

# **CHILDREN VACCINATION MANGEMENT SYSTEM**

## **A PROJECT REPORT**

*Submitted by*

**RANJANI P                      731122205041**

*in partial fulfilment for the award of the degree*

*of*

**BACHELOR OF TECHNOLOGY**

*in*

**INFORMATION TECHNOLOGY**

**GOVERNMENT COLLEGE OF ENGINEERING**

**ERODE**

**ANNA UNIVERSITY : CHENNAI 600 025**

**2025**

## **BONAFIDE CERTIFICATE**

Certified that this project report “**CHILD VACCINATION**” is the bonafide work of **RANJANI P 731122205041** who carried out the project work under my supervision.

**SIGNATURE OF HOD**

**Dr I BHUVANESHWARI, M.E, Ph.D**

HEAD OF THE DEPARTMENT

Department of IT

Government College of Engineering,

Erode – 638 316

**SIGNATURE OF SUPERVISOR**

**Dr M SATHYAKALA, M.Tech, Ph.D**

ASSISTANT PROFESSOR

Department of IT

Government College of Engineering

Erode – 638 316

Submitted to the university examination held on \_\_\_\_\_ at Government college of Engineering,  
Erode.

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	2
1	INTRODUCTION	3
	1.1 PROJECT DEFINITION	3
	1.2 OVERVIEW OF MODEL	4
	1.3 ALGORITHM	5
	1.4 FLOWCHART	6
2	LITERATURE REVIEW	7
	2.1 PREVIOUS WORK / RELATED STUDIES	7
3	METHODOLOGY / SYSTEM DESIGN (WITH DEVOPS)	8
	3.1 SYSTEM ARCHITECTURE	9
	3.2 FUNCTION MODULES	10
4	IMPLEMENTATION	12
5	JENKINS IN CHILD VACCINATION TRACKING SYSTEM	14
6	TESTING - PYTEST	16
7	DEPLOYMENT- RENDER	18
8	OUTPUT AND SCREENSHOTS	19
9	CONCLUSION AND FUTURE SCOPE	20
10	REFERENCES	21

## **ABSTRACT**

The Children Vaccination Management System is a web-based application developed using Python Flask, HTML, CSS, and an SQLite database to manage and automate child vaccination records. The system allows parents and healthcare providers to register children, store personal and immunization details, and generate vaccination schedules automatically based on the child's date of birth. Upcoming vaccines, completed doses, and missed vaccinations are displayed with clear status updates, ensuring that no dose is forgotten. By maintaining an organized medical database, the system supports better decision-making and efficient monitoring of vaccination coverage.

The system improves communication between healthcare authorities and parents by offering timely reminders before each vaccination due date. It also allows users to modify and view vaccination history, generate reports, and ensure secure data handling through authenticated access. The user-friendly interface makes the system easily accessible to all types of users, even those with limited technical knowledge. Overall, this project aims to enhance the immunization process through automation, reduce data loss, and improve public health outcomes by ensuring timely vaccination of every child.

Therefore, the Children Vaccination Management System significantly contributes to a healthier society by reducing the risks of preventable diseases and strengthening the healthcare infrastructure with modern digital technology.

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 PROJECT DEFINITION**

The Children Vaccination Management System is a web-based application designed to digitally manage and automate child vaccination records. The main objective of this project is to ensure that every child receives vaccinations on time by tracking their immunization schedule and providing timely reminders. The system enables parents and healthcare workers to register children, store personal and vaccination details, display upcoming and completed vaccines, and download vaccination records when needed. The platform replaces traditional paper-based documentation with an efficient digital solution that improves accuracy, accessibility, and reliability of vaccination data.

The application automatically generates a schedule for each child based on the standard immunization chart recommended by healthcare authorities. It tracks the status of vaccines such as due, completed, or missed. Using a secure database, all child records are stored safely and can be updated by authorized users. The system also helps healthcare organizations monitor vaccination coverage within a community or region.

This system is designed with a user-friendly interface to make the management process simpler for users. It ensures that no child misses a vaccine dose due to forgetfulness or lack of awareness. By integrating technology into healthcare delivery, this project helps strengthen the public health system and contributes to the prevention of many vaccine-preventable diseases.

Thus, the system aims to provide a reliable digital platform that enhances efficiency, increases parental awareness, and ultimately ensures the well-being and protection of children through timely vaccination.

## 1.2 OVERVIEW OF MODEL

The Children Vaccination Management System follows a client–server model, where the frontend interacts directly with the backend server to fetch, store, and update information. The project is developed using Python Flask as the backend framework, HTML and CSS for the user interface, and SQLite as the database for securely storing all records. The system architecture ensures smooth communication between users and the server, providing fast and reliable access to vaccination information.

