Project Deep Dive: How It Works

This system is designed to be completely **serverless and free**, meaning it doesn't require a dedicated computer at the clinic or any hosting fees. It cleverly uses free services from Google and GitHub to create a robust patient management workflow.

The Patient's Journey 🚶

1. **Arrival & Scan:** A patient arrives at the clinic and sees a printed QR code at the reception. They scan this code with their smartphone.
2. **Instant Form:** The QR code opens a simple, mobile-friendly webpage on their phone's browser. This webpage is a registration form.
3. **Data Entry:** The patient fills in their details (e.g., Name, Phone Number, Age). Since they are typing their own information, it reduces the chance of spelling errors.
4. **Submission & Token:** After clicking "Submit," their information is sent securely over the internet. In just a few seconds, their screen updates to show them a confirmation message with their permanent **Patient ID** and their **Token Number** for that day's queue.

The Staff's Experience 👩‍⚕️

1. **Live Dashboard:** The clinic staff has a **Google Sheet** open on a computer, tablet, or even a phone. This sheet acts as their live dashboard.
2. **Real-Time Updates:** The moment a patient submits their form, a new row with all their details (Name, Phone, Token Number, etc.) instantly appears at the bottom of the Google Sheet.
3. **Manage the Queue:** The staff can see the entire patient queue in a clean, organized, and time-stamped list. They know exactly who is next without any manual effort.

Behind the Scenes: The Technology ⚙️

The magic of the system lies in how three free services work together:

- **GitHub Pages (The Form):** The simple HTML registration form is stored and hosted for free on GitHub Pages. This is what the QR code links to.
- **Google Sheets (The Database):** This is where all the patient and appointment data is securely stored. It's the system's "brain" and the staff's dashboard.
- **Google Apps Script (The Connection):** A small piece of code running on Google's servers acts as the bridge. When a patient submits the form, the data is sent to this script. The script then intelligently processes the data (calculates the token number, checks if it's a new patient) and writes it neatly into the Google Sheet.

Detailed Project Workplan

Here is a comprehensive plan broken down into phases and tasks.

Project Overview

Item	Detail
Project Name	Serverless Clinic Patient Management System
Objective	To create a zero-cost, automated patient registration and queuing system using serverless technologies.
Key Technologies	Google Sheets, Google Apps Script, GitHub Pages, HTML/CSS/JavaScript
Total Estimated Time	**16 - 20 Hours** (Approx. 2-3 workdays for a developer)

🗓️ Phase 1: Foundation & Setup (Estimated Time: 2 Hours)

This phase involves preparing all the necessary accounts and files.

Task	Description	Estimated Time	Status
1.1: Create Accounts	Ensure access to a Google Account (for Sheets/Script) and a GitHub Account.	30 mins	To-Do
1.2: Setup Google Sheet	Create a new Google Sheet. Define and freeze the header row with the required columns (Timestamp, PatientID, Name, Phone, Age, TokenNumber, Date).	30 mins	To-Do
1.3: Setup GitHub Repo	Create a new, public GitHub repository to host the frontend code.	30 mins	To-Do
1.4: Initial File Creation	Create the basic index.html file in the repository. Set up the basic HTML structure and link to Tailwind CSS via CDN.	30 mins	To-Do

⚙️ Phase 2: Backend Development - The "Engine" (Estimated Time: 6 Hours)

This phase focuses on writing the core logic in Google Apps Script.

Task	Description	Estimated Time	Status
2.1: Create Apps Script	Open Apps Script from the Google Sheet. Write the main doPost(e) function to receive web requests.	1 Hour	To-Do
2.2: Implement Data Parsing	Write the code to parse the incoming JSON data from the form submission.	1 Hour	To-Do
2.3: Patient ID Logic	Implement the function (getPatientId) to search the sheet by phone number. If the patient exists, return their ID; if not, generate a new one.	1.5 Hours	To-Do

Task	Description	Estimated Time	Status
2.4: Token Number Logic	Implement the function (getNextToken) to count today's entries and calculate the next token number.	1.5 Hours	To-Do
2.5: Data Writing Logic	Write the code to append the final, processed row of data into the Google Sheet.	30 mins	To-Do
2.6: Initial Deployment	Deploy the script as a Web App with "Anyone" access to get the initial script URL for testing.	30 mins	To-Do

🎨 **Phase 3: Frontend Development - The "Face" (Estimated Time: 5 Hours)**

This phase focuses on building the patient-facing registration form.

Task	Description	Estimated Time	Status
3.1: Build Registration Form	Design the HTML form with fields for Name, Phone, and Age. Style it using Tailwind CSS to be clean and responsive.	2 Hours	To-Do
3.2: Build Success View	Design the confirmation screen that will be shown after successful submission. This will display the Patient ID and Token Number. Initially, keep it hidden.	1 Hour	To-Do
3.3: Write JavaScript	Write the client-side JavaScript to handle the form submission. This includes capturing data, sending it to the Apps Script URL via fetch, and handling the response.	2 Hours	To-Do

🧪 **Phase 4: Integration & Testing (Estimated Time: 3 Hours)**

This phase connects the frontend and backend and ensures everything works perfectly.

Task	Description	Estimated Time	Status
4.1: Connect Frontend to Backend	Add the final Google Apps Script URL to the JavaScript fetch call in the index.html file.	30 mins	To-Do
4.2: End-to-End Testing	Test the complete workflow: fill the form, submit, check if the data appears correctly in the Google Sheet, and verify the success message on the webpage.	1.5 Hours	To-Do

Task	Description	Estimated Time	Status
4.3: Error & Edge Case Testing	Test for errors like submitting an empty form, submitting with invalid data, or simulating a slow network. Ensure the user sees appropriate feedback.	1 Hour	To-Do

🚀 **Phase 5: Deployment & Handover (Estimated Time: 2 Hours)**

This phase involves finalizing the project and making it ready for the clinic.

Task	Description	Estimated Time	Status
5.1: Finalize GitHub Pages	Push the final, tested index.html to GitHub. Enable GitHub Pages in the repository settings and get the live public URL.	30 mins	To-Do
5.2: Generate QR Code	Use the live GitHub Pages URL to generate a high-quality QR code image. Save it for printing.	30 mins	To-Do
5.3: Create Documentation	Write a simple "How-To" guide for the clinic staff. It should explain how to view the Google Sheet and what the columns mean.	1 Hour	To-Do