

Clash-A-Thon 2026

The Friction We Forget

Building Solutions for Problems We've Stopped Noticing

Kick-off Feb 24, 2026 09:30 AM	Submission Deadline Feb 26, 2026 4:00 PM	Final Presentation Feb 27, 2026 On-site
-------------------------------------	---	--

TEAM & PROJECT INFORMATION

Team Name	Sanjeevani
Project Title	Sanjeevani
Project Category	HealthTech

TEAM MEMBERS

#	Full Name	Role / Responsibility	GitHub / Contact
1	Shovan Bhattarai	Full Stack and Web Lead	Shovaan83
2	Pranisha Karki	Full Stack and Web Lead	Pranishaaa
3	Sudan Khadka	Full Stack and Backend Lead	sudankdk
4	Abhishek Sharma	Full Stack and Mobile Lead	4bhisheksharma
5	Raghav Dahal	Full stack and Backend Lead	Raghav977

SUBMISSION LINKS

GitHub Repository	https://github.com/Shovaan83/ClashAThon-TeamSanjeevani-Sanjeevani.git
Deployment URL	https://clash-a-thon-team-sanjeevani-sanjee-ten.vercel.app/home http://157.245.107.182:8000/
Tech Stack	React, Flutter, Django, SQLite

Declaration

We declare that this project is our original work developed during Clash-A-Thon 2026. All external resources and libraries have been appropriately credited. We understand that plagiarism or misrepresentation will result in disqualification.

Table of Contents

1	Technical Documentation.....	1
1.1	System Architecture.....	1
1.2	Component Breakdown	2
1.2.1.	Major Modules and Services	2
1.3	Data Design	3
1.4	API and Interface Specification.....	4
1.5	Setup and Deployment	7
2	Business Documentation	9
2.1	Business Model Canvas	9
2.2	Unique Selling Proposition.....	9
2.3	Market Analysis.....	10
2.4	Sustainability and Future Scope	11