The model consists of major functional components such as User Registration, Login Authentication, Child Profile Management, Vaccination Schedule Management, Report Generation, and Notifications. Parents or healthcare staff can register a child by entering essential details such as name, date of birth, and contact information. Based on the child’s age, the system automatically generates a personalized vaccination timetable showing vaccine names, recommended dates, and completion status.

The system follows a modular design approach where each component works independently but contributes to the overall workflow. Data security and user authentication mechanisms are applied to ensure that only authorized users can view or modify child records. The user-friendly interface ensures ease of access from any location, supporting both hospital-based and home-based usage.

This model not only enhances efficiency in medical data handling but also minimizes human errors and information delays. The system supports scalability, which allows future improvements such as SMS/email reminders, government health department integration, and mobile application expansion.

## 1.3 ALGORITHM

The primary algorithm used in this project focuses on **age-based vaccination scheduling** and **status tracking**. The system calculates the age of the child using the Date of Birth entered during registration. Based on this age, the algorithm maps the child to the appropriate vaccination doses as per the national immunization schedule. Each vaccine has a predefined recommended age range, and the algorithm checks the current date against the expected due date to update the vaccination status.

### Algorithm Steps

#### Input

- Child Name
- Date of Birth
- Parent/Guardian Information

#### Process

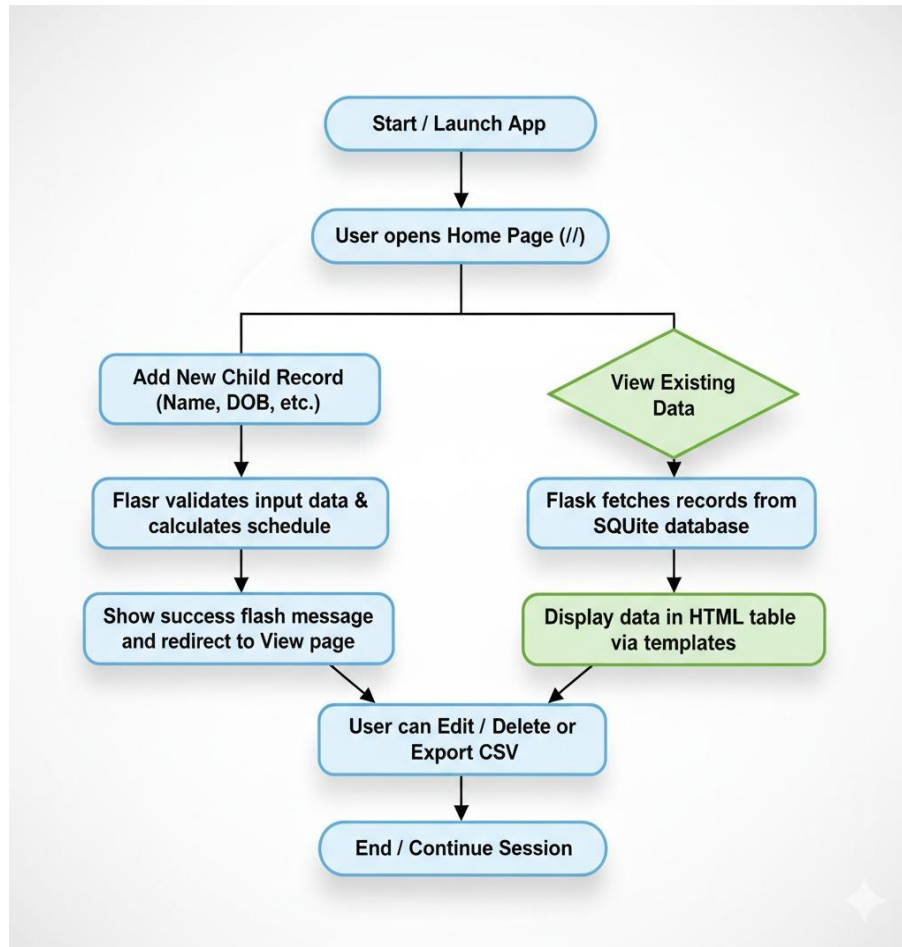
- Calculate current age of the child
- Retrieve corresponding vaccination schedule
- If due date > current date → Mark as “Upcoming”
- If due date == current date → Mark as “Due”
- If dose completed (manually updated by staff) → Mark as “Completed”
- If due date < current date and not completed → Mark as “Missed”

#### Output

- Vaccination status table
- Notification messages
- Printable health report

The algorithm ensures that each vaccination dose is monitored continuously and provides real-time updates. This automated logic eliminates the need for manual calculations by healthcare workers and provides accuracy in schedule tracking. The system is flexible enough to modify or extend the algorithm based on updated medical recommendations in the future.

