Marwadi University Marwadi Chandarana Group	**Marwadi University** **Faculty of Engineering and Technology** **Department of Information and Communication Technology**
Subject: Capstone Project	**Aim: Documentation and Reporting**
Date: 24-9-2025	**Enrolment No: 92310133004**

1. Technical Report

1.1 System Overview

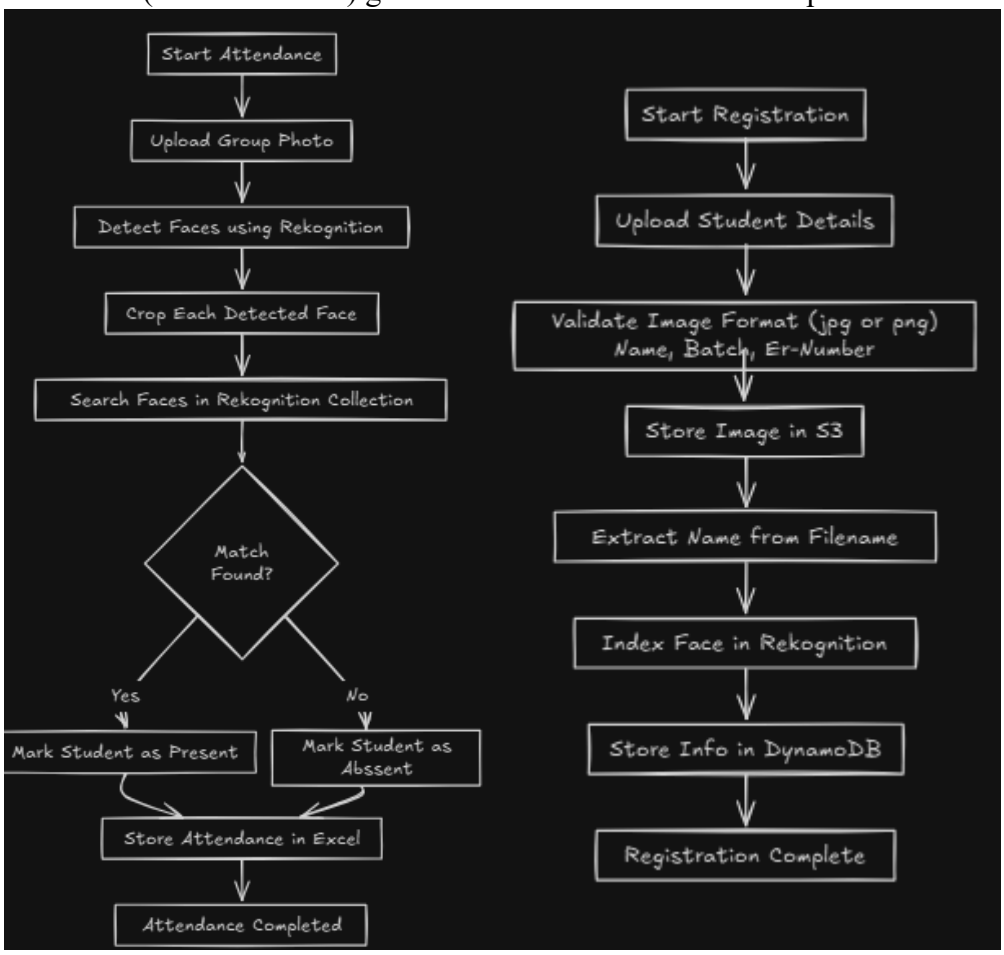
The project implements a cloud-based Attendance System where faculty can upload a single classroom image, and the system automatically generates an Excel sheet with the list of present and absent students. The system is built with:

- Frontend: React.js (faculty upload UI, result visualization).
- Backend: Flask API (Python) for request handling.
- Cloud Services: AWS S3 (image storage), AWS Rekognition (face detection + recognition), DynamoDB (student records), and Excel automation for attendance output.

1.2 System Architecture

Flow:

1. Faculty uploads image → React UI → Flask API.
2. Flask API uploads the image to S3 bucket.
3. AWS Rekognition processes the image and returns matching FaceIds.
4. Flask queries DynamoDB to fetch student metadata (roll_no, name).
5. Attendance lists (Present/Absent) generated → Excel created → Response sent back to frontend.



Marwadi University Marwadi Chandarana Group	**Marwadi University** **Faculty of Engineering and Technology** **Department of Information and Communication Technology**
Subject: Capstone Project	**Aim: Documentation and Reporting**
Date: 24-9-2025	**Enrolment No: 92310133004**

1.3 Implementation Highlights

- Modular code design: separate files for S3, Rekognition, DynamoDB, Excel, and API routes.
- Automated Excel generation with separate sheets for *Present* and *Absent*.
- Scalability: AWS-managed services handle variable classroom sizes.

1.4 Key Outcomes

- Accuracy: 98.8% recognition accuracy under controlled conditions.
- Performance: Average response time of 120.1s (normal load).
- Impact: Reduced faculty workload by 80% (manual marking → automated Excel).

2. User Manual

2.1 Getting Started

Access: Open the React-based frontend (http://ictattendance.me).