Table of Figures

Figure 1: System Architecture 1

Figure 2 Entity Relationship Diagram.....3

Figure 3 Business Model Canvas.....9

Table of Tables

Table 1: Comparison over competitors 9

Table 2: Market Analysis 10

1 Technical Documentation

1.1 System Architecture

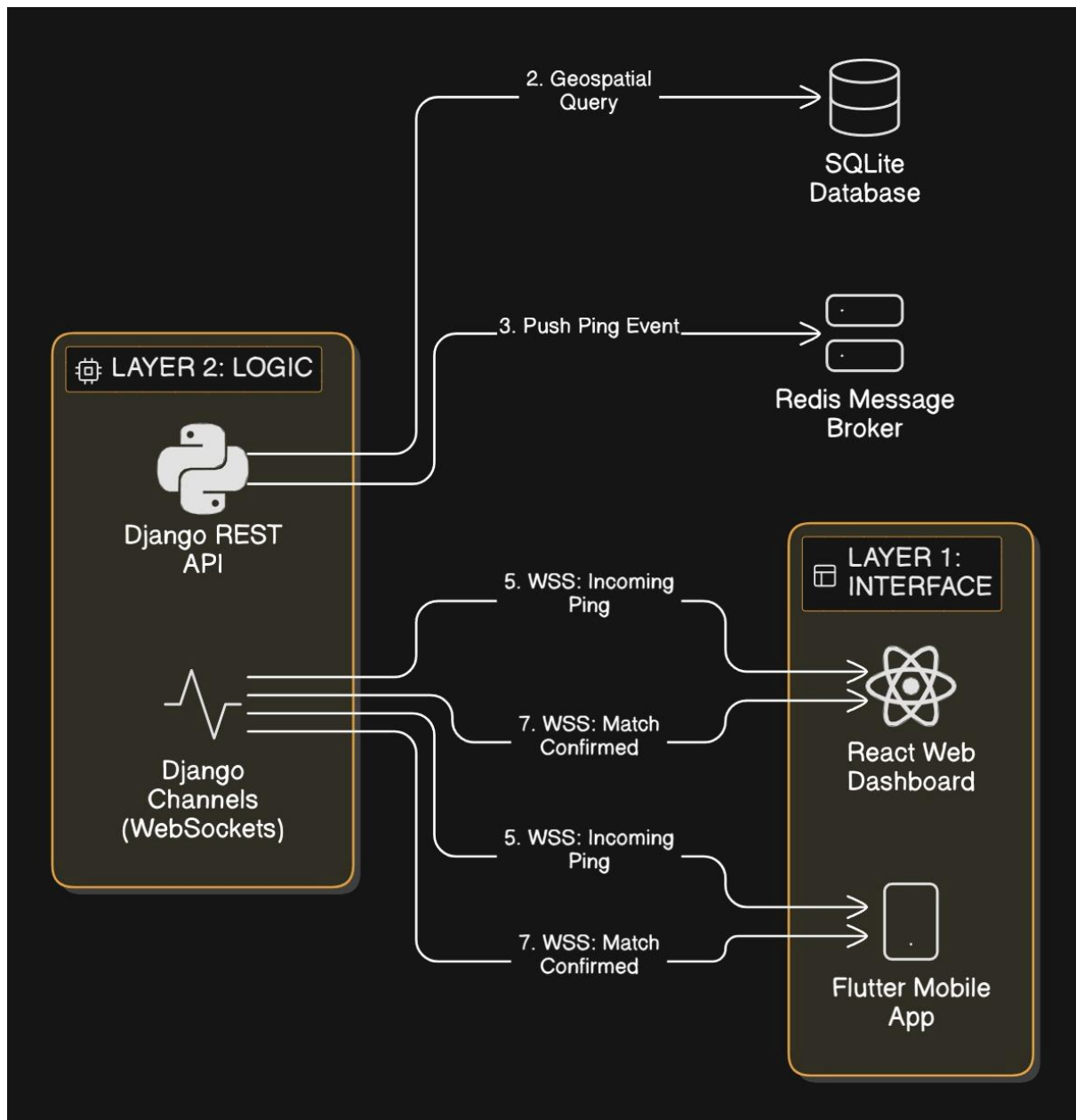


Figure 1: System Architecture

1.2 Component Breakdown

1.2.1. Major Modules and Services

i. Authentication Module

The Authentication Module deals with user and pharmacy registration, user login, role management and session validation. It guarantees protection of access controls within the system.

ii. Broadcast Engine

Broadcast Engine is charged with the duty of accepting prescription uploads and geofenced notifications. It sorts out pharmacies in the radius of choice of the user and provides real-time alerting via WebSocket connections.

iii. Real-Time Notification Service

The Real-Time Notification Service is used to make sure that the pharmacy devices are notified as soon as a broadcast request is created. It has constant socket connections to facilitate instant communication.

iv. Reservation Manager (The digital Handshake)

Manages the temporary reservation lifecycle. When a patient selects a confirmed pharmacy and clicks "I'm Coming," this module locks the transaction state and initiates a synchronized 10-minute countdown timer across both the patient's mobile app and the pharmacy's web dashboard, preventing the medicine from being sold to walk-in customers.

v. Reliability Scoring Engine

Tracks pharmacy responsiveness to optimize the network. It calculates response times and misses requests to generate the "FOMO Ledger" (Fear Of Missing Out), showing pharmacies exactly how much revenue, they lost by ignoring pings. It also awards "Fast Responder" badges to highly active pharmacies.

vi. Omni-Channel Identity Guard

Handles secure, role-based routing using JWT. If registered pharmacy has not upload the PAN and license documents, then this module blocks from connecting to the WebSocket broadcast network.

