MARIAN COLLEGE
KUTTIKKANAM
AUTONOMOUS
MAKING COMPLETE

DJANGO REST FRAMEWORK BASED PROJECT (JAM stack)

SUBMITTED BY,

ALVIN K SABU

23PMC105

SUBMITTED TO,

MR. SATHEESH KUMAR S

Assistant Professor

DOS: 25/04/2024

CanineConnect

1. Explanation of the architecture of the project

The JAMstack architecture, short for JavaScript, APIs, and Markup, is a contemporary method in web development that prioritizes pre-rendering a website's content wherever feasible and delivering it directly from a content delivery network (CDN), rather than dynamically generating content for each user request.

JavaScript (React.js Frontend):

React.js, a widely-used JavaScript library, manages user interface development in my project. It oversees UI rendering and user interactions.

Utilizing React.js involves creating components that represent various UI elements. These components fetch data from the Django backend via APIs and dynamically render it in the browser.

APIs (Django REST Framework Backend):

Django REST Framework (DRF) serves as the backend, offering APIs to access and manipulate data.

DRF enables the definition of endpoints returning JSON data representing application resources like users, dogs, etc.

The React.js frontend communicates with these API endpoints through HTTP requests to fetch data, submit forms, and perform CRUD operations without requiring page reloads.

The project embraces the JAMstack model by employing JavaScript for core functionality, interacting with APIs, and integrating results into web application markup. React handles client-side rendering and interactions, while Django manages backend tasks, providing data and dynamic content through APIs.

I chose to use the "VITE" framework for building the frontend because it offers a rapid development server and streamlined build process, making it a favoured tool in the React community. It serves as a viable alternative to platforms like Create React App, especially known for its efficiency within the React ecosystem. Here's how Vite's capabilities align with this project:

Build Tool:

- Vite's standout feature is its swift build times, particularly beneficial for sizable projects. Its utilization of modern JavaScript functionalities such as ES module imports contributes to this efficiency.
- While the project's configuration doesn't explicitly include a build script for Vite, it's plausible that Vite operates covertly in the background to build the frontend React application.
- Vite likely handles tasks such as bundling React components, optimizing assets, and producing deployment-ready code for production environments.

2. List each API endpoint with a brief description of its purpose:

1. Note Endpoints:

- **List Notes:** GET /api/notes/

  View: views.NotesViewset.as_view()

  Name: dog-list

  Description: Retrieves list of .

- **Delete Note:** DELETE /api/notes/delete/<int:pk>/

  View: views.NoteDelete.as_view()

  Name: delete-dog

  Description: Deletes a specific dog identified by its primary key (pk).

2. User Authentication Endpoints:

- **User Registration:** POST /api/user/register/

  View: CreateUserView.as_view()

  Name: register

  Description: Allows users to register by providing their credentials.

- **Obtain Token (Login):** POST /api/token/

  View: TokenObtainPairView.as_view()

  Name: token_obtain_pair

  Description: Generates an authentication token (JWT) when users provide valid login credentials.

- **Refresh Token:** POST /api/token/refresh/

  View: TokenRefreshView.as_view()

  Name: token_refresh

  Description: Allows users to refresh their expired authentication token by providing a valid refresh token.

3. Django REST Framework Authentication Endpoint:

- **DRF Authentication:** path ("api-auth/", include("rest_framework.urls"))

  Description: Provides built-in Django REST Framework authentication views for login, logout, and password change/reset**.**

