# Vaccination Service for Digital Village

Drive link: https://iiitaphyd-my.sharepoint.com/:f:/g/personal/siddh_jain_students_iiit_ac_in/EowCgp8FXP1LoYc0N67a_scBLH9RyEAjnFy7p8tjxh6rhA?e=MHeqYe

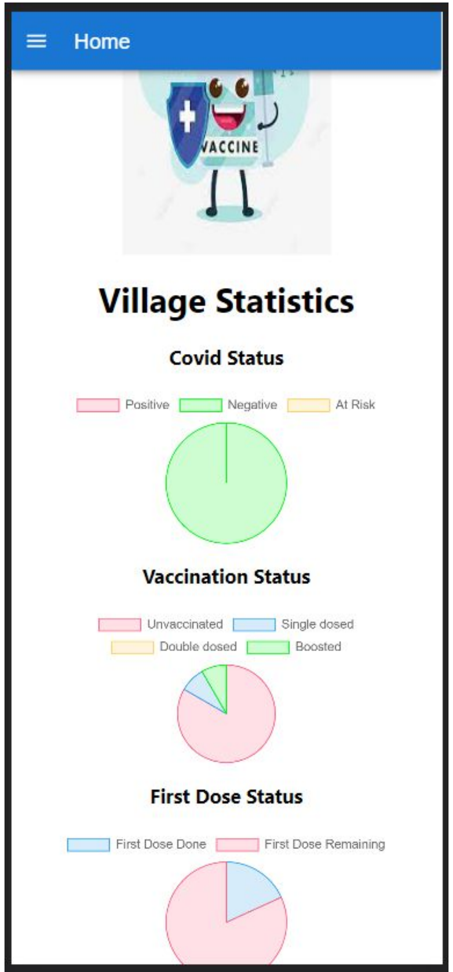
## Overview

## Project

The purpose of this project is to build an application to aid in the everyday working of the village panchayat. It is an application aimed to simplify complicated procedures such as announcements, complaint registration, payments, records of villagers, etc. within the village. It is built primarily to keep in view the working of the Panchayat and the requirements of the villagers. Parts of this app already exist.

Additionally, a vaccination micro-service will be incorporated with the general digital village app through which the village admin will be able to track, edit and update the Covid Vaccination details of every member in the village. Analysis updates related to current affairs say the spread of the new Covid variant shall be published so as to also work as a holistic local news medium. Mechanisms to send out personal notifications regarding vaccination camp information to the villagers shall also be implemented.

This way, the app shall serve the purpose of an all-in-one platform customized to suit the needs of the rural community in an area.



## Scoping

## Sprints - Major Milestones

**Sprint 0-1**
7 Jan - 28 Jan
Initial Requirements Document made

**Sprint 2**
28 Jan - 11 Feb
Wireframes Designed

**Sprint 3**
11 Feb - 25 Feb
Application started with crude frontend and apis

**Sprint 4**
25 Feb - 11 March
Backend, database, frontend connected

**Sprint 5**
11 March - 25 March
Application supports more functionality

**Sprint 6**
25 March - 8 April
Application optimized for mobile interface

**Sprint 7**
8 April - 22 April
Tests Done. Application hosted on Web-server

## Sprints in Phases

### 1 Requirements

Sprint 1- documentation, initial discussions

Sprint 3 - documentation version 2

Sprint 4 - R1 Test plan

Sprint 7 - R2 Test Plan

### 2 Models

Sprint 2 - Tech stack confirmed, wireframes designed

### 3 Stack learning/Setup

Sprint 1- learning postgres and react

Sprint 2- learning django and backend and api setup

### 4 Code

Sprint 2- primitive frontend setup complete

Sprint 3 - frontend complete with home page, personal page, list page, database started

Sprint 4

- Made analysis chart stubs
- Started directed notifications
- Backend setup and completed with view/edit/delete/add api calls
- Database integrated

Sprint 5

- Completed analysis charts
- Applied filter mechanism to villagers' list
- General UI improvement

Sprint 6 - Frontend complete with email notifications and announcements functionality