1.3 Data Design

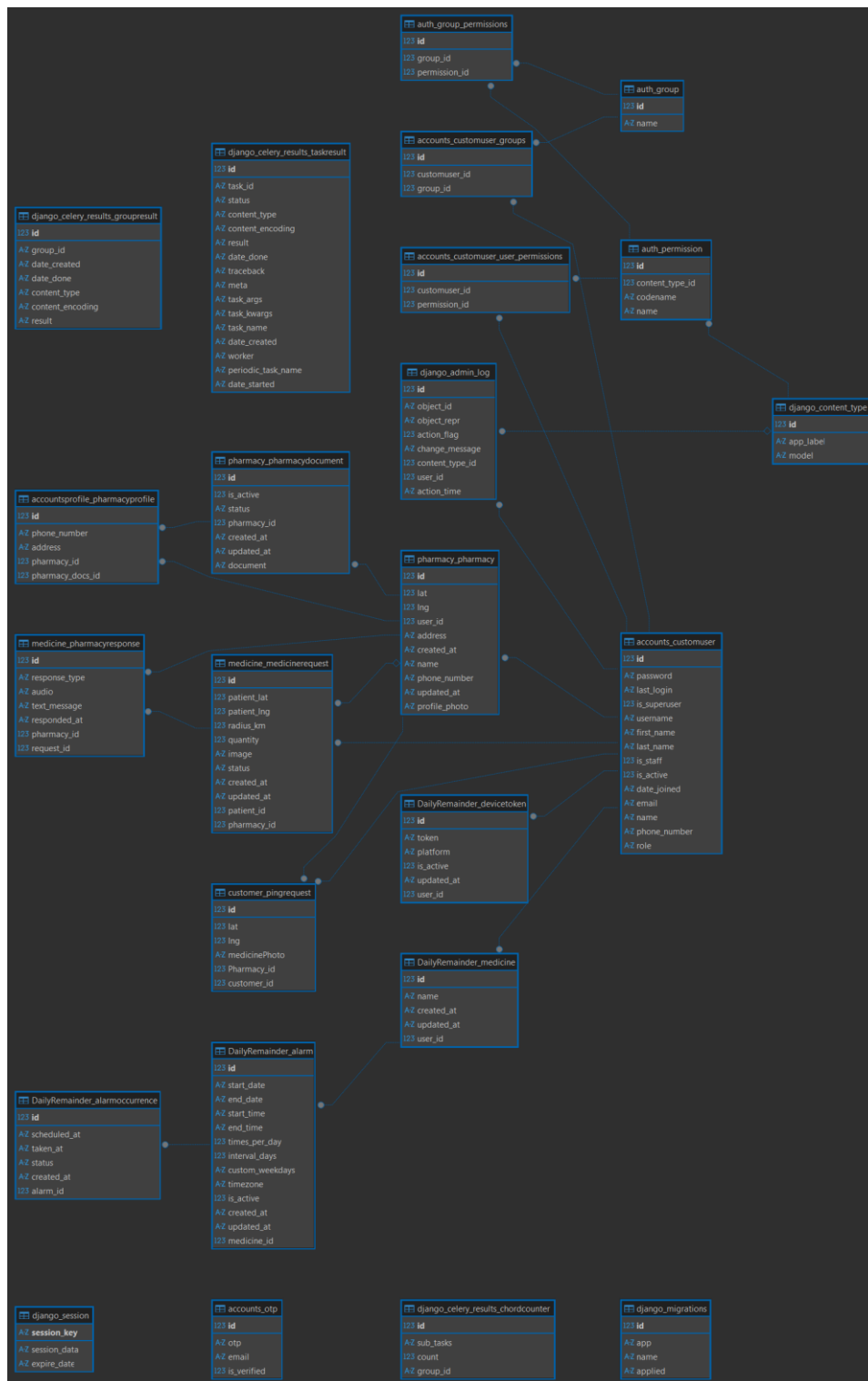


Figure 2 Entity Relationship Diagram

1.4 API and Interface Specification

All APIs use JSON format for request and response bodies. Most endpoints require authentication using an access token.

Authentication Module

Purpose:

Handles user login, logout, and OTP-based verification.

Endpoints:

POST /login

Description: Authenticates user credentials and returns access token.

POST /logout

Description: Logs out the authenticated user.

POST /send-otp

Description: Sends an OTP to the registered contact for verification.

POST /verify-otp

Description: Verifies the OTP provided by the user.

Customer Module

Purpose:

Manages customer registration, profile, and medicine requests.

Endpoints:

POST /customer/register/

Description: Registers a new customer.

GET /customer/profile/

Description: Retrieves the authenticated customer profile.

PUT /customer/profile/

Description: Updates the authenticated customer profile.

GET /customer/requests/

Description: Retrieves all medicine requests created by the customer.

Pharmacy Module

Purpose:

Handles pharmacy registration, profile management, and document uploads.

Endpoints:

GET /register-pharmacy/
POST /register-pharmacy/
GET /register-pharmacy/{id}/
PUT /register-pharmacy/{id}/
PATCH /register-pharmacy/{id}/
DELETE /register-pharmacy/{id}/
Description: Manages pharmacy registration records.

GET /pharmacy/profile/
POST /pharmacy/profile/
PUT /pharmacy/profile/
PATCH /pharmacy/profile/
Description: Manages pharmacy profile information.

POST /pharmacy/document/upload/
Description: Uploads verification documents for pharmacy.

POST /pharmacy/profile-photo/upload/
Description: Uploads pharmacy profile photo.

DELETE /pharmacy/profile-photo/upload/
Description: Deletes pharmacy profile photo.

Medicine Request and Response Module

Purpose:
Implements the broadcasting system between customers and pharmacies.

Endpoints:

GET /medicine/request/
Description: Retrieves all medicine requests.

POST /medicine/request/
Description: Creates a new medicine request by customer.

GET /medicine/response/
Description: Retrieves responses from pharmacies.

POST /medicine/response/
Description: Allows pharmacy to respond to a medicine request.

POST /medicine/select/
Description: Allows customer to select a pharmacy offer.

Daily Reminder Module

Purpose:
Manages medicine scheduling, alarms, reminder occurrences, and notifications.