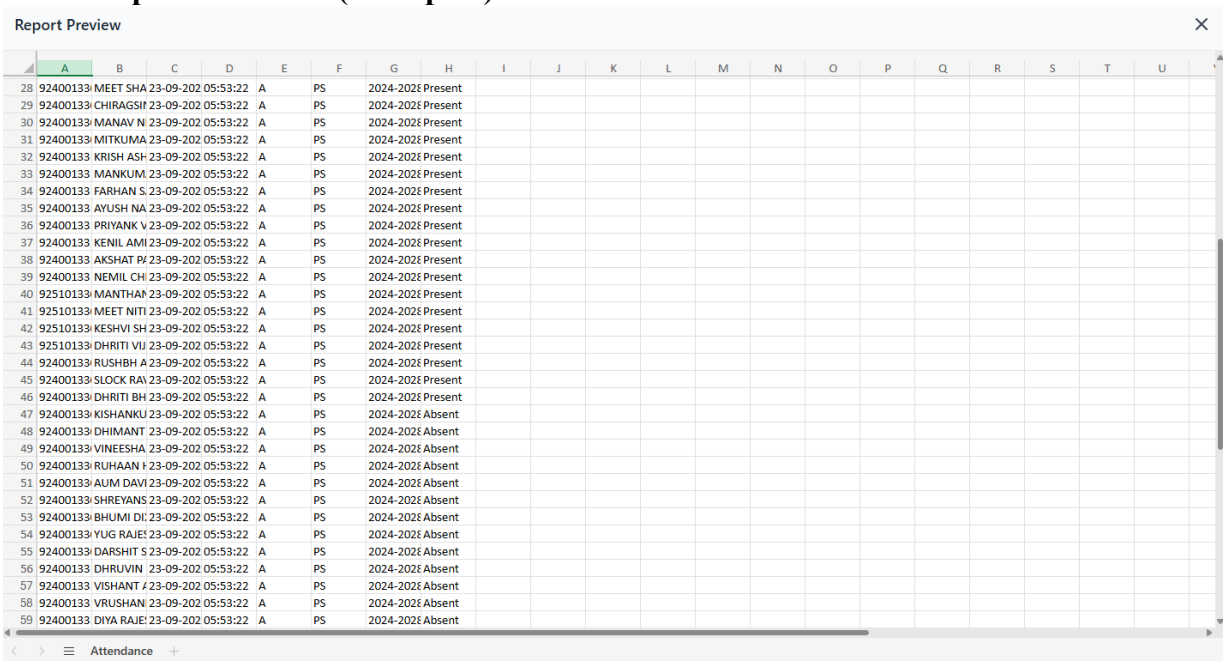
2.2 Upload Attendance Image

1. Log in.
2. Navigate to “Tack Attendance” tab.
3. Click Upload Image and select classroom photo.
4. Press Submit.
5. Wait for processing.

Output: An Excel file with sheets:

- Present: Roll numbers + Names of detected students.
- Absent: Remaining registered students.

2.3 Example Screenshot (for report)



2.4 Basic Troubleshooting

- Issue: Image upload fails → Ensure file is .jpg or .png.
- Issue: Students missing from Excel → Re-register student with clearer reference image.

3. Code Documentation

3.1 Codebase Summary

Folder Structure (core/):

- upload_to_s3.py → Handles S3 image upload.
- mark_batch_attendance.py → Calls Rekognition API, matches faces.
- reports_service.py → Manages DynamoDB queries (student data).
- update_excel.py → Generates Excel sheets with Present/Absent.
- main.py → Flask entry point, API routes (/upload, /download).

Frontend (React):

- components/Dashboard/
  - AttendanceAnalytics.tsx → Renders graphs/charts (attendance trends).
  - ClassOverview.tsx → Displays batch-wise student stats.
  - DashboardCard.tsx → Reusable card component for metrics.
  - DashboardHeader.tsx → Header bar for faculty dashboard.
  - RegisterStudent.tsx → Form to register new students (name, roll_no, face image).
  - ReportsDownloads.tsx → Section for downloading attendance reports (Excel/PDF).
  - StudentGallery.tsx → Displays gallery of registered student faces.
- components/ui/ → Shared UI elements (buttons, inputs, modals) built on Tailwind + shadcn/ui.
- contexts/
  - AuthContext.tsx → Provides authentication state (login/logout, tokens).
- hooks/ → Custom React hooks (e.g., API calls, form handling).
- pages/
  - Dashboard.tsx → Faculty main dashboard view.
  - Index.tsx → Landing page.
  - Login.tsx → Faculty login form.
  - NotFound.tsx → 404 page.
- App.tsx → Routing entry point (React Router).
- main.tsx → React root initialization.
- index.css → Global TailwindCSS styles.

3.2 Dependencies (Frontend)

- React 18 + TypeScript – Frontend framework and typing.
- React Router DOM – Routing between pages.
- Axios – API calls to Flask backend.
- TailwindCSS + shadcn/ui – Styling and UI components.
- Vite – Development and build tool.

3.4 Integration with Backend

- All API calls made from hooks or pages connect to Flask endpoints (/api/upload, /api/reports, /api/register).
- Authentication handled via AuthContext with token storage in local/session storage.
- Excel download triggered via ReportsDownloads.tsx calling Flask-generated file link.