Student Progress Management System

Assignment Submission Documentation

Project Title: Student Progress Management System
Development Stack: MERN (MongoDB, Express.js, React, Node.js)
Submission Date: June 19, 2025
GitHub Repository:https://github.com/pulkitgupta0624/Student-Progress-Management-System-Codeforces
Video:https://drive.google.com/file/d/1_w8BM7ag7XRhyRqmtX6L-s445jgBSr76/view?usp=sharing

---

Product Overview

Description

A comprehensive web application for tracking and managing student progress on the Codeforces competitive programming platform. The system automatically synchronizes student data, provides detailed analytics, and sends automated notifications to inactive students.

Key Objectives

- **Automated Data Management**: Sync Codeforces data automatically using cron jobs
- **Comprehensive Analytics**: Detailed progress tracking with visualizations
- **Student Engagement**: Automated email reminders for inactive students
- **User-Friendly Interface**: Modern, responsive design with dark/light themes

---

Features Implemented

✅ Core Requirements

1. Student Table View

- **Student Listing**: Complete table with Name, Email, Phone, Codeforces Handle, Current Rating, Max Rating
- **CRUD Operations**: Add, Edit, Delete students with form validation

- **CSV Export**: Download complete student dataset
- **Search & Filter**: Real-time search functionality
- **View Details**: Navigation to individual student profiles
- **Last Updated**: Display when data was last synchronized

2. Student Profile View

Contest History Section

- **Time Filters**: 30, 90, 365 days options
- **Rating Graph**: Interactive chart showing rating progression
- **Contest List**: Detailed table with rating changes and ranks
- **Performance Metrics**: Win rate, average rank, rating trends

Problem Solving Data Section

- **Time Filters**: 7, 30, 90 days options
- **Statistics Display**:
  - Most difficult problem solved (by rating)
  - Total problems solved
  - Average rating of solved problems
  - Average problems per day
- **Visualizations**:
  - Bar chart of problems by rating buckets
  - Submission heatmap with activity patterns

3. Codeforces Data Sync

- **Automated Sync**: Daily cron job (default: 2 AM)
- **Configurable Schedule**: Admin can change sync time and frequency
- **Real-time Sync**: Immediate sync when CF handle is updated
- **Data Storage**: Complete user info, contests, and submissions
- **Error Handling**: Graceful handling of API failures

4. Inactivity Detection

- **Automatic Detection**: Identify students inactive for 7+ days
- **Email Notifications**: Automated reminder emails with personalized content
- **Reminder Tracking**: Count of emails sent per student
- **Email Toggle**: Option to disable reminders per student

✅ Bonus Features Implemented

- **Responsive Design**: Mobile and tablet optimized
- **Dark/Light Theme**: Toggle with smooth transitions
- **Enhanced UI/UX**: Modern animations and micro-interactions
- **Comprehensive Documentation**: Well-documented codebase

Technology Stack

Backend

- **Node.js**: Runtime environment
- **Express.js**: Web framework
- **MongoDB**: NoSQL database
- **Mongoose**: Object Data Modeling (ODM)
- **node-cron**: Task scheduling
- **Nodemailer**: Email service
- **Axios**: HTTP client for API requests

Frontend

- **React 19**: Frontend library
- **TypeScript**: Type safety
- **Tailwind CSS**: Utility-first CSS framework
- **React Router**: Client-side routing
- **Recharts**: Data visualization
- **Date-fns**: Date manipulation

Development Tools

- **Vite**: Build tool
- **ESLint**: Code linting
- **Git**: Version control

Project Structure

```
student-progress-management/
├── backend/
│   ├── src/
│   │   ├── controllers/       # Request handlers
│   │   ├── models/            # Database schemas
│   │   ├── routes/            # API routes
│   │   ├── services/          # Business logic
│   │   ├── middleware/        # Custom middleware
│   │   └── app.js             # Main application
│   ├── package.json
│   └── .env                   # Environment variables
├── frontend/
│   ├── src/
```

```
│   │   ├── components/       # React components
│   │   ├── contexts/         # React contexts
│   │   ├── hooks/            # Custom hooks
│   │   ├── services/         # API services
│   │   ├── types/            # TypeScript types
│   │   ├── utils/            # Utilities
│   │   └── App.tsx           # Main component
│   ├── package.json
│   └── tailwind.config.js
└── README.md
```

API Documentation

Base URL: `http://localhost:5000/api`

Student Management

Get All Students

GET /students
Response: Array of student objects

Create Student

POST /students
Body: {
  "name": "string",
  "email": "string",
  "phoneNumber": "string",
  "codeforcesHandle": "string"
}

Update Student

PUT /students/:id
Body: {
  "name": "string",
  "email": "string",
  "emailRemindersEnabled": "boolean"
}

Delete Student

DELETE /students/:id

Get Student Codeforces Data
GET /students/:id/codeforces?days=30
Response: {
  "student": {...},
  "codeforcesData": {...},
  "analytics": {...}
}

Manual Sync Student Data
POST /students/:id/sync

System Management

Get Settings
GET /settings

Update Cron Schedule
PUT /settings/cron
Body: { "schedule": "0 3 * * *" }

Manual System Sync
POST /settings/sync

Data Export

Export Students CSV
GET /students/export/csv
Response: CSV file download

User Interfaces

1. Dashboard (Student Table View)

URL: /

Features:

- Statistics cards showing total students, active count, average rating
- Search bar for filtering students
- Student table with all required columns

- Action buttons for each row (View, Edit, Delete, Sync)
- Add Student and Export CSV buttons
- Theme toggle in navigation

Key UI Elements:

- Responsive table with mobile optimization
- Rating color-coding based on Codeforces standards
- Loading states and animations
- Status indicators for email reminders

2. Student Profile View

URL: `/student/:id`

Features:

- Student information header with avatar and details
- Contest History section with:
    - Time period filters (30/90/365 days)
    - Interactive rating graph
    - Contest details table
    - Performance statistics
- Problem Solving section with:
    - Time period filters (7/30/90 days)
    - Key metrics display
    - Rating distribution chart
    - Submission heatmap

Key UI Elements:

- Smooth animations and transitions
- Interactive charts with tooltips
- Responsive layout for mobile devices
- Back navigation to main dashboard

3. Modal Forms

- **Add/Edit Student**: Form with validation
- **Delete Confirmation**: Safety confirmation dialog
- **Settings**: System configuration options

4. Theme Support

- **Light Mode**: Clean, professional appearance
- **Dark Mode**: Easy on eyes for extended use
- **Smooth Transitions**: Animated theme switching

Installation Guide

Prerequisites

- Node.js (v16+)
- MongoDB (v4.4+)
- npm or yarn

Backend Setup

```
# Navigate to backend directory
cd backend

# Install dependencies
npm install

# Create environment file
cp .env.example .env

# Configure environment variables:
PORT=5000
MONGODB_URI=mongodb://localhost:27017/student_progress
EMAIL_USER=your-email@gmail.com
EMAIL_PASS=your-app-password

# Start development server
npm run dev
```

Frontend Setup

```
# Navigate to frontend directory
cd frontend

# Install dependencies
npm install

# Start development server
npm run dev
```

Access Application

- Frontend: `http://localhost:5173`
- Backend API: `http://localhost:5000`