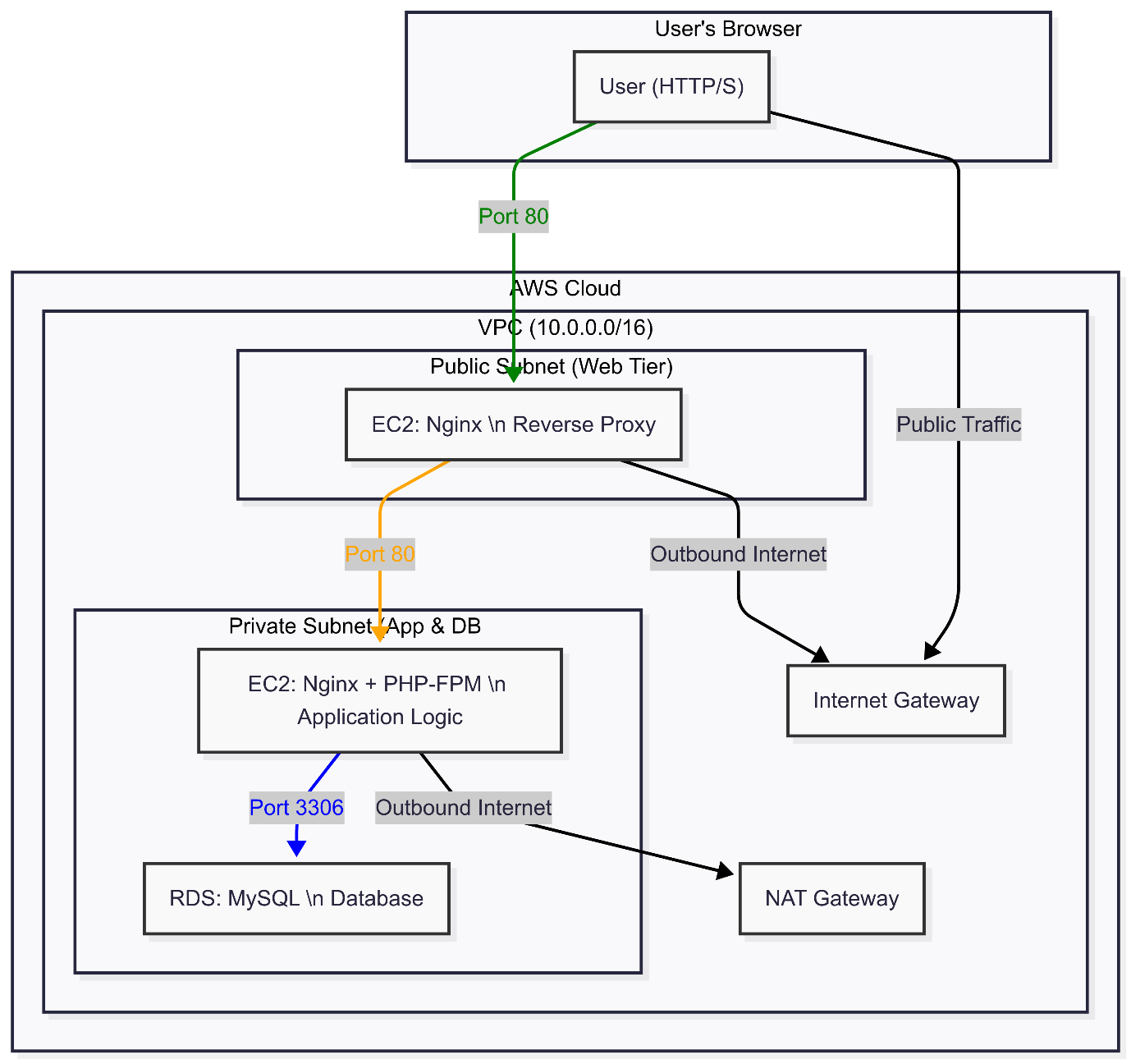
**3-Tier Web Application Deployment on AWS**

This repository contains the complete infrastructure-as-code solution for deploying a scalable 3-tier web application on Amazon Web Services (AWS). The project uses **Terraform** for infrastructure provisioning and **Ansible** for automated server configuration.

**1. Deployed Architecture Diagram**

The architecture consists of three distinct tiers, each with a specific responsibility, ensuring separation of concerns and enhanced security.