## 1.4 FLOW CHART





## CHAPTER 2

### LITERATURE REVIEW

Child vaccination is a critical public health measure aimed at preventing serious diseases and reducing child mortality. According to the World Health Organization (WHO), immunization prevents between 2–3 million deaths every year globally. Over the years, various vaccination tracking systems have been developed, ranging from manual record books to advanced computerized systems. These systems aim to ensure timely vaccinations, monitor vaccination coverage, and provide reminders to parents and healthcare providers. Modern child vaccination management systems focus on automation, accuracy, and real-time tracking. They often include features such as digital registration of children, automated calculation of vaccine schedules based on age, reminders via SMS or app notifications, and generation of comprehensive reports for healthcare administrators. Studies have shown that digital systems significantly reduce missed vaccinations and improve overall immunization rates.

#### 2.1 PREVIOUS WORK / RELATED STUDIES

1. **Electronic Immunization Registry (EIR):**

EIR systems track children's vaccination history and provide alerts for upcoming vaccines. A study in India showed a 20% improvement in on-time vaccination rates after the implementation of EIR.

2. **Mobile App-Based Vaccination Tracking:**

Applications like "MyVaccine" and "VaccinateMe" allow parents to track their child's vaccination schedule on smartphones. These apps include educational content, reminder notifications, and digital vaccine cards.

3. **Automated Scheduling Algorithms:**

Some researchers implemented algorithms that calculate due dates for vaccines based on birth date and previous doses. These algorithms can handle missed doses and adjust the schedule automatically, ensuring adherence to WHO guidelines.

4. **Integration with Health Records:**

Advanced systems integrate vaccination data with national health records, allowing policymakers to analyze trends, detect outbreaks, and plan immunization campaigns more effectively.

## CHAPTER 3

### METHODOLOGY / SYSTEM DESIGN (WITH DEVOPS)

#### 3.1 System Architecture

- The system is built using **Figma design**(frontend), **Python & Flask**(backend), **SQLite** (database).
- DevOps tools like **Jenkins, Git and CI/CD** are used to automate build, testing, and deployment.
- Include a **block diagram** showing interaction between users, server, database, and DevOps pipeline.

#### 3.2 Functional Modules

- **User Registration/Login Module** – Allows parents/guardians to register and login securely.
- **Child Profile Module** – Stores child information and vaccination schedule.
- **Vaccination Reminder Module** – Push notifications via Firebase for upcoming vaccines.
- **Dashboard Module** – Displays vaccination history and upcoming vaccinations.

#### 3.3 DevOps Pipeline

- **Source Code Management** – GitHub repository stores all code.
- **Continuous Integration (CI)** – Jenkins automatically builds and runs tests when new code is pushed.
- **Continuous Deployment (CD)** – Docker containers deploy the application to cloud (Azure/AWS) automatically.
- **Monitoring & Feedback** – Logs and alerts are monitored to ensure uptime and quick bug fixes.

#### 3.4 Data Flow Diagrams (DFD)

- Level 0: High-level data flow between user, server, and database.
- Level 1: Detailed flow including API calls, data storage, and notifications.

#### 3.5 Database Design

- Tables: Users, Children, Vaccination\_Schedule, Notifications.
- Include primary keys, foreign keys, and table relationships

### 3.6 Tools and Technologies

- **Flutter** – Frontend mobile app.
- **Python & Flask**– Backend REST APIs.
- **SQLite** – Database.
- **Firebase** – Notifications.
- **DevOps Tools** – Git, Jenkins, CI/CD.

### 3.7 Methodology

- Agile + DevOps approach: Requirement gathering → Development → Automated build & testing → Deployment → Monitoring & Feedback.

## CHAPTER 4

### IMPLEMENTATION

#### 4.1 Frontend Implementation

- Screens for **Login/Signup, Child Profile, Vaccination Schedule, Dashboard, and Notifications.**
- Features: User-friendly design, soft pastel theme, responsive layout.
- Use of **Flutter widgets** like List View, Card, Text Form Field, Elevated Button, etc.

#### 4.2 Backend Implementation (Spring Boot)

- RESTful APIs for **CRUD operations**:
  - Create/update child profiles
  - Fetch vaccination schedules
  - Send reminders
- Security with **JWT authentication.**
- Logging & exception handling for robust performance.

#### 4.3 Database Implementation (MySQL)

- Tables: Users, Children, Vaccination\_Schedule, Notifications.
- Relationships: One user → multiple children → multiple vaccination records.