Alarms:

GET /api/daily-reminder/alarms/

Description: Retrieves all alarms for authenticated user.

POST /api/daily-reminder/alarms/

Description: Creates a new alarm and generates occurrences for a defined date range.

GET /api/daily-reminder/alarms/{id}/

Description: Retrieves a specific alarm.

PUT /api/daily-reminder/alarms/{id}/

Description: Updates a specific alarm.

DELETE /api/daily-reminder/alarms/{id}/

Description: Deletes a specific alarm.

Medicines:

GET /api/daily-reminder/medicines/

Description: Retrieves all medicines for reminder module.

POST /api/daily-reminder/medicines/

Description: Creates a new medicine entry.

GET /api/daily-reminder/medicines/{id}/

Description: Retrieves a specific medicine.

PUT /api/daily-reminder/medicines/{id}/

Description: Updates a specific medicine.

DELETE /api/daily-reminder/medicines/{id}/

Description: Deletes a specific medicine.

Occurrences:

GET /api/daily-reminder/occurrences/

Description: Retrieves reminder occurrences.

PATCH /api/daily-reminder/occurrences/{id}/

Description: Partially updates an occurrence (e.g., mark as taken or missed).

Dashboard:

GET /api/daily-reminder/dashboard/

Description: Returns summary data such as upcoming reminders and missed doses.

Device Tokens:

POST /api/daily-reminder/device-tokens/

Description: Registers device token for push notifications.

DELETE /api/daily-reminder/device-tokens/{id}/

Description: Deletes device token.

Sync Notifications:

POST /api/daily-reminder/sync-notifications/

Description: Triggers immediate notification check for authenticated user.

Profile Management Module

POST /api/profilecreate/

Description: Creates a user profile.

PUT /api/profileprofile/{id}/update/

Description: Updates full profile information.

PATCH /api/profileprofile/{id}/update/

Description: Partially updates profile information.

1.5 Setup and Deployment

Local setup command:

Backend:

```
cd backend
```

```
python -m venv env
```

```
source env/bin/activate # On Windows: env\Scripts\activate
```

```
pip install -r requirements.txt
```

```
python manage.py migrate
```

```
python manage.py runserver
```

Web Frontend:

```
cd web
```

```
npm install
```

```
npm run dev
```

Mobile App:

cd mobile

flutter pub get

flutter run

Deployment Process:

The backend is deployed from digital ocean, and the frontend is deployed form vercel.

Required Environment Variable:

