# Student Attendance Management System - Complete Documentation

## Table of Contents

1. [Tech Stack](#tech-stack)
2. [Features](#features)
3. [AI Prompts Used in Development](#ai-prompts-used-in-development)
4. [Project Structure](#project-structure)
5. [Installation & Setup](#installation--setup)
6. [API Endpoints](#api-endpoints)
7. [Database Schema](#database-schema)

## Tech Stack

### Backend Technologies

* **Framework**: Django 5.2.5 (Python Web Framework)
* **Database**: SQLite3 (Development Database)
* **Authentication**: Custom JWT (JSON Web Tokens) implementation
* **API Architecture**: RESTful API with Django Views
* **File Generation**:
  + ReportLab 4.4.3 (PDF generation)
  + XlsxWriter 3.2.5 (Excel file generation)
* **AI Integration**: Google Generative AI (Gemini 1.5 Flash)
* **Environment Management**: python-dotenv 1.1.1

### Frontend Technologies

* **Framework**: React 19.1.1
* **Routing**: React Router DOM 7.8.0
* **HTTP Client**: Axios 1.11.0
* **Charts**: Recharts 3.1.2 (for data visualization)
* **Form Handling**: React Hook Form 7.62.0
* **PWA Support**: Progressive Web App capabilities
* **Styling**: Custom CSS with inline styles

### Development & Deployment

* **Package Management**: npm (Frontend), pip (Backend)
* **CORS Handling**: django-cors-headers 4.7.0
* **Password Hashing**: bcrypt 4.3.0
* **Environment**: Windows development environment
* **Version Control**: Git (with .gitignore)

### AI & External Services

* **Generative AI**: Google Gemini 1.5 Flash API
* **API Libraries**:
  + google-generativeai 0.8.5
  + google-ai-generativelanguage 0.6.15
  + google-api-python-client 2.179.0

## Features

### 1. User Management System

* **Multi-Role Authentication**: Admin, Teacher, Student roles
* **Registration & Approval Workflow**:
  + Users register with role selection
  + Admin approval required for activation
  + Class assignment during registration
* **Profile Management**: Users can update username and password
* **Role-Based Access Control**: Different dashboards and permissions per role

### 2. Admin Panel Features

* **User Management**:
  + View all users with filtering (by role, class)
  + Approve/disapprove user registrations
  + Delete users (except admins)
  + Monitor user status and class assignments
* **System Oversight**: Complete control over the attendance system
* **Class Management**: Ensure only one teacher per class

### 3. Teacher Panel Features

* **Attendance Management**:
  + Mark daily attendance for assigned class students
  + View student attendance history
  + Prevent duplicate attendance marking for same day
  + Visual indicators for low attendance students (<75%)
* **Report Generation**:
  + Download attendance reports in PDF format
  + Download attendance reports in Excel (XLSX) format
  + Comprehensive reports with student details and statistics
* **AI-Powered Chat Assistant**:
  + Integration with Google Gemini 1.5 Flash
  + Persistent chat history by date
  + Full-screen chat interface
  + Markdown rendering for AI responses
  + Date-wise chat history navigation
* **Student Monitoring**:
  + View all students in assigned class
  + Individual student attendance percentage tracking
  + Detailed attendance history for each student

### 4. Student Panel Features

* **Personal Attendance Tracking**:
  + View personal attendance history
  + See attendance percentage and statistics
  + Date-wise attendance records
* **Profile Management**: Update personal information

### 5. Advanced Features

* **Real-time Attendance Tracking**: Live updates of attendance percentages
* **Data Visualization**: Charts and graphs for attendance trends
* **Responsive Design**: Works on desktop and mobile devices
* **PWA Support**: Can be installed as a Progressive Web App
* **Secure Authentication**: JWT-based authentication with HTTP-only cookies
* **Data Export**: Multiple format support (PDF, Excel)

### 6. AI Integration Features

* **Conversational AI**: Teachers can ask questions to Gemini AI
* **Persistent Chat History**: All conversations saved with timestamps
* **Date-based Chat Navigation**: Browse chat history by specific dates
* **Markdown Support**: Rich text formatting in AI responses
* **Full-screen Chat Mode**: Enhanced chat experience

## AI Prompts Used in Development

### 1. Initial Project Setup Prompts

"Create a student attendance management system with Django backend and React frontend.   
Include user roles (Admin, Teacher, Student) with JWT authentication."