#### 4.4 DevOps Implementation

- **Source Control:** GitHub repository
- **Continuous Integration (CI):** Jenkins pipeline automatically builds code and runs unit tests
- **Continuous Deployment (CD):** Docker containers deployed to cloud (Azure/AWS)
- **Monitoring & Logging:** Cloud monitoring tools to track uptime and error logs

#### 4.5 Features Demonstration

- Login/Signup process.
- Adding child and vaccination schedule.
- Automatic reminders through Firebase.

## CHAPTER 5

### JENKINS IN CHILD VACCINATION TRACKING SYSTEM

#### 1. Introduction to Jenkins

Jenkins is an open-source **automation server** used to implement **Continuous Integration (CI) and Continuous Deployment (CD)**. In our project, Jenkins automated the testing and deployment of the Flask backend APIs, ensuring faster updates and higher reliability.

#### 2. Why Jenkins is Used

- **Continuous Integration:** Every time a developer updates the Flask code, Jenkins automatically builds and tests it, detecting errors early.
- **Continuous Deployment:** Once tests pass, Jenkins deploys the backend to the cloud automatically, saving manual deployment effort.
- **Automation:** Eliminates human error in repetitive tasks like testing, building, and deployment.
- **Version Control Integration:** Jenkins integrates with **GitHub**, allowing seamless updates whenever new code is pushed.

#### 3. How Jenkins Works in the Project

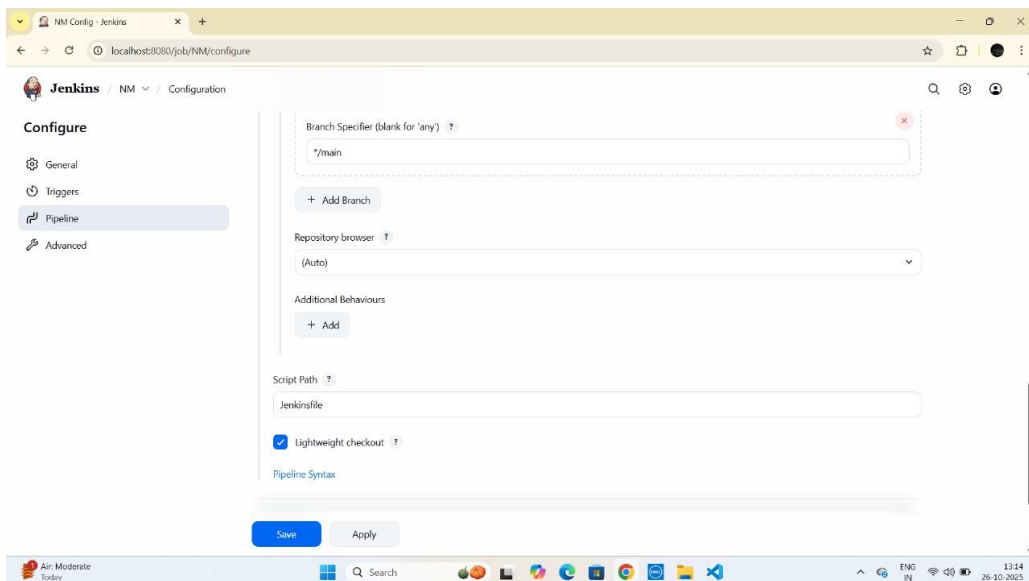
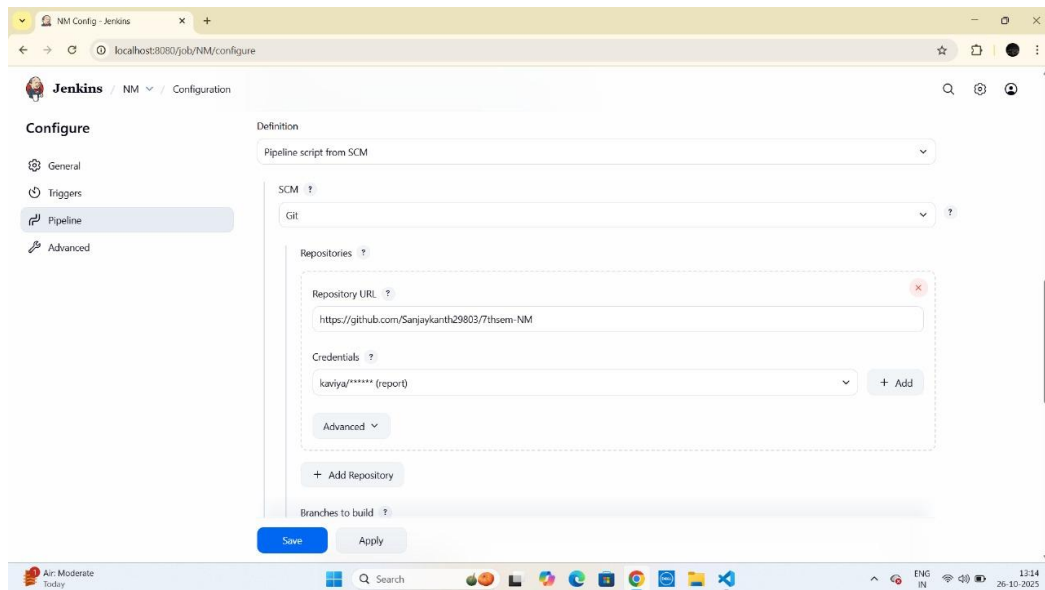
1. **Code Commit:** Developers push Flask backend code to GitHub.
2. **Build Trigger:** Jenkins pipeline detects changes in the repository.
3. **Automated Testing:** Unit tests for API endpoints are executed automatically.
4. **Deployment:** If tests pass, Jenkins deploys the backend to the cloud server (Heroku, AWS, or local VM).
5. **Notification:** Jenkins sends alerts on success or failure to the development team.

