip: For Aadhaar, use a clear photo showing all corners. For PAN, a scanned PDF or clear photo works best.

8. KYC Step 4 – Review & Confirmation

Step 4 of 5

Review & confirm your details

Progress: 80%

Please verify the details below before submission. You can edit any item if something looks wrong.

Personal details ✓ Completed

Name: RAJ CHOWDHURY
DOB: 1996-06-10
Father's Name: ANIL CHOWDHURY
Address: Nraengi, Guwahati, Guwahati 781026
Gender: Male

[Edit](#)

Selected document ✓ Selected

Aadhaar Card
This is the document type you chose in Step 2.

[Edit](#)

Document number ✓ Provided

*****3456
Masked in UI for privacy. Full value is submitted to verification service.

[Edit](#)

Uploaded file ✓ Uploaded

Passport Photo
The file you uploaded in Step 3 will be used to verify the document.

[Change](#)

[← Back](#) [Confirm →](#)

9. KYC Completed.

HDFC Digital Banking
KYC Portal

Home Dashboard Logout



10. The dashboard with the account details opens after clicking on **Go To Dashboard** button.

Welcome back
RAJ CHOUDHURY

Search banking services...

KYC Details

KYC Status: Completed

KYC ID: KYC-ZLJ3RWT4

Name: RAJ CHOUDHURY

Mobile: —

Account: XXXX-XXXX-7737

Primary Account

₹ — Available balance

Transfer Statement

Quick actions

Request Card Open FD

Banking Services

- Accounts
- Deposits
- Loans
- Investments
- Insurance
- Cards
- Payments
- Marketplace
- Travel

Confirmation Page



Digital KYC KYC Confirmation

Congratulations, RAJ CHOUDHURY!

Your KYC submission is received and recorded. Keep this confirmation for your records.

Applicant:

RAJ CHOUDHURY

KYC ID:

KYC-ZLJ3RWT4

Account No:

XXXX-XXXX-7737

Status:

Completed

Submitted at:

12/10/2025, 7:46:44 PM

Document:

Aadhaar Card / *****3456

Important

Please retain this confirmation. Full document numbers were submitted to the server securely and are masked here for privacy.

Frontend Design

- **Framework & libraries**
 - React (functional components + hooks) for UI and state management.
 - React Router for client-side routing (multi-step KYC flow).
 - TailwindCSS for utility-first styling and rapid theming.
 - Framer Motion for smooth micro-interactions and step transitions.
 - Axios for API requests; jsPDF / html2canvas used for client-side PDF generation.
- **Overall structure**
 - Single-page app with route-based pages: Landing, Register, Login, KYC steps (1–5), Result, Dashboard.
 - Shared Layout component for global header/footer; some pages (landing/dashboard) may use full-bleed layouts.
 - Lazy-loaded pages (React.lazy + Suspense) to reduce initial bundle size.
- **State & data flow**
 - Local component state (useState) for form inputs and small UI state.
 - useContext (AuthContext) to hold auth token, role and login/logout helpers.
 - LocalStorage caching for KYC step persistence (draft autosave) so users can resume.
 - Centralized API service (services/api.js) wrapping Axios and applying auth header via setToken.
- **Forms & validation**
 - Per-step validation rules (Aadhaar 12 digits, PAN regex, passport formats, file type/size).
 - Immediate inline error messaging and disabled/guarded Next buttons.
 - Accessibility: labels, focus states, keyboard support for drag/drop areas.
- **File upload UX**
 - Drag & drop + click-to-select file zones with file preview / filename & size.
 - Per-file client-side checks: MIME type whitelist (jpg/png/pdf) and max size (5 MB).
 - Upload progress bars (mapped to overall KYC progress when multiple files).
- **KYC progress**
 - Persistent progress bar across steps showing completion % (e.g., step-based mapping).
 - Visual completion ticks for completed sub-steps (personal details, documents selected, uploads done).

- **PDF confirmation**
 - Client-side generation of confirmation PDF (jsPDF/html2canvas) using on-screen review content to avoid blank pages/popups.
 - Single-click download that triggers a Blob-download, no navigations.
- **Security & UX considerations**
 - Avoid storing sensitive data in plain text; only non-sensitive draft fields in LocalStorage.
 - Mask sensitive numbers (account numbers show last digits only) in UI.
 - Rate-limit retry attempts on UI with clear user messaging (attempts left).
- **Performance & production**
 - Code splitting, route-level lazy loading, and caching of static assets.
 - Use environment variables for API URL; build-time minification.
 - Responsive design tested at mobile/tablet/desktop breakpoints.
- **Testing & quality**
 - Unit tests for critical form validation functions.
 - Manual E2E test flows for KYC happy-path and failure-path.
 - Linting (ESLint) and Prettier for consistent code style.

