Patient Care Tracker - Requirements Document

Project Overview

A web-based application to help patients manage their daily healthcare routines, including medication reminders, appointment tracking, and exercise logging. The system should be built using Replit with no additional subscription services required.

---

1. Functional Requirements

1.1 Medication Management

- **Daily medication reminder system**
  - Display list of medications with scheduled times
  - Visual indicators for taken/missed medications
  - Mark medications as taken with timestamp
  - Support multiple medications per day
  - Allow adding/editing/deleting medications
  - Each medication should include: name, dosage, frequency, time(s)

1.2 Appointment Tracking

- **Appointment calendar system**
  - View upcoming appointments
  - Add new appointments (date, time, doctor/facility, purpose, notes)
  - Edit or cancel existing appointments
  - Visual calendar view (daily/weekly/monthly)
  - Mark appointments as completed
  - Past appointment history

1.3 Exercise Routine Tracking

- **Exercise logging system**
  - Create exercise routines with multiple exercises
  - Log completed exercises with date/time

- Track exercise details: name, duration, sets/reps, notes
- View exercise history
- Mark exercises as complete
- Pre-defined exercise templates (walking, stretching, strength training, etc.)

1.4 Dashboard & Notifications

- **Central dashboard**
  - Today's overview: medications due, appointments scheduled, exercises planned
  - Quick action buttons for common tasks
  - Progress indicators (medications taken today, exercises completed)
  - Visual status indicators (color-coded alerts)

---

2. Technical Requirements

2.1 Technology Stack

- **Frontend**: HTML5, CSS3, JavaScript (Vanilla JS or React)
- **Backend**: Node.js or Python (Flask/FastAPI)
- **Database**: SQLite (file-based, no external service needed)
- **Hosting**: Replit (built-in hosting)
- **No external APIs or subscription services**

2.2 Data Storage

- **Local database (SQLite) with tables for:**
  - Medications (id, name, dosage, frequency, times, active)
  - Medication logs (id, medication_id, timestamp, taken)
  - Appointments (id, date, time, doctor, purpose, notes, completed)
  - Exercises (id, name, description, duration, sets, reps)
  - Exercise logs (id, exercise_id, timestamp, completed, notes)

2.3 Browser Compatibility

- Support modern browsers: Chrome, Firefox, Safari, Edge

- Responsive design for desktop, tablet, and mobile devices

---

3. User Interface Requirements

3.1 Navigation

- Simple, intuitive menu structure
- Main sections: Dashboard, Medications, Appointments, Exercises
- Consistent navigation across all pages
- Mobile-friendly hamburger menu

3.2 Visual Design

- Clean, accessible design with good contrast
- Large, easy-to-tap buttons for mobile users
- Color coding:
  - Green: completed/taken
  - Yellow: upcoming/pending
  - Red: missed/overdue
- Clear typography with readable font sizes
- Calm, healthcare-appropriate color palette

3.3 User Experience

- Minimal clicks to complete common tasks
- Confirmation dialogs for destructive actions
- Success/error messages for user actions
- Loading indicators for data operations
- Intuitive forms with validation

---

4. Data Management

4.1 Data Persistence

- All data stored locally in SQLite database

- All data stored locally in SQLite database
- No cloud storage or external services
- Data persists across browser sessions
- Simple backup/export functionality (JSON/CSV download)

4.2 Data Privacy

- All data stored locally on Replit instance
- No third-party data sharing
- Optional password protection for app access

---

5. Core Features Priority

Phase 1 (MVP - Minimum Viable Product)

1. Basic dashboard
2. Add/view/mark medications as taken
3. Add/view appointments
4. Add/log exercises
5. Simple SQLite database setup

Phase 2 (Enhancements)

1. Calendar view for appointments
2. Exercise templates and routines
3. Statistics and progress tracking
4. Data export functionality
5. Improved UI/UX with animations

Phase 3 (Advanced Features)

1. Medication streak tracking
2. Exercise progress charts
3. Appointment reminders (browser notifications)
4. Custom themes

5. Print-friendly reports

---

6. Technical Specifications for AI Agent

6.1 File Structure

```
/patient-care-tracker
├── index.html
├── style.css
├── script.js
├── server.py (or server.js)
├── database.db (SQLite - auto-generated)
├── /assets
│   └── (icons, images)
└── README.md
```