#### 4. Benefits Observed

- Reduced manual intervention and human errors.
- Faster release cycles with frequent updates to the system.
- Real-time monitoring of build status and automated rollback in case of failures.
- Improved collaboration among developers through shared CI/CD pipelines.

## 5. Jenkins Pipeline Diagram

GitHub (Code Commit) → Jenkins (Build & Test) → Deployment Server (Flask Backend) → Notification



## CHAPTER 6

### TESTING - PYTEST

#### 1.Introduction to Testing

Testing ensures that the software works as expected and eliminates bugs or errors before deployment. For our child vaccination system, we focused on automated testing of the Flask backend APIs using Pytest, a popular Python testing framework.

#### 2. Why Pytest is Used

- Automation: Pytest allows running tests automatically every time code is updated.
- Ease of Use: Minimal boilerplate code is required, making writing tests simple.
- Integration with Jenkins: Pytest can be directly integrated into Jenkins pipelines for CI/CD automation.
- Reliable Verification: Ensures that API endpoints, data processing, and database operations work correctly.

#### 3. How Testing is Done Using Pytest

1. Test Setup: Create a separate Python file (e.g., test\_app.py) for test functions.
2. Writing Test Cases: Each function tests a specific feature, such as:
  - Registering a child for vaccination
  - Fetching vaccination records
  - Updating vaccination status
3. Running Tests: Execute all tests using the command:
4. `pytest test_app.py`
5. Validation: Pytest checks if the actual output matches the expected output.
6. Reporting: Jenkins collects the Pytest results and provides pass/fail status for each test.

#### 4. Example Test Case

```
from app import app

def test_home_route():
    tester = app.test_client()
    response = tester.get('/')
```

```
assert response.status_code == 200
```

- This test ensures the home route is accessible and returns HTTP status 200.

## 5. Benefits Observed

- Early detection of bugs or broken features.
- Ensured data consistency in vaccination records.
- Automated testing reduced manual effort and increased deployment confidence.
- Seamless integration with Jenkins allowed continuous testing for every code update.

- **CODE**

```
def test_home_page(client):
    """Test if home page loads correctly"""
    response = client.get('/')
    assert response.status_code == 200
    assert b"Vaccination Management System" in response.data

def test_add_child(client):
    """Test adding a new child"""
    response = client.post('/add_child', data={
        'name': 'Test Child',
        'dob': '2020-01-01',
        'phone': '9999999999'
    }, follow_redirects=True)

    assert response.status_code == 200
    assert b"Child added successfully" in response.data

    # Verify in database
    conn = sqlite3.connect(TEST_DB)
    cursor = conn.cursor()
    cursor.execute("SELECT * FROM children WHERE name = ?", ('Test Child',))
    child = cursor.fetchone()
    conn.close()
    assert child is not None

def test_vaccine_schedule(client):
    """Test vaccine schedule generation"""
    response = client.get('/schedule')
    assert response.status_code == 200
    assert b"Vaccine Schedule" in response.data
```



## CHAPTER 7

### DEPLOYMENT- RENDER

Deployment is the process of making the Child Vaccination Tracking System accessible to users such as healthcare workers, parents, and administrators. It ensures real-time monitoring of child vaccination records, schedules, and alerts, thereby improving healthcare management.

The system is built using **Python Flask** for the backend, **HTML/CSS** for the frontend and a **database** such as SQLite or MySQL. For continuous integration and deployment, we have used **Jenkins**, which automates building, testing, and deploying the application.

