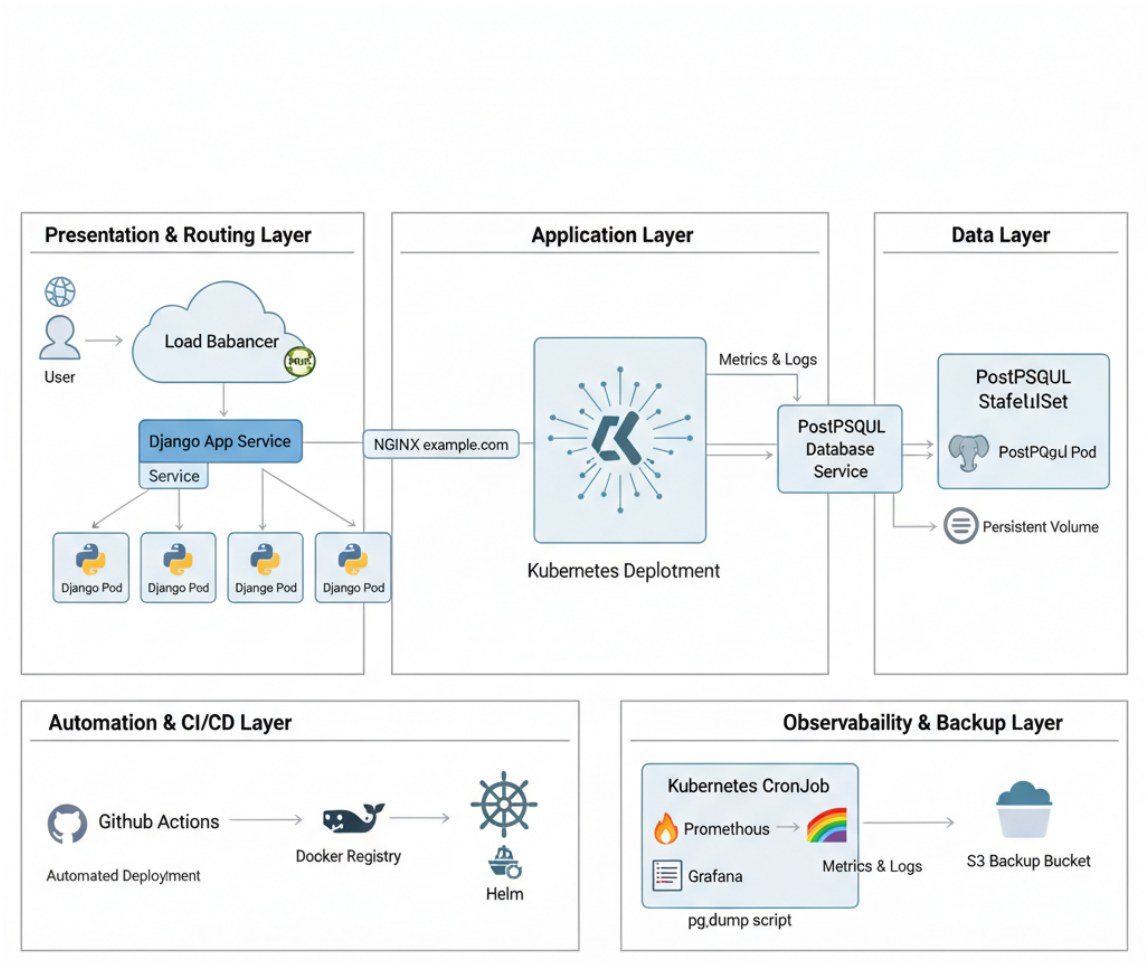
CryptoExchange Infrastructure ProposalSubject

1. Executive Summary

This proposal outlines a cloud-native, scalable, and automated infrastructure for the CryptoCurrencyExchange application. The objective is to migrate the existing Django monolith to a Kubernetes-based environment. The proposed solution prioritizes industry-standard, open-source tools to ensure reliability, maintainability, and rapid deployment.

2. Schematic Design

The following diagram illustrates the proposed high-level architecture, integrated with GitLab CI/CD for automated delivery.



Schematic Architecture Explained

The diagram above illustrates a highly resilient, cloud-native infrastructure designed for Kubernetes:

Traffic Flow: User requests are secured and routed via an **NGINX Ingress Controller**, which load-balances traffic across the application pods.

Application Management: The core Django application runs within a **Kubernetes Deployment**. This ensures high availability by maintaining multiple active pod replicas (self-healing) and enables zero-downtime rolling updates when new code is released.

Data Persistence: Critical application data is stored in a **PostgreSQL StatefulSet**, ensuring data integrity and stable network identity even during pod restarts.

Automated Delivery: GitLab CI/CD powers the pipeline, automatically building Docker images and deploying updates via Helm whenever code is pushed.

Observability & Maintenance: The cluster is continuously monitored by **Prometheus and Grafana** for health metrics, while a dedicated **CronJob** performs automated daily database backups to off-site object storage (S3).

3. Selected Solutions

These are the tools which I have selected to meet the requirements for deploying ,scalability, observability, and automation.

Component	Selected Solution	Justification
Orchestration	**Kubernetes**	The industry standard for managing containerized applications, providing self-healing, scaling, and declarative configuration.
CI/CD	**GitLab CI/CD**	A unified DevOps platform that seamlessly integrates source code management with automated build and deployment pipelines.
Ingress	**NGINX Controller**	A high-performance, production-grade ingress controller for managing external access and Layer 7 routing to services.

Database	**PostgreSQL (StatefulSet)**	Deployed as a StatefulSet to ensure stable network identity and persistent storage, essential for data integrity.
Monitoring	**Prometheus & Grafana**	Provides deep, real-time visibility into cluster health, resource usage, and application performance metrics.
Logging	**Grafana Loki**	A lightweight, cost-effective log aggregation system that integrates tightly with Grafana, allowing correlation between metrics and logs.
Backup	**Kubernetes CronJob**	A cloud-native approach to scheduled tasks, running ephemeral containers to securely dump and ship database backups to object storage.

4. Future Considerations

- Implement Horizontal Pod Autoscaling (HPA) for the application layer.
- Upgrade the database to a highly available (HA) cluster with automatic failover.
- Harden security with NetworkPolicies to isolate database traffic.