# Sprints in Phases

5 **Integration**

Sprint 4

- Integrated database with backend
- Integrated frontend with database

Sprint 5 - Integrated email functionality in frontend

6 **Deployment**

Sprint 6- application consolidation

Sprint 7 - application deployed and hosted on web-server

7 **Testing**

Sprint 4- R1 error checking and mitigation

Sprint 7- R2 error checking and mitigation

## Team

**Pratham**
- Django Application - backend setup and database management
- React app resources
- Frontend - backend interface and authentication
- Application deployment
- Documentation and resources
- Client Communication

**Siddh**
- React App - frontend development and improvements
- Database resources
- Application deployment
- Documentation and resources
- Client Communication

**Nithin**
- React App - frontend setup and development
- Database resources
- Documentation and resources
- Client Communication

**Harshavardhan**
- Database resources
- Fronted - contributed
- Documentation and resources
- Client Communication

## Completed and Pending tasks

## Sprint 1-completed

- Create draft requirements

  Requirement document prepared with elaborate discussions with the client. Vaccination module is an integrated project built on digital village

- Analyze and study requirements

  Plan of action discussed between team-members based on the initial requirements

## Sprint-2

- Finalize requirements

    More discussions and deliberations led to some changes in the design and the deliverables.

    Vaccination module decided to be a independent microservice for digital village

- Design wireframe for vaccination module

    Wireframes designed and accorded by all the parties

## Sprint-3

Vaccination Module Design-

- Village Vaccination List

  List of all villagers shown as a tab accessed through the homepage

- Personal Details Tab

  Details of a single villager shown by applying a search method on the database based on their Aadhar number

## Sprint-4

Vaccination Module Design-

- Village Vaccination List-Vaccination Status Filter

  Frontend code pending

- Village Vaccination List-Covid  Status Filter

  Frontend code pending

# Sprint-5

- Data Charts

  Data analysis charts displayed on the home page

- Viewing and Sending Notifications

  Messages sent as emails to the villagers using filters

- Adding new Announcements

  Vaccination/covid related announcement emails sent can now be sent to all the village through the announcement page

- Integration of Vaccination Service with Old module

  Was deemed unrequired. Both the modules function independently

## Sprint-6

- Analysis of Admin Use case

  The Vaccination Module webapp is fully functional with no errors

- Debugging and Error Solving

  All errors faced have been resolved

## Sprint-7

- Testing

  The Vaccination Module Webapp has been successfully deployed on the webserver with interface optimized to make it usable from mobile devices as well

- Review and feedback

  The end product has been delivered to the client

## Tech Stack

## Wireframe

- MockingBot - Wondershare Mockitt offers rapid design and prototyping with intuitive features, allowing designers to collaborate effectively and do what they do best.

## Frontend

- Reactjs - free and open-source front-end JavaScript library for building user interfaces based on UI components. It is maintained by Meta and a community of individual developers and companies.
    - Mui frameworks
    - Axios library for REST calls
    - Rest framework for authentication etc
- Nodejs - Node.js is an open-source, cross-platform JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser.