Backend Design

- **Platform & libraries**
 - Node.js + Express for REST API server.
 - Sequelize ORM for relational DB access (MySQL).
 - Multer for multipart file handling (uploads).
 - bcrypt for password hashing; jsonwebtoken (JWT) for auth tokens.
- **High-level architecture**
 - Stateless REST API endpoints serving the React frontend.
 - Persistent relational DB for users, KYCAplications, Documents, Attempts.
 - Local file storage under /uploads (or S3 for production) with generated randomized filenames.
- **Core models / tables**
 - Users — id, name, email, mobile, passwordHash, role, kycId.
 - KYCAplication — id, userId, status, failureReason, accountNumber, progress.
 - Documents — id, kycId, type (address/identity/passport/etc.), filePath, isValidated.
 - Attempts — id, kycId, step, count (for limiting retries).
 - Optional audit/logs table for actions and timestamps.

- **Key API endpoints**
 - POST /api/auth/register — create user + KYCApplication.
 - POST /api/auth/login — authenticate and return JWT.
 - GET /api/kyc/dashboard — returns user KYC summary and masked account number.
 - POST /api/kyc/upload — authentication + multer; stores file metadata; returns attemptsLeft.
 - POST /api/kyc/photo-match — runs simulation; updates status or rejects.
 - GET /api/kyc/result — final KYC status and failure reason.
- **Upload handling & validation**
 - Multer storage with disk (or S3) and randomized filenames.
 - File-filter to accept only jpg/jpeg/png/pdf and file-size limit (5MB).
 - Server-side validation of file type + basic content checks.
 - Attempts counter: increment Attempts per upload step and reject after max (e.g., 3).
- **Authentication & authorization**
 - JWT tokens signed with secret, short expiration (e.g., 1 day).
 - Middleware auth decodes token, loads user and KYCApplication and attaches to req.user.
 - Role checks if admin endpoints are added.
- **Business logic**
 - KYC state machine: Started → Documents Uploaded → Under Review → Completed / Rejected.
 - On successful completion, generate account number (pseudo-random with masked display) and set status Completed.
 - Create final confirmation record retrievable for PDF data.
- **Security controls**
 - Hash passwords with bcrypt (salted).
 - CORS configured to allow only the frontend domain.
 - Input validation for all request bodies to prevent injection & malformed data.
 - Sanitize filenames and store file metadata rather than exposing raw paths.
 - Use HTTPS in production; secure JWT storage & rotate secrets per policy.

- **Logging, monitoring & error handling**
 - Centralized error middleware that returns consistent error payloads.
 - Structured logs (timestamp, userId, route, error) and optional integration with log services (Papertrail, Loggly).
 - Return helpful but non-sensitive errors to clients.
- **Testing & reliability**
 - Unit tests for controllers and validation logic.
 - Integration tests for file upload endpoints and KYC flows.
 - DB migrations via Sequelize and schema version control.
 - Backups and retention policy for uploaded documents.
- **Scalability & deployment**
 - Containerize server (Docker) and use process manager (PM2) or Kubernetes in production.
 - Move uploads to object storage (S3) with CDN for performance.
 - Use connection pooling for DB; horizontal scaling behind load balancer.
 - Add rate-limiting middleware on public endpoints to prevent abuse.
- **Operational considerations**
 - Background worker queue (e.g., Bull) for CPU/IO heavy tasks (OCR, face-match in future).
 - Admin dashboard or alerting for rejected KYCs and manual review flows.
 - Endpoint to regenerate/download confirmation PDF server-side if needed (secure, authenticated).

Conclusion

- The Digital KYC portal successfully transforms a traditionally complex process into a streamlined, online, and user-friendly workflow.
- By combining React, TailwindCSS, Node.js, Express, and secure document handling, the system ensures high performance and reliability.
- The multi-step guided flow reduces user confusion, minimizes drop-offs, and ensures accurate data submission.
- Automated validations, progress tracking, and instant PDF generation provide a modern and professional onboarding experience.
- The final dashboard gives users clarity on their account status and personal details, increasing transparency and trust.
- The use of JWT authentication, hashed passwords, and controlled file uploads ensures strong system security.
- The modular structure of the project makes it scalable for future enhancements such as OCR, facial recognition, or real-time verification.

- Overall, the project demonstrates how digital transformation can significantly improve operational efficiency in customer onboarding.
- The Digital KYC system provides a solid foundation for banks aiming to adopt fully paperless and automated onboarding workflows.
- This project proves the feasibility and benefits of adopting modern full-stack development practices in financial technology solutions.
