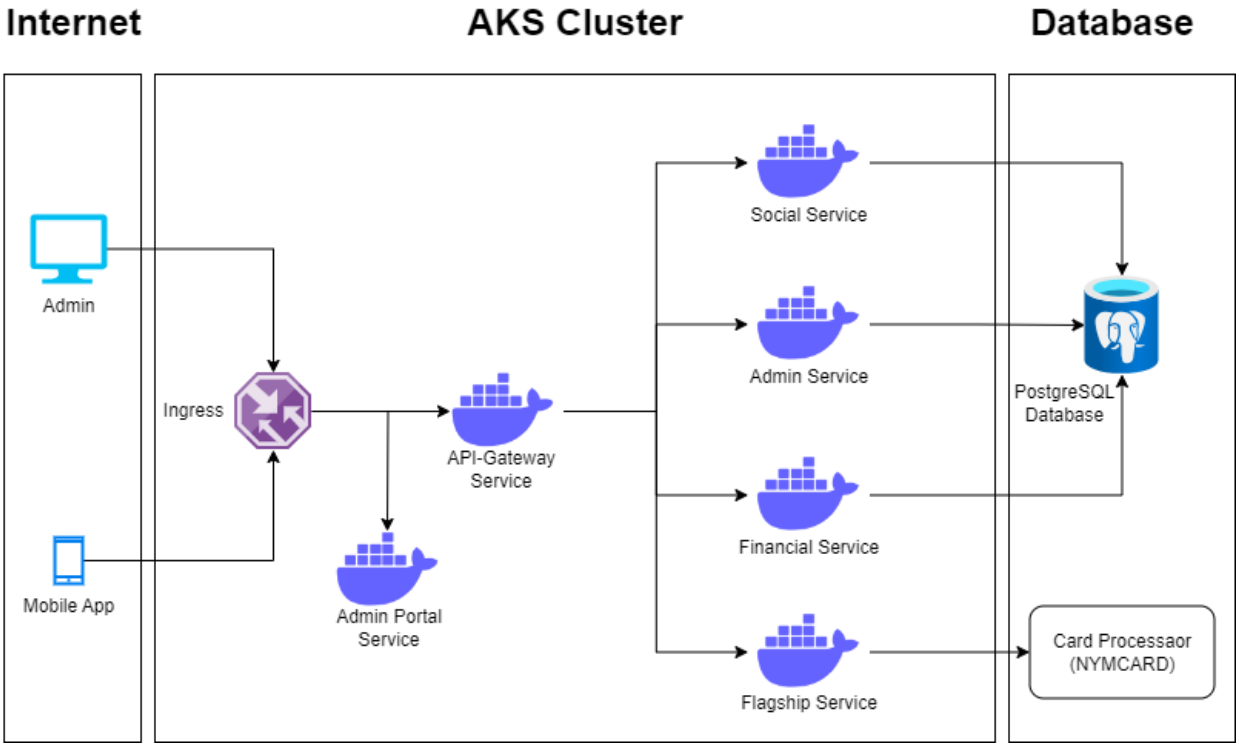
Architecture Overview for Clync Cluster (Current)



1. AKS Cluster Overview

We are utilizing **Azure Kubernetes Service (AKS)** for managing our containerized applications. Our architecture revolves around routing traffic, backend services interacting with databases, and an automated DevOps pipeline.

1.1 Traffic Flow with Ingress

- **NGINX Ingress Controller** is responsible for routing external traffic within the AKS cluster.
- The ingress routes traffic to two primary services:
    - **API Gateway**: A central Node.js service that handles routing for backend microservices.
    - **Admin Portal**: The admin panel service that is hosted on a separate path (/admin).

1.2 Backend Microservices

- The **API Gateway** directs requests to specific backend services based on the endpoints. These backend services are designed to handle various business logic:
    - **Social Service**: Manages the social features of the app.

    - **Financial Service**: Manages financial-related features on Clync backend.
    - **Flagship Service**: Manages financial-related transactions on flagship side.
    - **Admin Service**: Manages admin-related APIs.

- Backend services are containerized and run as separate services within the AKS cluster, scaling independently and interacting with the database.

1.3 Database Setup

- Currently, all backend services are querying a shared **Azure SQL Database**.
- The intention is to shard the database by separating the **social** and **financial** module data into distinct databases. However, this sharding implementation is delayed due to continuous updates in design and flows, because of this we are unable to structure our database schema. So, it will be done once all the scenarios and flows are finalized.

2. DevOps Implementation

Our project follows a modern DevOps approach that automates the build, deployment, and management of our application.

2.1 GitHub Repository Structure

- The codebase for our application is hosted in a GitHub repository, including all services and configurations.
- The repository includes:
    - Backend services
    - Frontend admin portal
    - Helm charts for Kubernetes deployments
    - Docker files for containerization

2.2 Continuous Integration (CI)

- **GitHub Actions** is employed for CI/CD pipelines. Each time code is committed to the GitHub repository, a set of actions are triggered automatically:
    - The code is pulled and the necessary steps for the build process are executed.
    - **Docker images** are built for each of our services.
    - The images are then pushed to **Azure Container Registry (ACR)**.

2.3 Continuous Deployment (CD)

- Once the Docker images are pushed to ACR, **Helm charts** are used to deploy and manage the services on the AKS cluster.
- Helm charts contain deployment files, service definitions, and NGINX configurations for routing.
- The deployment files ensure that new images are pulled from ACR and deployed to the appropriate pods, ensuring our services remain up-to-date without manual intervention.

Architecture Overview for Clync Cluster (Future)