6.2 Database Schema

Medications Table

- id (INTEGER PRIMARY KEY)
- name (TEXT)
- dosage (TEXT)
- frequency (TEXT)
- time_slots (TEXT - JSON array)
- created_at (DATETIME)
- active (BOOLEAN)

Medication_Logs Table

- id (INTEGER PRIMARY KEY)
- medication_id (INTEGER FOREIGN KEY)
- taken_at (DATETIME)
- scheduled_time (TEXT)
- status (TEXT: taken/missed/skipped)

Appointments Table

- id (INTEGER PRIMARY KEY)
- date (DATE)
- time (TIME)
- doctor_name (TEXT)
- facility (TEXT)
- purpose (TEXT)
- notes (TEXT)
- completed (BOOLEAN)
- created_at (DATETIME)

Exercises Table

- id (INTEGER PRIMARY KEY)
- name (TEXT)
- description (TEXT)
- duration_minutes (INTEGER)
- sets (INTEGER)
- reps (INTEGER)
- created_at (DATETIME)

Exercise_Logs Table

- id (INTEGER PRIMARY KEY)
- exercise_id (INTEGER FOREIGN KEY)
- completed_at (DATETIME)
- duration_actual (INTEGER)
- notes (TEXT)

6.3 API Endpoints (if using backend)

Medications

- GET /api/medications - Get all medications
- POST /api/medications - Add new medication

- PUT /api/medications/:id - Update medication
- DELETE /api/medications/:id - Delete medication
- POST /api/medications/:id/log - Log medication taken

Appointments

- GET /api/appointments - Get all appointments
- GET /api/appointments/upcoming - Get upcoming appointments
- POST /api/appointments - Add new appointment
- PUT /api/appointments/:id - Update appointment
- DELETE /api/appointments/:id - Delete appointment

Exercises

- GET /api/exercises - Get all exercises
- POST /api/exercises - Add new exercise
- POST /api/exercises/:id/log - Log exercise completed
- GET /api/exercises/logs - Get exercise history

---

7. Success Criteria

7.1 Functionality

- ✓ User can add and track medications
- ✓ User can view and mark medications as taken
- ✓ User can add and view appointments
- ✓ User can log exercises
- ✓ Data persists across sessions
- ✓ Responsive design works on mobile devices

7.2 Performance

- Page load time < 2 seconds
- Smooth interactions with no lag

- Database queries execute quickly (< 100ms)

7.3 Usability

- Intuitive navigation requiring no instructions
- Accessible design meeting WCAG 2.1 AA standards
- Works offline (after initial load)

---

8. Constraints & Limitations

8.1 Platform Constraints

- Must work within Replit's free tier
- No external paid services or APIs
- File-based database only (SQLite)
- Limited to Replit's built-in hosting

8.2 Technical Limitations

- No real-time SMS/email notifications (browser notifications only)
- No multi-user support (single patient use)
- No mobile app (web-based only)
- No cloud sync across devices

---

9. Future Enhancements (Post-MVP)

- Family/caregiver access with permission system
- PDF report generation for doctor visits
- Integration with health devices (if APIs become free)
- Medication interaction checker (using free APIs)
- Voice reminders using Web Speech API
- Progressive Web App (PWA) for offline use
- Multi-language support

10. Instructions for AI Agent

Build this application with:

1. Clean, commented code
2. Semantic HTML5
3. Modern CSS with Flexbox/Grid
4. Vanilla JavaScript or React (lightweight)
5. SQLite for database
6. Python Flask or Node.js Express for backend
7. RESTful API structure
8. Error handling and validation
9. Responsive design (mobile-first)
10. Accessibility features (ARIA labels, keyboard navigation)

Do NOT use:

- External APIs requiring keys or payment
- Cloud databases (MongoDB Atlas, Firebase, etc.)
- Authentication services (Auth0, etc.)
- Payment processing
- Any subscription services

Deliverables:

- Fully functional web application
- README with setup instructions
- Commented code for easy understanding
- Sample data for testing
- Basic documentation

Contact & Support

For questions or modifications to requirements, please provide feedback through the development process.