Recarnation Cloud

A Scalable Cloud Platform for the Used Car Marketplace

Team: Nano Clouds

Team Members: Apoorva, Ananya, Junie, Rinku

8th May 2025

Contents

1. Problem Statement & Motivation
2. Proposed Solution
3. Overview at a Glance
4. Technology Stack
5. Single Sign-On (SSO) with Okta
6. Frontend: AWS S3 & RDS
7. Backend: AWS Elastic Beanstalk (VPC, IAM, Security Groups)
8. CI/CD Pipeline: Jenkins & GitHub Actions
9. Live Demonstration
10. Future Enhancements

Problem Statement & Motivation

❑ Rising tariffs and economic constraints have made buying new cars more difficult for many consumers.

❑ Growing demand for affordable, eco-friendly transport exposes limitations in traditional and existing digital used car platforms.

❑ Current platform lacked scalability, security, and user experience.

❑ Build a transparent, secure, and efficient cloud-based solution that empowers both buyers and sellers.

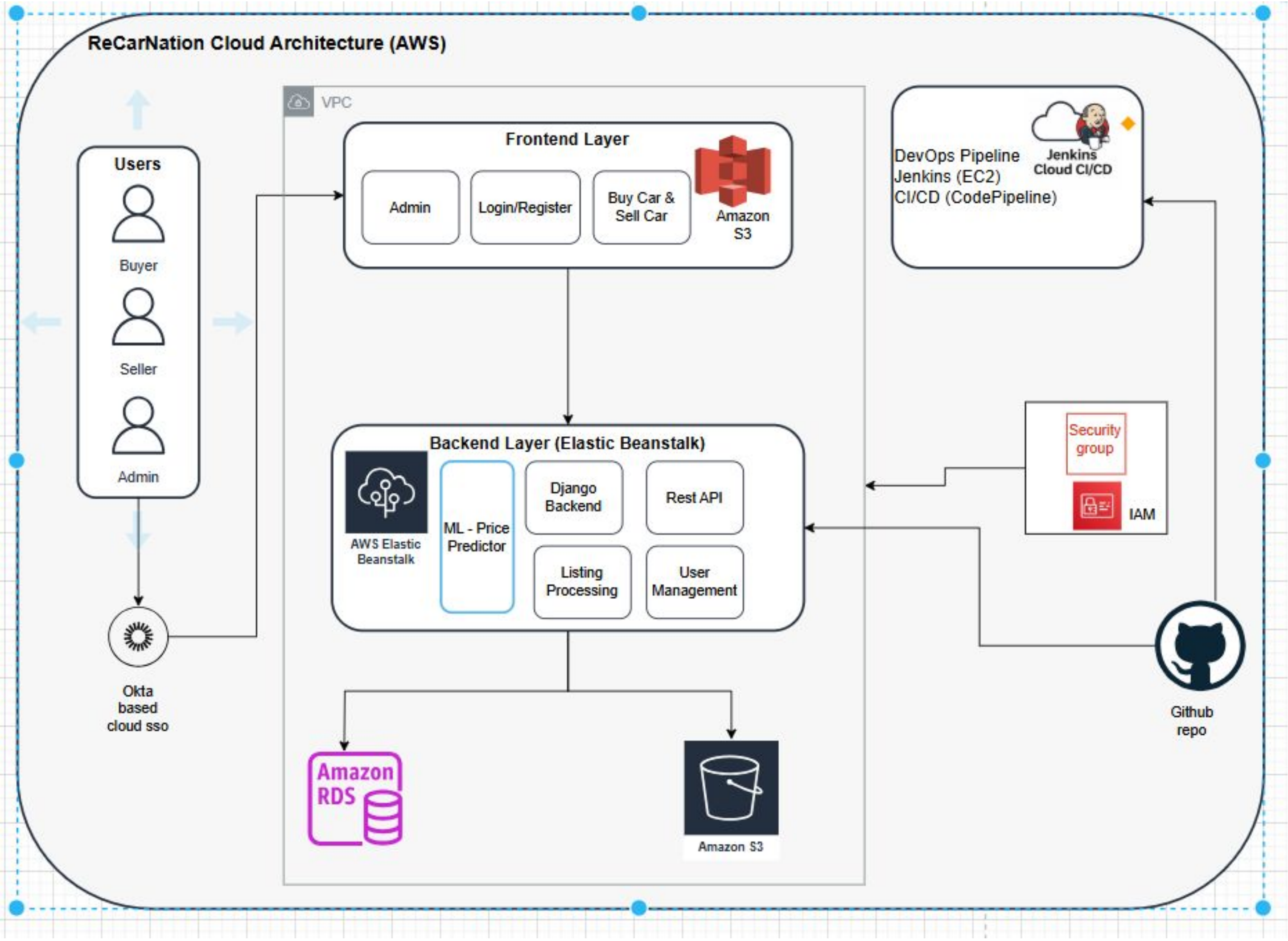
Proposed Solution

- Cloud native Architecture
- Secure Authentication via Okta SSO
- Buy & Sell Interface with Filtering and Listings
- Admin and Price prediction ML model
- Deployed on AWS with CI/CD Integration

Technology Stack

- Frontend: HTML, CSS, Javascript served via S3
- Backend: Django – Elastic Beanstalk
- ML Model: XGBoost using libraries like Pandas, Numpy
- Database: PostgreSQL – Amazon RDS
- CI/CD: Cloud based Jenkins CI/CD pipeline, Github webhooks
- Authentication: Okta SSO
- Version Control: Github
- Orchestration : Docker

Recarnation Architecture



Single Sign-On (SSO) with Okta

- Okta enables secure Single Sign-On (SSO) for users and administrators across all applications.
- GitHub SSO is integrated to streamline authentication for developers.
- Centralized authentication improves security and simplifies user access management.
- User credentials are protected through robust identity and access control mechanisms.