#### Deployment Steps:

1. **Preparation:** All project dependencies are listed in requirements.txt and verified locally using python app.py.
2. **Jenkins Pipeline:** A Jenkins job is configured to pull code from the GitHub repository. On each code push, the pipeline installs dependencies, runs tests using **Pytest**, and deploys the application if tests pass.
3. **Hosting:** The Flask application is hosted on a server using **Gunicorn** as the WSGI server and optionally **NGINX** as a reverse proxy. Jenkins ensures continuous deployment, automatically updating the application whenever new code passes testing.

#### Benefits:

- **Automation:** Minimal manual intervention is required.
- **Reliability:** Only tested code is deployed.
- **Rapid Updates:** Any new feature or bug fix is live immediately after successful testing.
- **Scalability:** The system can be deployed on cloud servers or containerized environments for larger-scale use.

Environment

7thsem-NM

Events

Settings

MONITOR

Logs

Metrics

MANAGE

Environment

Shell

Scaling

Previews

Disks

Changelog

Invite a friend

Contact support

Render Status

My project / Production / 7thsem-NM

Search

+ New

Upgrade

Connect

Manual Deploy

7thsem-NM

Python 3

Free

Upgrade your instance

Service ID: `srv-d3uo8mfdow73e762tg`

`Sanjaykanth29803 / 7thsem-NM` `13` main

`https://seventhsem-nm-q0t.onrender.com`

Settings

General

Name

7thsem-NM

A unique name for your Web Service.

Edit

Region

Oregon (US West)

Your services in the same region can communicate over a private network.

Free

0.1 CPU

612 MB

Update

Instance Type

Please enter your payment information to select an instance type with higher limits.

Environment

7thsem-NM

Events

Settings

MONITOR

Logs

Metrics

MANAGE

Environment

Shell

Scaling

Previews

Disks

Changelog

Invite a friend

Contact support

Render Status

My project / Production / 7thsem-NM

Search

+ New

Upgrade

Connect

Manual Deploy

Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more.

Upgrade now

October 26, 2025 at 11:37 AM

In Progress

Cancel deploy

All logs

Search

Live tail

GMT+5:30

Oct 26 11:37:37 AM

Collecting six>=1.5 (from python-dateutil>=2.8.2->pandas->r requirements.txt (line 2))

Oct 26 11:37:37 AM

Downloading six-1.17.0-py2.py3-none-any.whl.metadata (1.7 kB)

Oct 26 11:37:37 AM

Downloading flask-3.1.2-py3-none-any.whl (103 kB)

Oct 26 11:37:38 AM

Downloading pandas-2.3.3-cp313-cp313-manylinux\_2\_24\_x86\_64.manylinux\_2\_28\_x86\_64.whl (12.3 MB)

Oct 26 11:37:38 AM

12.3/12.3 MB 138.6 MB/s eta 0:00:00

Oct 26 11:37:38 AM

Downloading numpy-2.3.4-cp313-cp313-manylinux\_2\_27\_x86\_64.manylinux\_2\_28\_x86\_64.whl (18.6 MB)

Oct 26 11:37:38 AM

16.6/18.6 MB 147.9 MB/s eta 0:00:00

Oct 26 11:37:38 AM

Downloading scikit\_learn-1.7.2-cp313-cp313-manylinux2014\_x86\_64.manylinux\_2\_17\_x86\_64.whl (9.4 MB)

Oct 26 11:37:38 AM

9.4/9.4 MB 72.4 MB/s eta 0:00:00

Oct 26 11:37:38 AM

Downloading blinker-1.9.0-py3-none-any.whl (8.5 kB)

Oct 26 11:37:38 AM

Downloading click-8.3.0-py3-none-any.whl (107 kB)

Oct 26 11:37:38 AM

Downloading itsdangerous-2.2.0-py3-none-any.whl (16 kB)

Oct 26 11:37:38 AM

Downloading Jinja2-3.1.6-py3-none-any.whl (134 kB)

Oct 26 11:37:38 AM

Downloading joblib-1.5.2-py3-none-any.whl (308 kB)

Oct 26 11:37:38 AM

Downloading markupsafe-3.0.3-cp313-cp313-manylinux2014\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux\_2\_28\_x86\_64.whl (22 kB)

Oct 26 11:37:38 AM

Downloading python\_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)

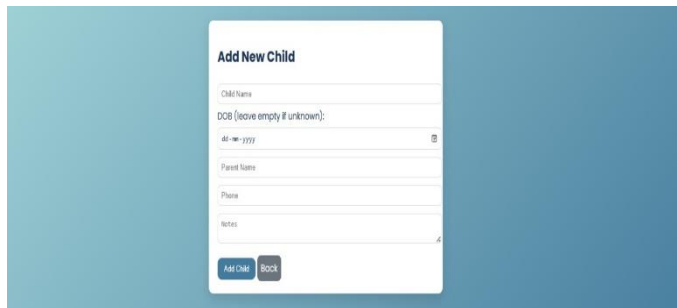
Need better ways to work with logs? Try the [Render CLI](#), [Render MCP Server](#), or set up a [log stream integration](#)