### 2. Database Design Prompts

"Design Django models for a school attendance system with:  
- Custom User model with roles (Admin, Teacher, Student)  
- Attendance tracking with date and status  
- Class assignment system  
- User approval workflow"

### 3. Authentication System Prompts

"Implement JWT authentication in Django without using external libraries.   
Include login, logout, registration with role-based access control."

### 4. Frontend Development Prompts

"Create React components for:  
- Login and registration forms  
- Role-based dashboards (Admin, Teacher, Student)  
- Attendance marking interface for teachers  
- Student attendance history display"

### 5. Report Generation Prompts

"Add PDF and Excel report generation functionality using ReportLab and XlsxWriter.   
Include student details, attendance statistics, and formatted layouts."

### 6. AI Integration Prompts

"Integrate Google Gemini AI into the teacher panel:  
- Allow teachers to ask questions  
- Save chat history with timestamps  
- Display responses with markdown formatting  
- Add date-wise chat navigation"

### 7. UI/UX Enhancement Prompts

"Improve the user interface with:  
- Modern styling and responsive design  
- Visual indicators for low attendance  
- Modal dialogs for detailed views  
- Full-screen modes for better user experience"

### 8. Data Visualization Prompts

"Add attendance trend charts using Recharts:  
- Class-wise attendance statistics  
- Date-wise attendance trends  
- Student performance indicators"

### 9. Security Enhancement Prompts

"Implement security best practices:  
- CORS configuration for React frontend  
- Secure JWT token handling  
- Input validation and error handling"

### 10. File Management Prompts

"Add file download functionality:  
- Generate attendance reports in multiple formats  
- Handle file streaming and downloads  
- Add proper MIME types and headers"

### 11. Advanced Features Prompts

"Enhance the system with:  
- Real-time attendance percentage calculations  
- Student attendance history modals  
- Profile update functionality  
- Attendance marking validation (prevent duplicates)"

## Project Structure

Student Attendance App v2/  
├── backend/  
│ ├── attendance/  
│ │ ├── migrations/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── admin.py  
│ │ ├── apps.py  
│ │ ├── models.py # User, Attendance, GenAIChat models  
│ │ ├── views.py # Main API endpoints  
│ │ ├── views\_extra.py # Additional endpoints  
│ │ ├── urls.py # URL routing  
│ │ └── tests.py  
│ ├── \_\_init\_\_.py  
│ ├── .env # Environment variables (Gemini API key)  
│ ├── asgi.py  
│ ├── db.sqlite3 # SQLite database  
│ ├── manage.py  
│ ├── settings.py # Django configuration  
│ ├── urls.py # Main URL configuration  
│ └── wsgi.py  
├── frontend/  
│ ├── public/  
│ │ ├── favicon.ico  
│ │ ├── index.html  
│ │ ├── manifest.json # PWA manifest  
│ │ └── robots.txt  
│ ├── src/  
│ │ ├── pages/  
│ │ │ ├── AdminPanel.js  
│ │ │ ├── Dashboard.js  
│ │ │ ├── Login.js  
│ │ │ ├── Register.js  
│ │ │ ├── StudentPanel.js  
│ │ │ └── TeacherPanel.js  
│ │ ├── utils/  
│ │ │ ├── AuthContext.js # Authentication context  
│ │ │ ├── axios.js # HTTP client configuration  
│ │ │ └── markdown.js # Markdown rendering utility  
│ │ ├── App.css  
│ │ ├── App.js # Main React component  
│ │ ├── index.js  
│ │ └── service-worker.js # PWA service worker  
│ ├── package.json  
│ └── package-lock.json  
├── .gitignore  
├── requirements.txt # Python dependencies  
└── Readme.md # Basic setup instructions

