MediTrack System – Frontend Documentation

1. Introduction

The frontend of the **MediTrack System** is developed with **React (Vite)** and provides a professional, secure, and user-friendly interface for managing prescriptions. It connects seamlessly with the Spring Boot backend, handling all CRUD operations, reporting, and analytics. The UI is fully responsive, styled with a dark gradient theme, and optimized for a smooth workflow.

2. Technology Stack

- **React (with Hooks)** – UI development and state management.
- **Vite** – fast dev server and build system.
- **React Router DOM** – client-side navigation and protected routes.
- **React-Bootstrap + Bootstrap** – responsive grid, forms, modals, and components.
- **Axios** – API requests with promise-based handling.
- **Recharts** – line charts for day-wise reports.
- **Chart.js + react-chartjs-2** – dashboards and analytical charts.
- **jsPDF + jsPDF-autotable** – PDF report generation.
- **react-to-print** – printing support.
- **dayjs** – date manipulation.
- **MUI + React Icons** – icons and additional UI components.

3. Dependencies

Installed via `npm install`:

- `react`, `react-dom`, `vite`
- `react-router-dom`
- `bootstrap`, `react-bootstrap`
- `axios`
- `recharts`
- `chart.js`, `react-chartjs-2`
- `jspdf`, `jspdf-autotable`
- `react-to-print`
- `dayjs`
- `@mui/material`, `@mui/icons-material`, `@emotion/react`, `@emotion/styled`
- `react-icons`

4. Project Structure

```
src/
├── api/
│   └── prescriptionApi.js
├── components/
│   ├── Header.jsx
│   └── Footer.jsx
├── pages/
│   ├── Login.jsx
```

```
|   ├── PrescriptionList.jsx
|   ├── CreatePrescription.jsx
|   ├── EditPrescription.jsx
|   ├── DailyPrescriptionReport.jsx
|   ├── InsightsDashboard.jsx
|   ├── PatientHistory.jsx
|   ├── PdfView.jsx
|   └── PatientHistory.css
├── styles/
├── App.jsx
└── main.jsx
```

5. Routing Map

- `/login` → Login page
- `/prescriptions` → Prescription list (dashboard)
- `/prescriptions/new` → Create prescription
- `/prescriptions/:id/edit` → Edit prescription
- `/reports/daywise` → Daily prescription report
- `/insights` → Analytics dashboard
- `/history` → Patient history (archived prescriptions)
- `/pdf` → PDF export page

6. API Endpoints Consumed

- `POST /api/v1/auth/login`

- `GET /api/v1/prescription`
- `GET /api/v1/prescription/:id`
- `POST /api/v1/prescription`
- `PUT /api/v1/prescription/:id`
- `DELETE /api/v1/prescription/:id`
- `DELETE /api/v1/prescription/by-date?start&end`
- `GET /api/v1/prescription/by-date?start&end`
- `GET /api/v1/prescription/daywise-report?start&end`
- `GET /api/v1/history`
- `POST /api/v1/history`

7. Functional Features

Authentication & Session

- **Login:**
    - Validates username/password, calls POST /auth/login.
    - Stores user in localStorage and redirects to /prescriptions.
    - Invalid credentials show clear error messages.
- **Logout:** Clears session and navigates to /login.
- **Protected routes:** Only logged-in users can access main pages.

Prescription List (Dashboard)

- Fetches all prescriptions or by date range.
- **Features:**
    - Search by patient or diagnosis.
    - Filter by date range (auto-applies).
    - Client-side pagination (10 rows/page).
    - Sorts by date (newest first).
- **Actions per row:**
    - Edit (navigates to form).
    - Delete (confirmation modal).
    - Archive (moves entry to History via POST /history).
- **UX:** Loading spinners, hover row effects, confirmation dialogs, error/success alerts.

Create Prescription

- Form with validation:
    - Mandatory: prescription date, patient name, age (0–120), gender.
    - Optional: diagnosis, medicines, next visit date.
- Client-side error feedback using isInvalid.
- Submits via POST /prescription.
- Success: shows alert, redirects to dashboard.

Edit Prescription

- Fetches prescription by ID.
- Pre-fills form, validates inputs.
- Submits via PUT /prescription/{id}.
- Shows spinner during load, alert on save.

Daily Prescription Report

- Inputs: start & end date.
- Fetches day-wise counts.
- Displays table + Recharts line chart.
- Export: PDF with styled header and filename using range.
- Handles empty state gracefully.

Insights Dashboard

- Loads all prescriptions + last 30 days' day-wise counts.
- Computes analytics:
    - Top diagnoses (comma-separated parsing).
    - Most prescribed medicines.
    - Age group distribution.
    - Gender breakdown.
    - Monthly trends (line chart).
    - Top patients by visit count.

- Charts with Chart.js (bar, doughnut, line).

Patient History

- Fetches /history.
- Features:
    - Search by name.
    - Sort (New→Old / Old→New).
    - Pagination with “Load more” button.
- Styled cards show full details; medicines listed bullet-style.

PDF Export

- From PdfView:
    - Filters by date.
    - Exports full list to PDF via jsPDF + autoTable.
- Report page also supports PDF export.
- Print-ready configuration included (react-to-print).

8. Styling & UX Enhancements

- **Theme:** Dark, animated gradient background with light text.
- **Cards:** Rounded corners, gradient borders, hover lift.
- **Tables:** Responsive, shadows, hover highlight.

- **Forms:** Dark inputs, clear validation messages, styled selects.
- **Buttons:** Gradient buttons with hover animations.
- **Alerts & Modals:** Bootstrap-styled for consistency.
- **Patient History cards:** Gradient overlays, medicines rendered cleanly.

9. Edge Cases & Improvements

- Axios vs Fetch mixed usage (should unify).
- LocalStorage used for auth instead of JWT.
- Bulk delete endpoint available but no UI button yet.

10. Running the Frontend

1. Ensure backend is running at `http://localhost:8080`.

In frontend directory:

```
npm install
npm run dev
```

2. Open `http://localhost:5173`.

For production:

```
npm run build
npm run preview
```

3.

Optional proxy (`vite.config.js`):

```
server: {
  proxy: {
    '/api': {
      target: 'http://localhost:8080',
      changeOrigin: true,
    },
  },
}
```

11. Conclusion

The frontend of the **Prescription Management System** fully supports authentication, prescription CRUD, reports, analytics, archiving, and PDF export. It is styled with a modern dark theme, provides excellent usability with validations and modals, and integrates seamlessly with the backend APIs.