## Backend

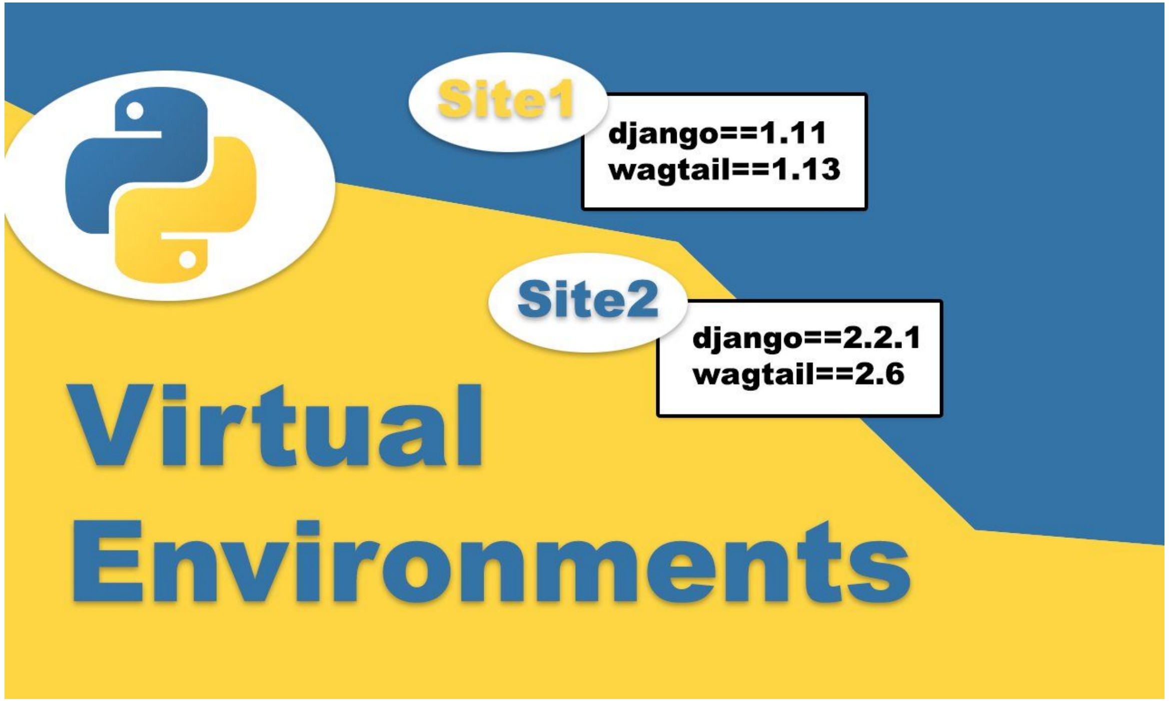
- Django - Python-based free and open-source web framework that follows the model–template–views architectural pattern. It is maintained by the Django Software Foundation, an independent organization established in the US as a 501 non-profit.

## Database

- PostgreSQL- also known as Postgres, it's a free and open-source relational database management system emphasizing extensibility and SQL compliance. It was originally named POSTGRES, referring to its origins as a successor to the Ingres database developed at the University of California, Berkeley

## Research/Study

## Python virtual environments for Django



## Importance of virtual environment setup for django with python

A virtual environment is a way for you to have multiple versions of python on your machine without them clashing with each other, each version can be considered as a development environment and you can have different versions of python libraries and modules all isolated from one another

## Importance of virtual environment setup for django with python

It's very important. For example without a virtualenv, if you're working on an open source project that uses django 1.5 but locally on your machine, you installed django 1.9 for other personal projects. It's almost impossible for you to contribute because you'll get a lot of errors due to the difference in django versions. If you decide to downgrade to django 1.5 then you can't work on your personal projects anymore because they depend on django 1.9.
A virtualenv handles all this for you by enabling you to create separate virtual (development) environments that aren't tied to each other and can be activated and deactivated easily when you're done. You can also have different versions of python

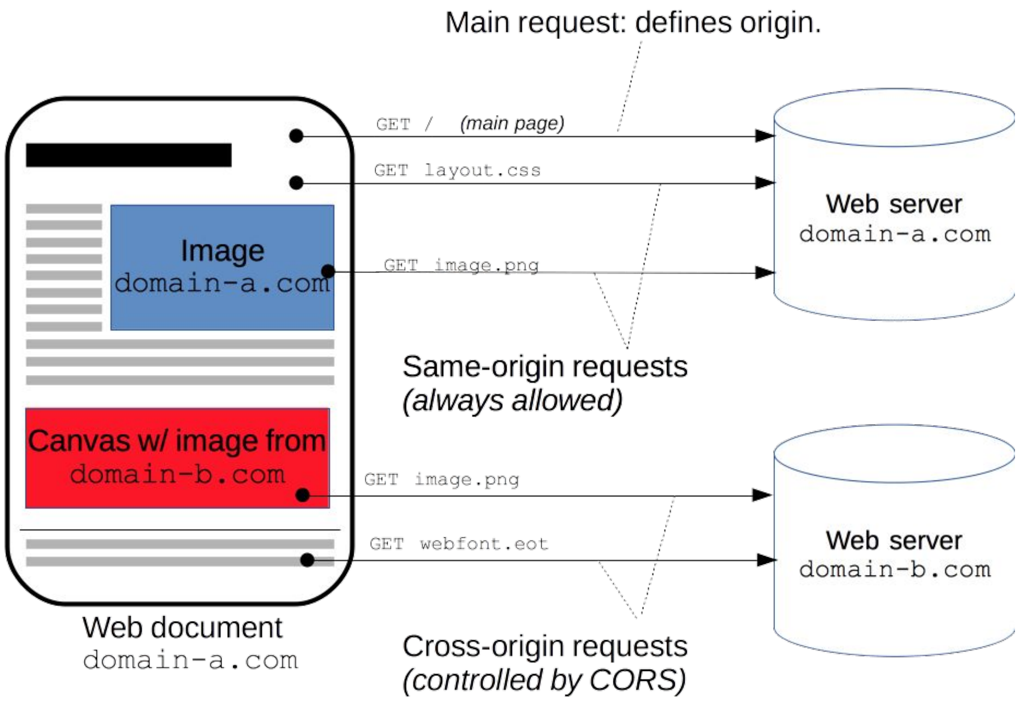
## Importance of virtual environment setup for django with python