## Installation & Setup

### Prerequisites

* Python 3.8+
* Node.js 14+
* npm or yarn
* Google Gemini API key

### Backend Setup

1. **Create Virtual Environment**:

* python -m venv venv  
  venv\Scripts\activate # Windows  
  # or  
  source venv/bin/activate # Linux/Mac

1. **Install Dependencies**:

* pip install -r requirements.txt

1. **Environment Configuration**: Create .env file in backend directory:

* GENAI\_API\_KEY=your\_gemini\_api\_key\_here

1. **Database Setup**:

* cd backend  
  python manage.py migrate  
  python manage.py createsuperuser # Create admin user

1. **Run Backend Server**:

* python manage.py runserver

### Frontend Setup

1. **Install Dependencies**:

* cd frontend  
  npm install

1. **Run Frontend Server**:

* npm start

### Access the Application

* Frontend: http://localhost:3000
* Backend API: http://localhost:8000

### Default Credentials

* Admin: username: admin, password: admin
* Teacher: username: teacher10, password: teacher10

## API Endpoints

### Authentication Endpoints

* POST /login/ - User login
* POST /logout/ - User logout
* POST /register/ - User registration
* GET /me/ - Get current user info
* POST /profile/update/ - Update user profile

### Admin Endpoints

* GET /users/ - List all users (with filters)
* POST /approve\_user/ - Approve/disapprove users
* POST /delete\_user/ - Delete user accounts

### Teacher Endpoints

* GET /my\_students/ - Get students in teacher’s class
* POST /attendance/mark/ - Mark daily attendance
* GET /attendance/marked\_today/ - Check if attendance marked today
* GET /attendance/report/ - Download attendance reports (PDF/XLSX)
* POST /genai/ask/ - Ask question to Gemini AI
* GET /genai/ask/ - Get chat history for date
* GET /genai/chat\_dates/ - Get available chat dates

### Student Endpoints

* GET /attendance/history/ - Get personal attendance history

### General Endpoints

* GET /attendance/history/?student\_id=X - Get student attendance (Admin/Teacher)
* GET /class\_attendance\_trends/ - Get class attendance statistics

## Database Schema

### User Model

class User(AbstractBaseUser):  
 username = CharField(max\_length=150, unique=True)  
 role = CharField(choices=['Admin', 'Teacher', 'Student'])  
 is\_active = BooleanField(default=False)  
 is\_approved = BooleanField(default=False)  
 class\_assigned = PositiveSmallIntegerField(null=True, blank=True)  
 roll\_number = PositiveIntegerField(null=True, blank=True)  
 is\_staff = BooleanField(default=False)  
 is\_superuser = BooleanField(default=False)

### Attendance Model

class Attendance(Model):  
 student = ForeignKey(User, on\_delete=CASCADE)  
 date = DateField(default=timezone.now)  
 status = CharField(choices=['Present', 'Absent'])  
 timestamp = DateTimeField(auto\_now\_add=True)

### GenAIChat Model

class GenAIChat(Model):  
 teacher = ForeignKey(User, on\_delete=CASCADE)  
 prompt = TextField()  
 response = TextField()  
 date = DateField(default=timezone.now)  
 timestamp = DateTimeField(auto\_now\_add=True)

## Key Features Implementation Details

### JWT Authentication

* Custom implementation without external libraries
* HTTP-only cookies for security
* Role-based access control
* Token expiration handling

### AI Integration

* Google Gemini 1.5 Flash API integration
* Persistent chat history storage
* Markdown rendering for responses
* Date-wise conversation management

### Report Generation

* PDF reports using ReportLab
* Excel reports using XlsxWriter
* Comprehensive attendance statistics
* Downloadable file streaming

### Real-time Features

* Live attendance percentage calculations
* Instant UI updates after attendance marking
* Dynamic student status indicators

This documentation provides a complete overview of the Student Attendance Management System, including all technical details, features, and the AI-assisted development process used to create this comprehensive application.