# Teacher Management UI Documentation

## Setup & Installation

1. Clone the repository:  
 git clone https://github.com/yourusername/teacher-management-ui.git  
 cd teacher-management-ui  
  
2. Install dependencies:  
 npm install  
  
3. Run the development server:  
 npm run dev  
  
4. Build & production:  
 npm run build  
 npm start  
  
5. Environment requirements:  
 - Node.js v18+  
 - Next.js 15  
 - TypeScript  
 - Tailwind CSS

## Features

**Teacher Management:**- Add and delete teacher profiles.  
- View teacher details in card format.  
  
**Attendance Tracking:**- Select any date to mark attendance.  
- Toggle present/absent for each teacher.  
  
**Salary Calculation & Payment:**- Calculate salary based on attendance proportion.  
- Track cumulative payments and remaining balance.  
- Dedicated QR payment flow with UPI.  
- Print salary slip after payment.  
  
**Persistence & State:**- Data stored in localStorage.  
- Centralized state via React Context.

## Design Decisions & Rationale

- Frontend-only for simplicity and portability.  
- Next.js App Router for rapid routing setup.  
- TypeScript and Zod for type safety and validation.  
- React Context for centralized state management.  
- Tailwind CSS for utility-first styling.  
- Incremental payment logic to preserve payment history.

## Assumptions

- Salary is prorated by the ratio of days present to total working days in a month.  
- Attendance records are marked as full-day present or absent.  
- UPI is the sole payment method.  
- LocalStorage persistence: data not shared across devices.  
- No authentication implemented.

## Expanded Setup & Installation Instructions

1. Clone the repository from GitHub. This step downloads the source code of the Teacher Management UI project to your local machine:  
 git clone https://github.com/yourusername/teacher-management-ui.git  
 cd teacher-management-ui  
  
2. Install dependencies using npm. This command reads the package.json file and installs all required packages:  
 npm install  
  
3. Start the development server:  
 npm run dev  
 This will run the app in development mode, typically available at http://localhost:3000.  
  
4. For production, build and start the optimized app:  
 npm run build  
 npm start  
  
5. Ensure you have the correct development environment:  
 - Node.js (v18 or newer)  
 - Internet access to fetch packages from npm registry  
 - An IDE like VS Code is recommended for development

## Detailed Feature Explanations

1. Teacher Management:  
- Add Teacher: Users can add new teachers by providing essential information such as full name, subject, email, UPI ID for payment, monthly salary, and attendance count.  
- Delete Teacher: Remove any teacher profile permanently from the system.  
- View Teachers: Teachers are displayed in card format with quick action buttons for management.  
  
2. Attendance Tracking:  
- Date Selection: A date picker allows the admin to choose any specific date.  
- Mark Attendance: Toggle switch or checkbox per teacher marks them present or absent for the selected date.  
  
3. Salary Management:  
- Salary is calculated proportionally based on the number of days a teacher is marked present compared to the total working days of the month.  
- Each teacher has a salary table entry showing base salary, days attended, due amount, paid amount, and pending balance.  
- If the pending balance is greater than zero, a 'Pay' button redirects to a UPI QR code page.  
- After payment, the system records the transaction and updates the paid amount.  
  
4. Payment Tracking:  
- Every payment is recorded with a cumulative value under each teacher.  
- Updating attendance does not remove past payments; the system recalculates the difference.  
- Payments can be made in parts, and status updates dynamically as 'Paid' or 'Partially Paid'.  
  
5. Print Salary Slip:  
- Once a teacher is fully paid, a 'Print Slip' button becomes available.  
- This generates a printable salary slip that includes teacher details, attendance, base salary, and paid amount.

## Expanded Assumptions

- The app assumes there is only one admin or user using the system from a single machine.  
- All data is stored locally in the browser (localStorage) and is not shared across users.  
- Payment tracking is limited to UPI and does not integrate with external payment verification APIs.  
- It is assumed that teachers work a standard full month and attendance is tracked on a per-day basis.  
- No login or role-based access control is implemented; the UI is open.  
- The application does not handle real-time sync or cloud backup.  
- Attendance and payments are managed independently per teacher profile.