DevFlow Visualizer System Design Document

Executive Summary

Project Type: Web Application

An internal developer tool that helps engineers visualize system architecture and application flows in real time.

Core Features

- Architecture visualization

- real-time diagram rendering

- collaborative editing

- version tracking

- export to image/PDF

- access control

Constraints

- Performance

- Cost

- Security

- Compliance

User Scale: 100-500 MAU, 500 concurrent users

System Overview

Functional Goals

- Real-time system architecture visualization

- Dynamic diagram rendering

- Collaborative editing and version tracking

Non-Functional Requirements

- High performance and scalability

- Strong security and compliance

- Ease of use and accessibility

Primary User Personas

- Software engineers

- System architects

- DevOps teams

Architecture Design

The system will use a microservices architecture with a web frontend, API gateway, and backend services for diagram rendering and data storage.

System Components

Component	Responsibility	Technologies	Interfaces
Web Frontend	User interface and diagram editor	React, TypeScript	API Gateway
API Gateway	Handles incoming requests and routes to backend services	NGINX, Kubernetes	Web Frontend, Diagram Rendering Service
Diagram Rendering Service	Renders diagrams in real-time	Node.js, Mermaid.js	API Gateway, Data Storage Service
Data Storage Service	Stores diagram data and user information	PostgreSQL, Redis	Diagram Rendering Service, API Gateway

Database Design

Database Type: Relational database

High availability and durability

Table: diagrams

Column	Type	Nullable	Description
id	integer	False	Unique diagram ID
name	string	False	Diagram name
data	json	False	Diagram data
created_at	datetime	False	Diagram creation timestamp

Table: users

Column	Type	Nullable	Description
id	integer	False	Unique user ID
username	string	False	Username
email	string	False	Email address
password	string	False	Password hash

Cost Estimation

Cost Item	Monthly Cost	Rationale
Infrastructure costs	$1000.0	Estimated infrastructure costs
Personnel costs	$5000.0	Estimated personnel costs

Testing & QA Strategy

Unit testing for backend services – Backend services

Integration testing for API gateway – API gateway

End-to-end testing for web frontend – Web frontend

Load and stress testing for system – System

Security testing for system – System

Appendices

Glossary

DevFlow Visualizer: Internal developer tool for visualizing system architecture and application flows

Mermaid.js: Diagram rendering library

References

https://www.mermaid-js.com/

https://kubernetes.io/

System Architecture & Diagrams

System Architecture Diagram (Mermaid Code):

```
flowchart LR
User -->| Requests | API_Gateway
API_Gateway -->| Routes | Diagram_Rendering_Service
Diagram_Rendering_Service -->| Renders | Diagram
Diagram --> API_Gateway
API_Gateway --> User
```

User Flow Diagram (Mermaid Code):

```
flowchart TD
Start --> Login
Login --> Dashboard
Dashboard --> View_Diagrams
View_Diagrams --> End
```

Database ER Diagram (Mermaid Code):

```
erDiagram
DIAGRAM {
int id
string name
}
USER {
int id
string username
}
DIAGRAM ||--o{ USER : created_by
```