You're not forced to but you should, it's as easy as:

virtualenv newenv
cd newenv
source bin/activate # The current shell uses the virtual environment

Moreover it's very important for testing, let's say you want to port a django web app from 1.5 to 1.9, you can easily do that by creating different virtualenv's and installing different versions of django. it's impossible to do this without uninstalling one version (except you want to mess with sys.path which isn't a good idea)

# CORS - Cross Origin Request Sharing



## CORS

- Cross-Origin Resource Sharing (CORS) is an HTTP-header based mechanism that allows a server to indicate any origins (domain, scheme, or port) other than its own from which a browser should permit loading resources. CORS also relies on a mechanism by which browsers make a "preflight" request to the server hosting the cross-origin resource, in order to check that the server will permit the actual request. In that preflight, the browser sends headers that indicate the HTTP method and headers that will be used in the actual request.

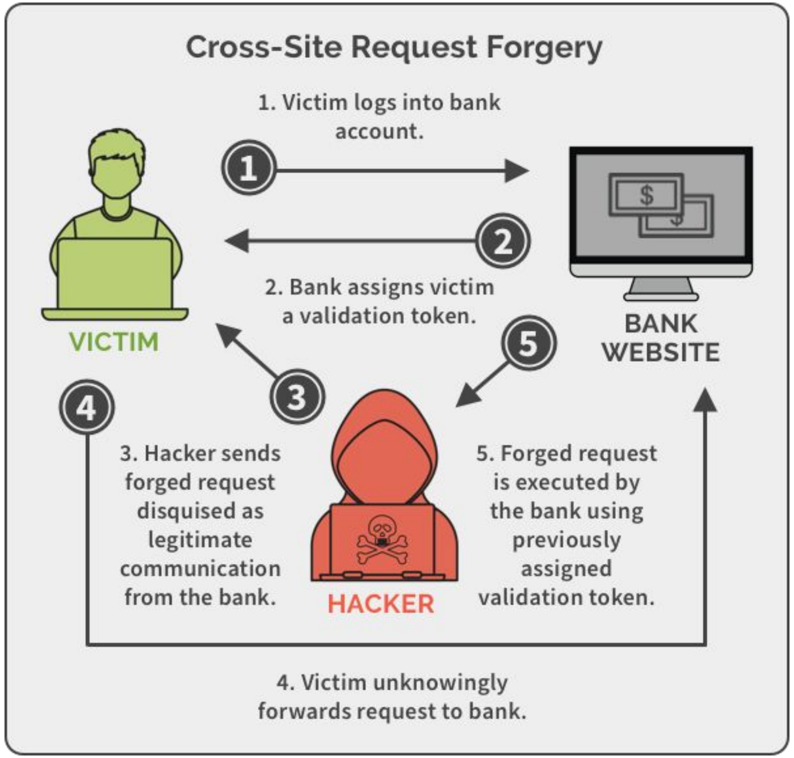
## CORS

- An example of a cross-origin request: the front-end JavaScript code served from https://domain-a.com uses XMLHttpRequest to make a request for https://domain-b.com/data.json.
- For security reasons, browsers restrict cross-origin HTTP requests initiated from scripts. For example, XMLHttpRequest and the Fetch API follow the same-origin policy. This means that a web application using those APIs can only request resources from the same origin the application was loaded from unless the response from other origins includes the right CORS headers.