3. Complete Source Code (Public GitHub Repository Link - hyperlink)

https://github.com/Alvin-sabu/fullapi.git

4. Hosting details

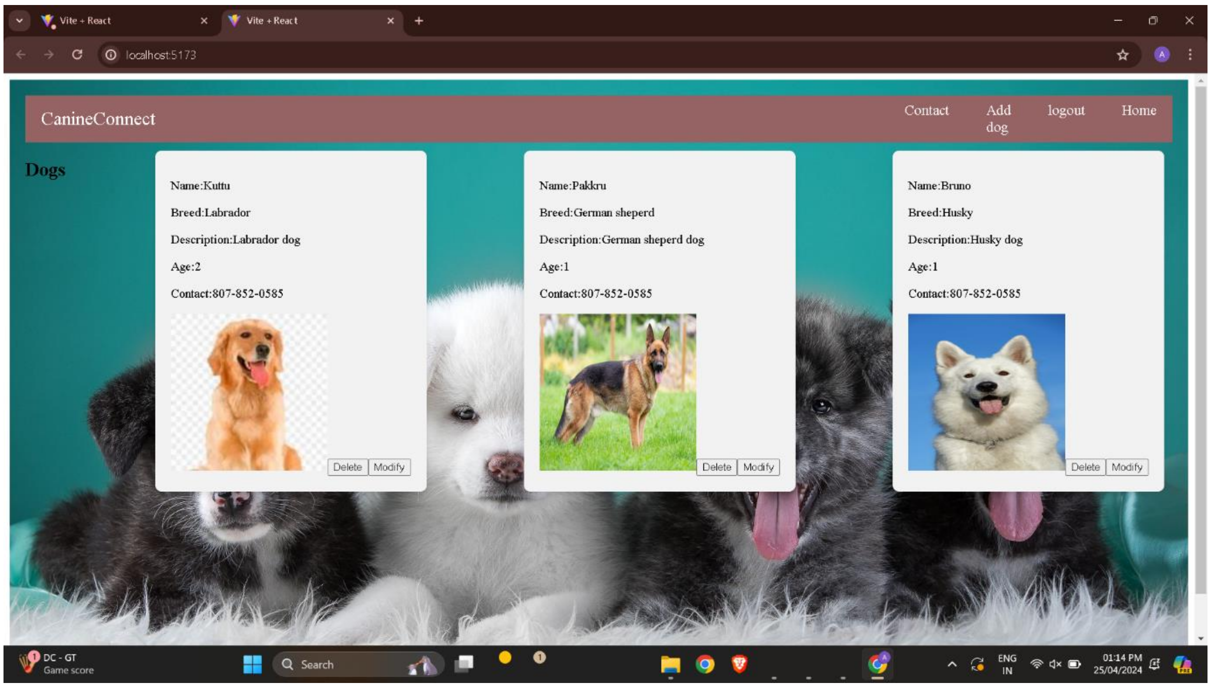
a.Backend

I've opted to host the backend using PythonAnywhere due to its user-friendly platform, which simplifies the deployment and administration of Django applications. PythonAnywhere provides a convenient environment for managing Django apps effortlessly.

