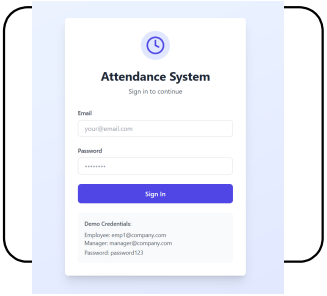
Employee Attendance System

A full-stack web application for managing employee attendance with role-based access for employees and managers.



📋 Table of Contents

✨ Features

Employee Features

- ✅ Register/Login with secure authentication
- ✅ Check In/Check Out with timestamp tracking
- ✅ View attendance history (table view)
- ✅ View monthly summary (Present/Absent/Late days)
- ✅ Dashboard with personal statistics
- ✅ Real-time status updates

Manager Features

- ✅ Login with manager credentials
- ✅ View all employees' attendance records
- ✅ Filter by employee, date, and status
- ✅ View team attendance summary
- ✅ Export attendance reports to CSV
- ✅ Dashboard with team statistics
- ✅ Weekly attendance trends
- ✅ Department-wise breakdown

🛠 Tech Stack

Frontend

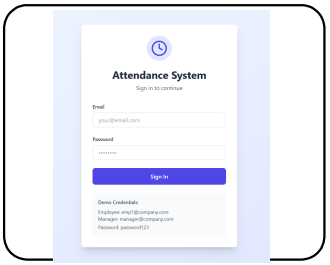
- React 18
- Redux Toolkit (State Management)
- Axios (HTTP Client)
- Lucide React (Icons)
- Tailwind CSS (Styling)

Backend

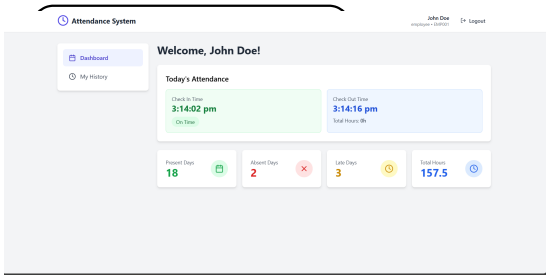
- Node.js
- Express.js
- MongoDB (Database)
- Mongoose (ODM)
- JWT (Authentication)
- bcryptjs (Password Hashing)

📸 Screenshots

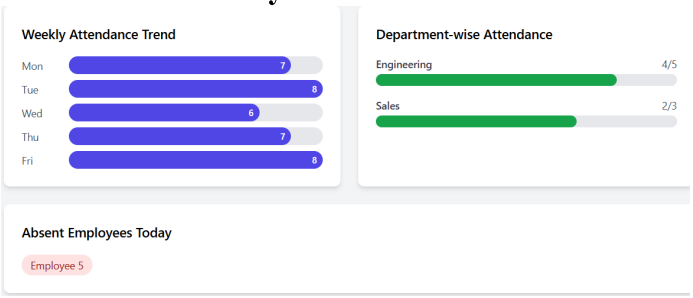
Login Page



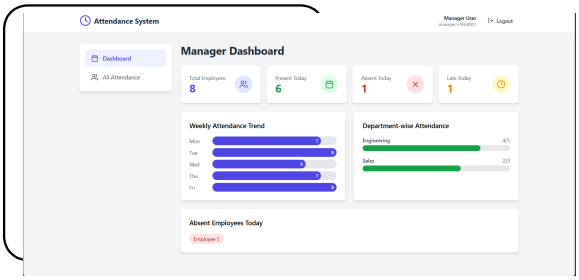
Employee Dashboard



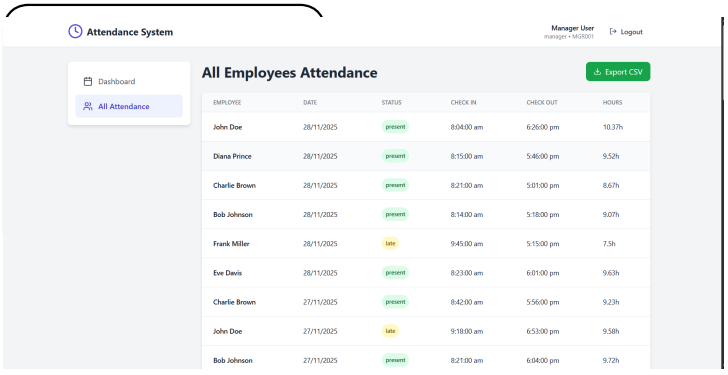
Attendance History



Manager Dashboard



All Attendance View



📦 Installation

Prerequisites

- Node.js (v14 or higher)
- MongoDB (v4.4 or higher)
- npm or yarn

Clone the Repository

```bash
git clone https://github.com/yourusername/employee-attendance-system.git
cd employee-attendance-system
```

Backend Setup

```bash
# Navigate to backend directory
cd backend

# Install dependencies
npm install

# Create .env file (see Environment Variables section)
cp .env.example .env

# Update .env with your MongoDB URI and JWT secret
```

Frontend Setup

```bash
# Navigate to frontend directory
cd ../frontend

# Install dependencies
npm install

# Create .env file
cp .env.example .env

# Update .env with your API URL
```

🔐 Environment Variables

Backend `.env`

```env
MONGO_URI=mongodb://localhost:27017/attendance_system
JWT_SECRET=your_super_secret_jwt_key_change_this_in_production
PORT=5000
NODE_ENV=development
```