## CORS

- During the initial stages of making ‘axios’ requests from the react app to the backend django app, errors were faced concerning the Cross-Origin Resource Sharing authentication. This was because, without proper authentication, the backend server does not allow requests to be made to it from a foreign unidentified origin.
- This was finally resolved by extensive research and understanding of the concept and adding the backend site as a proxy to the frontend.

## CSRF- Cross Site Request Forgery


Cross-Site Request Forgery
1. Victim logs into bank account.
2. Bank assigns victim a validation token.
3. Hacker sends forged request disquised as legitimate communication from the bank.
4. Victim unknowingly forwards request to bank.
5. Forged request is executed by the bank using previously assigned validation token.
VICTIM
BANK WEBSITE
HACKER

## CSRF

- To defeat a CSRF attack, applications need a way to determine if the HTTP request is legitimately generated via the application's user interface. The best way to achieve this is through a CSRF token. A CSRF token is a secure random token (e.g., synchronizer token or challenge token) that is used to prevent CSRF attacks. The token needs to be unique per user session and should be of large random value to make it difficult to guess.
- A CSRF secure application assigns a unique CSRF token for every user session. These tokens are inserted within hidden parameters of HTML forms related to critical server-side operations. They are then sent to client browsers.

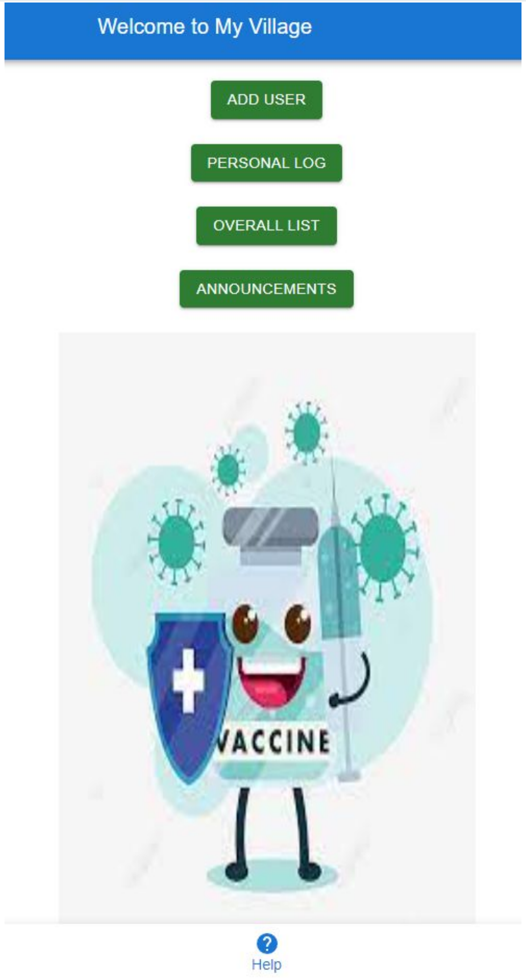
## CSRF

- It is the application team's responsibility to identify which server-side operations are sensitive in nature. The CSRF tokens must be a part of the HTML form—not stored in session cookies. The easiest way to add a non-predictable parameter is to use a secure hash function (e.g., SHA-2) to hash the user's session ID. To ensure randomness, the tokens must be generated by a cryptographically secure random number generator.
- Whenever a user invokes these critical operations, a request generated by the browser must include the associated CSRF token. This will be used by the application server to verify the legitimacy of the end-user request. The application server rejects the request if the CSRF token fails to match the test.

## CSRF

- Further in the project, as we needed REST api requests including push, delete etc., the CSRF prohibition policy of DJANGO caused errors. This was because, without proper authentication, the backend server does not allow data requests to be made to it from a foreign unidentified origin since they are vulnerable to CSRF attacks.
- This was resolved by extensive research and understanding of the concept and including code to release the CSRF restrictions in the test phase of the project. Using rest_framework decorators and its authentication_classes, and  permission_classes, the CSRF authentication was completed

# Thanks!


Welcome to My Village
ADD USER
PERSONAL LOG
OVERALL LIST
ANNOUNCEMENTS
VACCINE
Help