Product Requirements & Specification Document

Project Name

Mediform - Healthcare Appointment Booking UI

Description

Mediform is a responsive frontend application for booking, managing, and viewing healthcare appointments. The UI emphasizes modern design, intuitive form handling, robust validation, and seamless user experience. Built with React, JavaScript, HTML, CSS, and Tailwind, Mediform targets startup and futuristic themes.

---

1. Goals & Objectives

Goal	Description
Appointment Booking	Enable users to book healthcare appointments via dynamic forms
Appointment Management	Allow users to view, edit, and cancel appointments
Responsive & Modern UI	Ensure accessibility and usability across devices
Media Integration	Support profile images and document uploads
Data Validation & Feedback	Provide real-time form validation and user feedback

---

2. Core Features

Feature	Description
Appointment Form	Dynamic form for booking appointments with validation
Appointment List	Display upcoming and past appointments with status indicators
Appointment Details	View detailed information for each appointment
Edit/Cancel Actions	Modify or cancel existing appointments
Media Upload	Upload and preview profile images or documents
Responsive Design	Optimized for mobile, tablet, and desktop
Theming	Futuristic, startup-inspired visual style using Tailwind CSS

---

3. User Stories

As a…	I want to…	So that…
Patient	Book an appointment	I can see a doctor at my convenience

Patient	View my upcoming and past appointments	I can manage my healthcare schedule
Patient	Edit or cancel an appointment	I can adjust my plans if needed
Patient	Upload documents or images	I can provide necessary information
Patient	Receive feedback on form errors	I can correct mistakes before submission

---

4. Functional Requirements

4.1 Appointment Booking

- Dynamic form fields: date, time, doctor, reason, optional notes
- Real-time validation (required fields, date/time constraints)
- Media upload (profile image, documents; preview before submit)
- Submit triggers confirmation and updates appointment list

4.2 Appointment Management

- List view: upcoming and past appointments, sortable/filterable
- Detail view: appointment info, uploaded media, status
- Edit/cancel actions with confirmation dialogs

4.3 UI/UX

- Responsive layout (mobile-first)
- Futuristic, clean design (Tailwind CSS)
- Accessible (WCAG 2.1 AA compliance)
- Loading and error states for all async actions

---

5. Non-Functional Requirements

Requirement	Specification
Performance	<1s response for UI actions
Browser Support	Latest Chrome, Firefox, Safari, Edge
Accessibility	WCAG 2.1 AA
Security	Client-side validation, input sanitization
Code Quality	ESLint, Prettier, modular React components

---

6. Technical Specifications

Area	Specification
Framework	React (functional components, hooks)
Styling	Tailwind CSS, custom themes
State Mgmt	React Context or local state

Form Handling	Controlled components, custom validation
Media Upload	File input, preview with FileReader API
Data	Mocked with local state or JSON; API-ready
Routing	React Router (if multi-page)

---

7. UI Wireframe Overview

```
[Header: Logo | User Profile]
[Main]
  [Appointment List]
    [Book Appointment Button]
    [Appointment Cards: Date, Doctor, Status, Actions]
  [Modal: Appointment Form]
    [Fields: Date, Time, Doctor, Reason, Notes, Upload]
    [Validation/Error Messages]
    [Submit/Cancel Buttons]
[Footer: Links, Copyright]
```

---

8. Acceptance Criteria

ID	Criteria
AC1	Users can book, view, edit, and cancel appointments
AC2	Forms validate input and display errors in real-time
AC3	Media uploads are previewed and attached to appointments
AC4	UI is responsive and visually consistent across devices
AC5	All user actions provide feedback (loading, success, error)

---

9. Out of Scope

- Backend integration (API endpoints, authentication)
- Payment processing
- Doctor/admin dashboards

---

10. Milestones & Timeline

Milestone	Target Date
UI Design & Wireframes	Week 1
Core Components	Week 2
Form Handling & Validation	Week 3

Media Integration	Week 4
Responsive Testing	Week 5
Final QA & Handover	Week 6

---

11. Risks & Mitigations

Risk	Mitigation
Complex validation logic	Use modular validation functions
Media upload compatibility	Test across browsers, limit file types
UI consistency	Strict use of Tailwind and design tokens

---

12. Appendix

- **Design References:** Startup, futuristic UI inspiration
- **Accessibility Checklist:** WCAG 2.1 AA
- **Sample Data:** Provided as JSON for development

End of Document