**2. How the System Works**

1. **User Interaction**: A user accesses the web server's public IP address in their browser.
2. **Web Tier**: The Nginx server in the public subnet receives the request. It serves the static index.html registration form. When the user submits the form, Nginx acts as a reverse proxy, forwarding the submit.php request to the Application Tier's private IP address.
3. **Application Tier**: The Nginx server in the private subnet receives the request from the Web Tier and passes it to the PHP-FPM service. The submit.php script processes the form data, connects to the RDS database, and inserts the new user information.
4. **Database Tier**: The RDS MySQL instance, located in a private subnet, receives the connection exclusively from the Application Tier (enforced by security groups) and stores the data.
5. **Security**:
   * Only the Web Tier is publicly accessible.
   * The App and Database Tiers are in private subnets, inaccessible from the internet.
   * Security Groups act as stateful firewalls, strictly controlling traffic between tiers.
   * A NAT Gateway allows the App Tier to access the internet for software updates without being publicly exposed.

**3. Code Repository**

All Terraform and Ansible code for this project is hosted on GitHub.

* **GitHub Repository Link:** [**<< https://github.com/VIJAY-989060/3-tier-Terraform-ansible/tree/master>>**](https://www.google.com/search?q=%23)

**4. Prerequisites**

To deploy this infrastructure, you will need:

* An AWS Account with appropriate permissions.
* AWS CLI installed and configured (aws configure).
* Terraform installed.
* Ansible installed.
* An SSH key pair generated in your target AWS region.

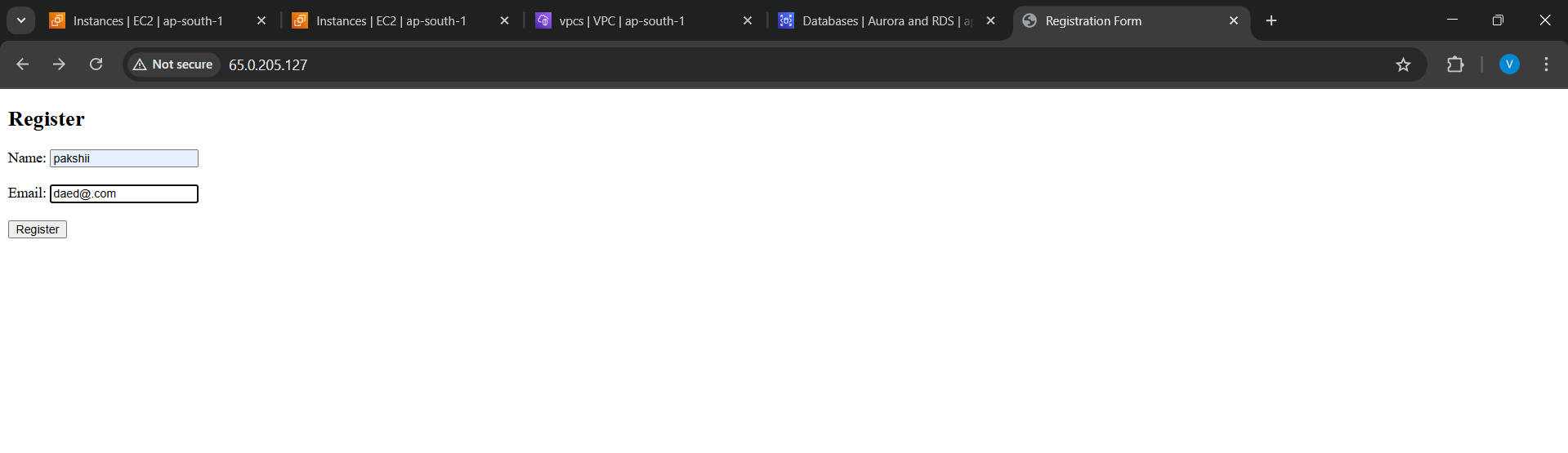
**5. Steps to Deploy**

1. **Clone the Repository:**
2. git clone <<YOUR\_GITHUB\_REPO\_LINK>>
3. cd 3-tier-project
4. **Configure Terraform Variables:**
   * Navigate to the terraform/ directory.
   * Rename terraform.tfvars.example to terraform.tfvars.
   * Edit terraform.tfvars with your specific AWS region, EC2 key pair name, and desired database credentials.
5. **Deploy AWS Infrastructure:**
   * From the terraform/ directory, run the following commands:
   * terraform init
   * terraform apply
   * Review the plan and type yes to confirm. This will build all AWS resources and generate the ansible/inventory.ini file.
6. **Configure Servers with Ansible:**
   * Navigate to the ansible/ directory.
   * Run the playbooks in sequence:
   * ansible-playbook web\_setup.yml
   * ansible-playbook app\_setup.yml
   * ansible-playbook db\_setup.yml
7. **Verify Deployment:**
   * Open a web browser and navigate to the web\_server\_public\_ip provided in the Terraform output.
   * The registration form should be visible and fully functional.

**6. Working Setup Demonstration**

Here are screenshots demonstrating the successfully deployed and functional application.

**Screenshot 1: The Registration Form**



**Screenshot 2: Successful Submission**