https://alvinksabu200115.pythonanywhere.com/

5. Screenshots of the API endpoints in action

a. **Home page:** List all



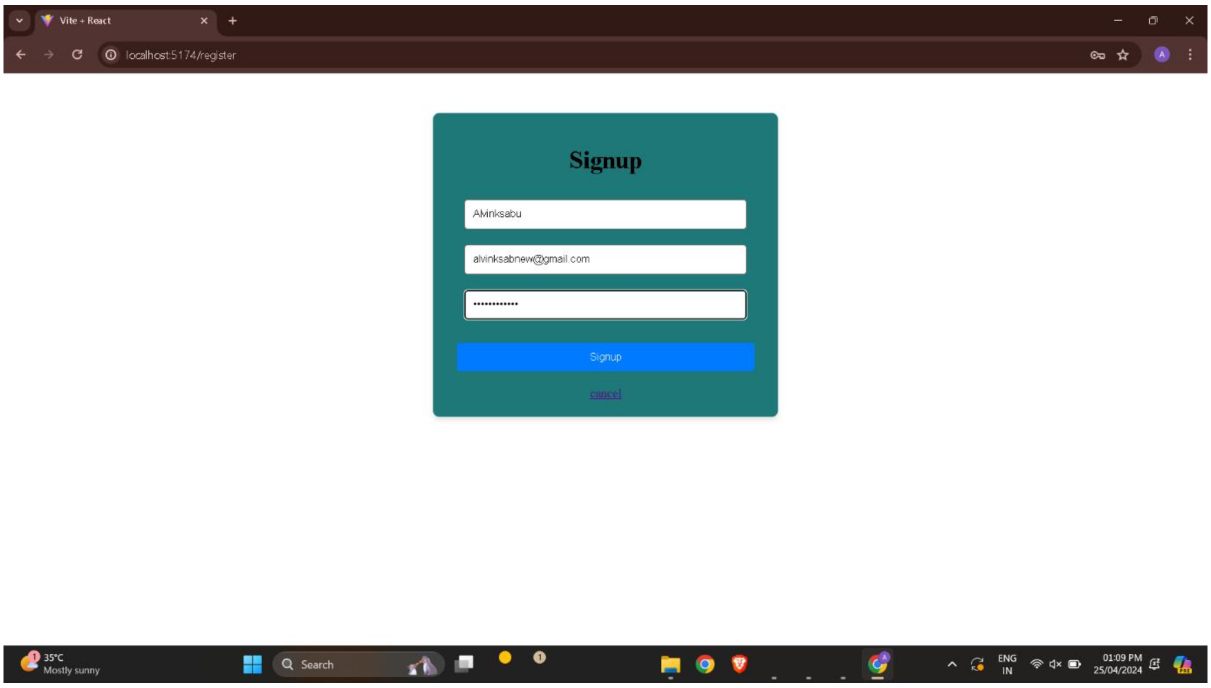
Dogs

Name:Kuttu
Breed:Labrador
Description:Labrador dog
Age:2
Contact:807-852-0585

Name:Pakkru
Breed:German sheperd
Description:German sheperd dog
Age:1
Contact:807-852-0585

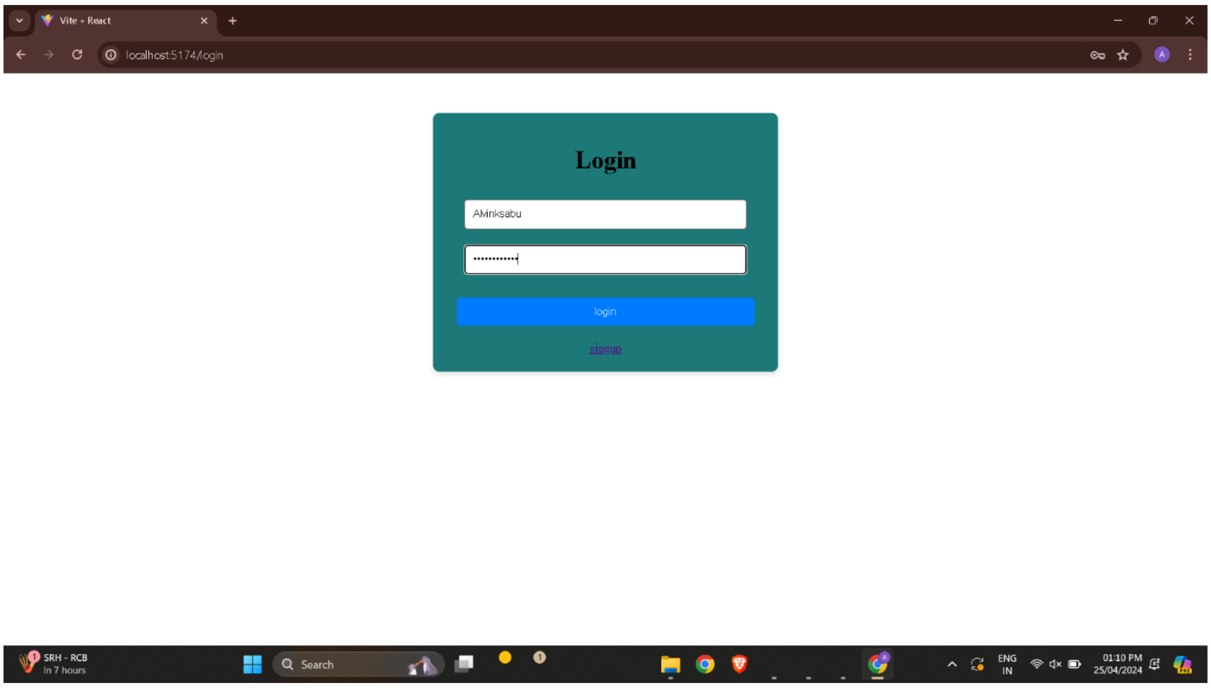
Name:Bruno
Breed:Husky
Description:Husky dog
Age:1
Contact:807-852-0585