Frontend `.env`

```env
REACT_APP_API_URL=http://localhost:5000/api
```

🚀 Running the Application

Step 1: Start MongoDB

```bash
# On macOS with Homebrew
brew services start mongodb-community

# On Linux
sudo systemctl start mongod

# On Windows
# Start MongoDB service from Services
```

Step 2: Seed Database (Optional but Recommended)

```bash
cd backend
npm run seed
```

This creates:

- 1 Manager account
- 8 Employee accounts

- 30 days of sample attendance data

Step 3: Start Backend Server

```bash
cd backend
npm run dev
```

The backend will run on `http://localhost:5000`

Step 4: Start Frontend Application

```bash
cd frontend
npm start
```

The frontend will run on `http://localhost:3000`

🔑 Demo Credentials

Manager Account

- **Email:** manager@company.com
- **Password:** password123

Employee Accounts

- **Email:** emp1@company.com (or emp2, emp3, ... emp8)
- **Password:** password123

📡 API Documentation

Authentication Endpoints

Register User

```http
```

```
POST /api/auth/register
Content-Type: application/json

{
  "name": "John Doe",
  "email": "john@company.com",
  "password": "password123",
  "role": "employee",
  "employeeId": "EMP009",
  "department": "Engineering"
}
```

Login

```
http

POST /api/auth/login
Content-Type: application/json

{
  "email": "emp1@company.com",
  "password": "password123"
}
```

Get Current User

```
http

GET /api/auth/me
Authorization: Bearer <token>
```

Attendance Endpoints (Employee)

Check In

```
http

POST /api/attendance/checkin
Authorization: Bearer <token>
```

Check Out

```
http
```

```
POST /api/attendance/checkout
Authorization: Bearer <token>
```

Get My History

```http
GET /api/attendance/my-history?month=11&year=2024
Authorization: Bearer <token>
```

Get My Summary

```http
GET /api/attendance/my-summary?month=11&year=2024
Authorization: Bearer <token>
```

Get Today's Status

```http
GET /api/attendance/today
Authorization: Bearer <token>
```

Attendance Endpoints (Manager)

Get All Attendance

```http
GET /api/attendance/all?status=present&startDate=2024-11-01
Authorization: Bearer <token>
```

Get Team Summary

```http
GET /api/attendance/summary
Authorization: Bearer <token>
```

Export to CSV

```http
GET /api/attendance/export
Authorization: Bearer <token>
```

Dashboard Endpoints

Employee Dashboard

```http
GET /api/dashboard/employee
Authorization: Bearer <token>
```

Manager Dashboard

```http
GET /api/dashboard/manager
Authorization: Bearer <token>
```

📁 Project Structure

```
employee-attendance-system/
├── backend/
│   ├── config/
│   │   └── db.js
│   ├── controllers/
│   │   ├── authController.js
│   │   ├── attendanceController.js
│   │   └── dashboardController.js
│   ├── middleware/
│   │   └── auth.js
│   ├── models/
│   │   ├── User.js
│   │   └── Attendance.js
│   ├── routes/
│   │   ├── auth.js
│   │   ├── attendance.js
│   │   └── dashboard.js
│   ├── utils/
│   │   ├── csvExport.js
│   │   └── seed.js
```

```
│   ├── .env
│   ├── .env.example
│   ├── package.json
│   └── server.js
│
└── frontend/
    ├── public/
    ├── src/
    │   ├── redux/
    │   │   ├── slices/
    │   │   │   ├── authSlice.js
    │   │   │   └── attendanceSlice.js
    │   │   └── store.js
    │   ├── services/
    │   │   └── api.js
    │   ├── App.jsx
    │   ├── index.js
    │   └── index.css
    ├── .env
    ├── .env.example
    └── package.json
```

🗄️ Database Schema

Users Collection

javascript

```
{
  _id: ObjectId,
  name: String,
  email: String (unique),
  password: String (hashed),
  role: String (employee/manager),
  employeeId: String (unique),
  department: String,
  createdAt: Date,
  updatedAt: Date
}
```

Attendance Collection

javascript

```
{
  _id: ObjectId,
  user: ObjectId (ref: User),
  date: Date,
  checkInTime: Date,
  checkOutTime: Date,
  status: String (present/absent/late/half-day),
  totalHours: Number,
  createdAt: Date,
  updatedAt: Date
}
```

🧪 Testing

Test Employee Flow

1. Login with employee credentials
2. Check in (should show current time)
3. Navigate to History page
4. Check out
5. Verify total hours calculated

Test Manager Flow

1. Login with manager credentials
2. View dashboard statistics
3. Navigate to All Attendance
4. Apply filters (employee, date, status)
5. Export CSV report

🚢 Deployment

Backend Deployment (Railway/Render)

1. Create account on Railway or Render
2. Connect GitHub repository
3. Set environment variables
4. Deploy

Frontend Deployment (Vercel/Netlify)

1. Create account on Vercel or Netlify
2. Connect GitHub repository
3. Set environment variable: `REACT_APP_API_URL`
4. Deploy

🤝 Contributing

Contributions are welcome! Please follow these steps:

1. Fork the repository
2. Create a feature branch (`git checkout -b feature/AmazingFeature`)
3. Commit your changes (`git commit -m 'Add some AmazingFeature'`)
4. Push to the branch (`git push origin feature/AmazingFeature`)
5. Open a Pull Request

📄 License

This project is licensed under the MIT License.

👨‍💻 Author

Your Name

- GitHub: @yourusername
- Email: your.email@example.com

🙏 Acknowledgments

- React Team for amazing frontend library
- MongoDB for powerful database
- Express.js for robust backend framework
- Tailwind CSS for beautiful styling

Happy Coding! 🎉