## CHAPTER 8

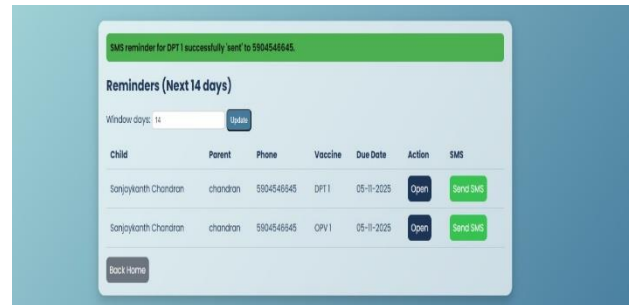
### OUTPUT AND SCREENSHOTS

#### Dashboard/Homepage

- Displays child details, vaccination schedules, and alerts for upcoming vaccines.
- Visual indicators show vaccination status: completed, pending, or overdue.
- Expected Screenshot: Dashboard with child list and vaccination status charts or indicators.

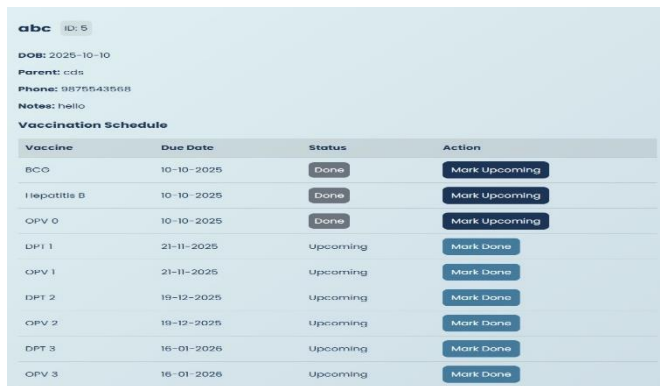


A form titled "Add New Child" with fields for Child Name, DOB (with a date picker), Parent Name, Phone, and Notes. It includes "Add Child" and "Back" buttons at the bottom.



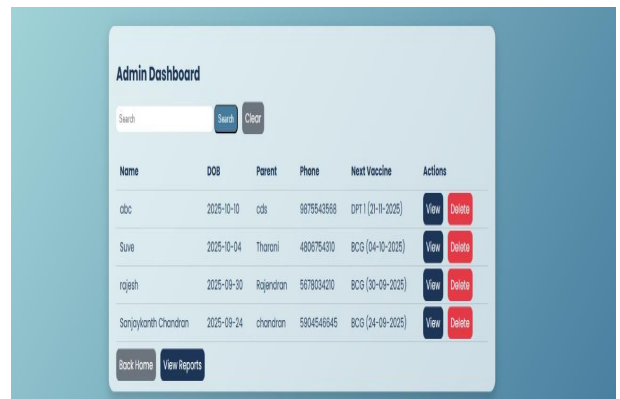
A screen titled "Reminders (Next 14 days)" showing a table of reminders. A green banner at the top says "SMS reminder for DPT1 successfully sent to 5904546645." The table has columns for Child, Parent, Phone, Vaccine, Due Date, Action, and SMS. It lists two reminders for Sanjaykaranth Chandran.

Child	Parent	Phone	Vaccine	Due Date	Action	SMS
Sanjaykaranth Chandran	chandran	5904546645	DPT1	05-11-2025	Open	Send SMS
Sanjaykaranth Chandran	chandran	5904546645	OPV1	05-11-2025	Open	Send SMS



Child details for "abc" (ID: 5) including DOB, Parent, Phone, and Notes. Below is a "Vaccination Schedule" table.

Vaccine	Due Date	Status	Action
BCG	10-10-2025	Done	Mark Upcoming
Hepatitis B	10-10-2025	Done	Mark Upcoming
OPV 0	10-10-2025	Done	Mark Upcoming
DPT 1	21-11-2025	Upcoming	Mark Done
OPV 1	21-11-2025	Upcoming	Mark Done
DPT 2	19-12-2025	Upcoming	Mark Done
OPV 2	19-12-2025	Upcoming	Mark Done
DPT 3	16-01-2026	Upcoming	Mark Done
OPV 3	16-01-2026	Upcoming	Mark Done



Admin Dashboard with a search bar and a table of children. It includes "Back Home" and "View Reports" buttons.