b. **User register:** User can register



Alvinksabu
alvinksabnew@gmail.com
••••••••••••

c. User login:User can login after registration


localhost:5174/login

Login

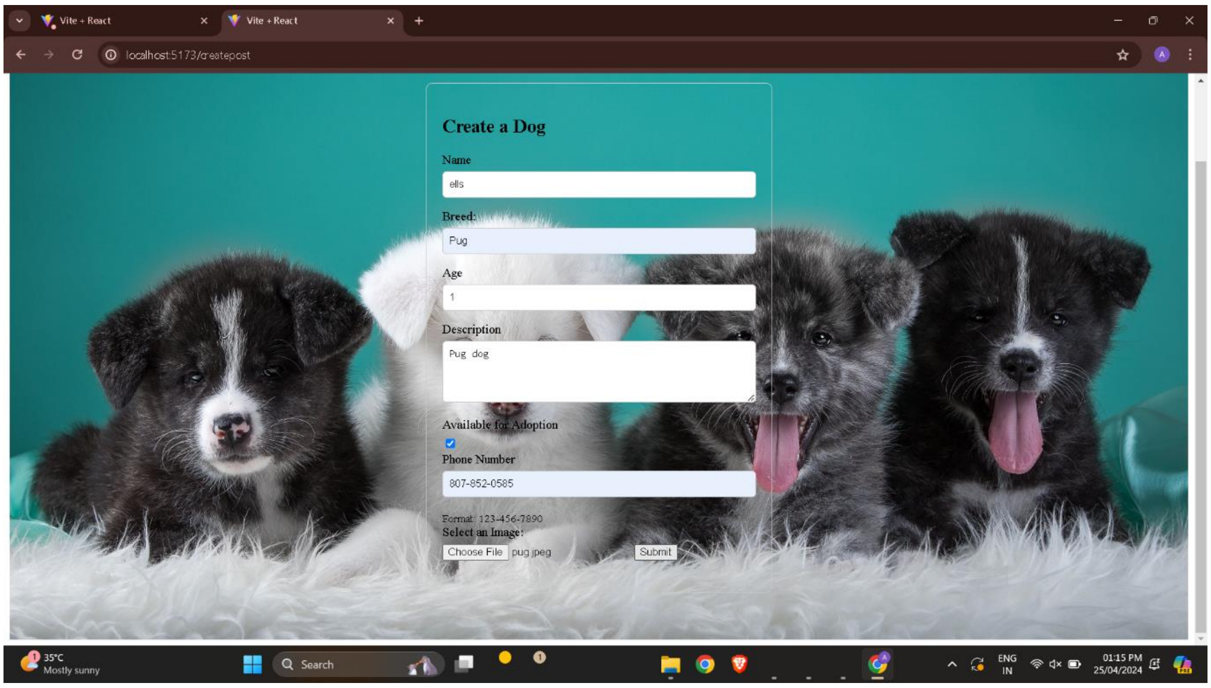
AMinksabu

············

login

singup

d. Add a new dog data:


localhost:5173/createpost

Create a Dog

Name

ells

Breed:

Pug

Age

1

Description

Pug dog

Available for Adoption

☑

Phone Number

807-852-0585

Format: 123-456-7890

Select an Image:

Choose File pug.jpeg

Submit

Also user can update and delete an existing dog data .

6. References (Tutorial Link, Articles etc)

a) https://developer.mozilla.org/enUS/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/React_getting_started

b) https://www.youtube.com/watch?v=c-QsfbznSXI

c) https://youtu.be/SqcY0GlETPk?feature=shared