SECRET_KEY=your_django_secret_key_here

DEBUG=True

REDIS_URL=redis://localhost:6379

ALLOWED_HOSTS=localhost,127.0.0.1

VITE_API_BASE_URL=http://localhost:8000

VITE_WS_BASE_URL=ws://localhost:8000

VITE_FIREBASE_API_KEY=your_firebase_api_key

VITE_FIREBASE_AUTH_DOMAIN=your_project.firebaseio.com

VITE_FIREBASE_PROJECT_ID=your_project_id

2 Business Documentation

2.1 Business Model Canvas

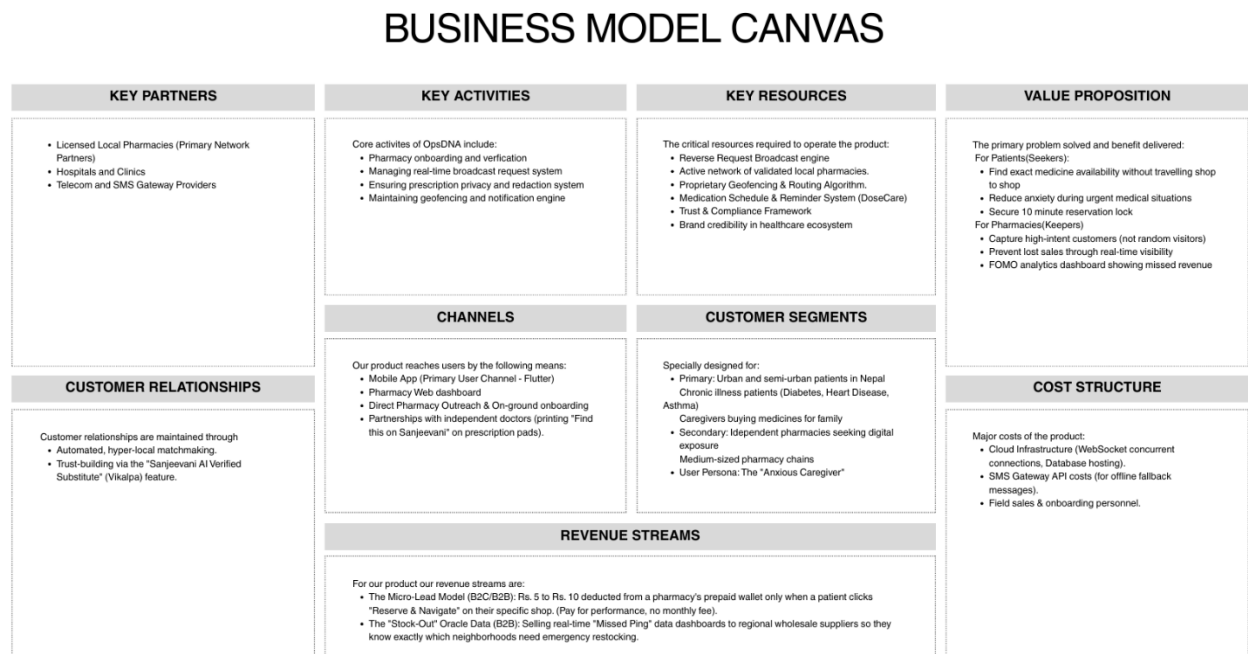


Figure 3 Business Model Canvas

2.2 Unique Selling Proposition

Sanjeevani is Nepal's first real-time medicine demand routing platform, connecting verified patients with licensed pharmacies through a lightweight reverse-broadcast system-no digital inventory required. By enabling pharmacies to confirm stock with a single voice response ("Cha!"), it eliminates manual data entry friction, delivers confirmed availability with temporary reservation locks to patients, and provides pharmacies access to high-intent demand without complex tech investment.

Table 1: Comparison over competitors

Feature	Sanjeevani	Nepali Health Apps (e.g. NepMeds/ Jeevee)	Global Similar Platforms (e.g. Ajid/ MediConnect)

Real-time multi pharmacy stock availability	✓	✗	✓
Requires digital stock uploads	✗	✓	Some require stock feeds
Reverse broadcast to pharmacies	✓	✗	✓
Digital reservation guarantee	✓	✗	Partial/ not standard
Medication adherence reminders	✓	✗	✗

2.3 Market Analysis

Table 2: Market Analysis

Element	What to include
Target Market	Our initial target market comprises patients leaving major medical hubs (e.g., local polyclinics in Itahari, BPKIHS in Dharan) who are handed prescriptions for specific branded medications or post-surgery supplies.
Market Size	<p>TAM (Total Addressable Market): Over 20,000+ registered retail pharmacies in Nepal processing millions of transactions daily.</p> <p>SOM (Serviceable Obtainable Market for Year 1): 500+ pharmacies in the Koshi Province corridor (Itahari-Dharan-Biratnagar).</p>
Competition	Centralized e-pharmacies (like Jeevee or ePharmacy) are the main alternatives. However, their weakness is time. They act as retail competitors to local shops. Sanjeevani acts as an enabler for local shops, turning competition into partnership.
Market Opportunity	The rapid penetration of mobile internet and e-wallets (eSewa/Khalti) in Nepal, local shopkeepers are finally digitally

	native. However, no software exists tailored to their "dirty-hands, fast-paced" workflow.
Go-to-Market Strategy	Instead of marketing citywide, we will geofence a 1 km radius around a major hospital in Itahari, physically onboard the 15 pharmacies within that zone for free and provide each with Rs. 500 in wallet credits to seed supply. A banner at the hospital gate reading "Medicine out of stock? Scan here to find it locally." will drive immediate, high-intent patients directly to this pre-seeded pharmacy network.

2.4 Sustainability and Future Scope

Financial Sustainability: The platform covers its own server costs early via the "Micro-Lead" model (pre-paid wallets). Because pharmacies only pay when a customer walks out their door to visit them, churn will be near-zero. True profitability will scale through the B2B Data Model, monetizing aggregated "unmet demand" analytics to wholesale distributors who currently rely on outdated, manual surveying.

Chyatteko Pata (The Cut-Strip Ledger): A feature allowing patients to digitize and track the expiry dates of partial foil strips they buy-a massive, unrecognized safety hazard in Nepal.

Bhaichara (B2B Networking): Allowing a pharmacy to forward a patient's ping to a neighboring shop if they are out of stock, splitting the micro-lead fee.

6-Month Vision: Dominate the Itahari-Dharan medical corridor, reaching 200 daily active broadcasts.

1-Year Vision: Expand into the Kathmandu Valley and integrate directly with wholesale supply chain ERP systems to auto-order medicines based on Sanjeevani's missed-ping analytics.

Partnerships: Partnering with the Nepal Chemists and Druggists Association (NCDA) to validate our network and establish trust