Name	DOB	Parent	Phone	Next Vaccine	Actions
abc	2025-10-10	cds	9876543210	DPT1 (21-11-2025)	View Delete
Sure	2025-10-04	Tharani	490675430	BCG (04-10-2025)	View Delete
rajesh	2025-09-30	Rajendran	5678901230	BCG (30-09-2025)	View Delete
Sanjaykaranth Chandran	2025-09-24	chandran	5904546645	BCG (24-09-2025)	View Delete

#### Add Child Details

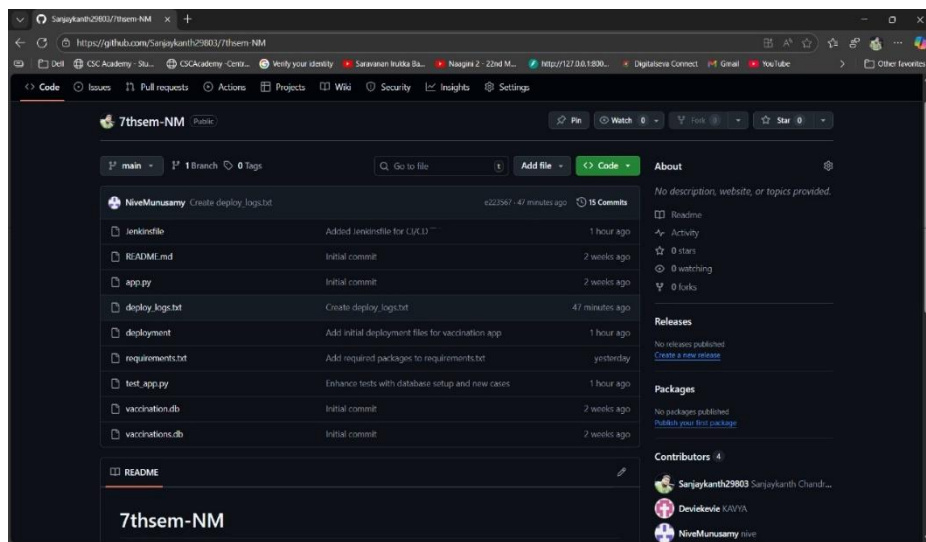
- Users can add new child records including name, age, parent contact, and vaccination history.

#### Vaccination Schedule & Alerts

- Shows upcoming vaccination dates and reminders.
- Provides automatic alerts to parents about pending vaccines.

## Reports & Analytics

- Monthly and weekly vaccination completion reports.
- Graphs visualize trends, such as number of children vaccinated per month.



## CHAPTER 9

### CONCLUSION AND FUTURE SCOPE

#### 9.1 Conclusion

The Child Vaccination Tracking System successfully provides an efficient, secure, and user-friendly platform to manage child vaccination records. The system helps parents and healthcare providers to:

- Keep accurate records of vaccination history.
- Receive timely alerts and reminders for upcoming vaccines.
- Generate reports and visualize vaccination trends.
- Reduce missed or delayed vaccinations, improving overall child health.

#### 9.2 Future Scope

The system can be enhanced further to provide additional features and wider applicability:

1. **Mobile Application Integration** – Develop Android and iOS apps to allow parents to access vaccination schedules on the go.
2. **SMS and Email Notifications** – Automatic reminders through SMS and email to ensure vaccines are not missed.
3. **Integration with Government Health Databases** – Sync data with national vaccination programs for better coverage and reporting.
4. **AI-based Analytics** – Predict vaccination trends and identify areas with low coverage for targeted health interventions.
5. **Multi-language Support** – Make the system accessible to users across different regions by providing regional language options.
6. **Security Enhancements** – Implement advanced authentication mechanisms, such as OTP or biometric login, to protect sensitive child health data.
7. **Telemedicine Features** – Provide virtual consultations and guidance on vaccine-related queries.

By implementing these enhancements, the system can evolve into a comprehensive **child healthcare management platform**, contributing significantly to public health and improving vaccination rates worldwide.

## CHAPTER 10

### REFERENCES

- World Health Organization. *Immunization coverage*. WHO, 2023. <https://www.who.int/news-room/fact-sheets/detail/immunization-coverage>
- Centers for Disease Control and Prevention (CDC). *Childhood Vaccines*. CDC, 2023. <https://www.cdc.gov/vaccines/parents/index.html>
- Python Software Foundation. *Python Documentation*. Python.org, 2024. <https://docs.python.org/3/>
- Flask Documentation. *Flask Web Development*. Flask, 2024. <https://flask.palletsprojects.com/>
- Jenkins Documentation. *Jenkins User Handbook*. Jenkins.io, 2024. <https://www.jenkins.io/doc/>
- Pytest Documentation. *pytest: simple powerful testing with Python*. Pytest.org, 2024. <https://docs.pytest.org/en/stable/>