Frontend: Amazon S3 & RDS

- ReCarNation Cloud uses Elastic Beanstalk leverages Amazon S3 to **efficiently store** and serve all static frontend assets.
- All the frontend assets are stored and delivered through S3
- S3 provides a highly scalable and durable storage solution for project-specific media, particularly the **numerous images** associated with each car listing on ReCarNation Cloud.

- Amazon RDS, running **PostgreSQL** engine in our project, serves as the p**rimary persistent datastore** for ReCarNation Cloud.
- It securely manages all crucial application data, including user accounts, detailed car listings (specifications, pricing, seller info), and any transaction or interaction records.
- Amazon RDS, is a **fully managed relational database service**. This handles time-consuming administration tasks like patching, backups, and scaling for our PostgreSQL database.

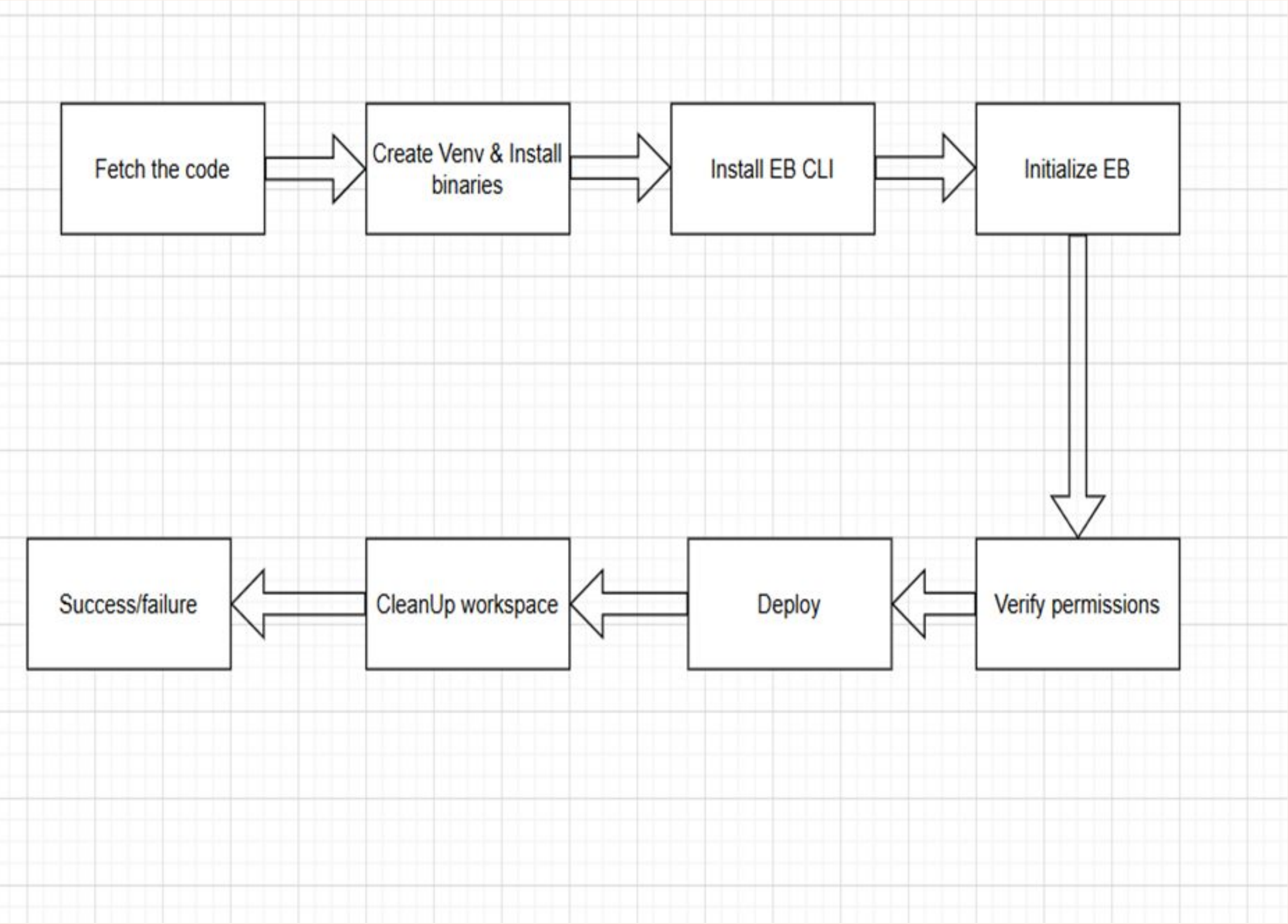
Backend: AWS Elastic Beanstalk (VPC, IAM, Security Groups)

- Django backend is deployed using AWS Elastic Beanstalk.
- Elastic Beanstalk automatically handles the provisioning, scaling, and health monitoring of the Django application.
- Simplifies deployment and auto-scales based on demand.
- Connected Elastic Beanstalk environment securely to an RDS PostgreSQL instance using Django's settings and EB-managed env vars.
- Elastic Beanstalk runs Gunicorn as the WSGI server, configured to handle multiple threads and processes efficiently.

CI/CD Pipeline: Jenkins & GitHub Webhook

Our goal is to automate the end-to-end build and deployment process.

Our Jenkins pipeline is automatically initiated by code commits to github repository, via webhooks ensuring continuous integration of new changes.



Live Demonstration

Future Enhancements

- Add support for AI-based recommendations and fraud detection.
- Globalization: multi-language and currency support.

Thank You!