



Page left blank intentionally

**INDEX**

[**1 PURPOSE 1**](#_gjdgxs)

[**2 PROJECT SCOPE 1**](#_30j0zll)

[**3 SYSTEM OVERVIEW 1**](#_1fob9te)

[1. CI/CD Pipeline 1](#_77z81tfpcnr3)

[2. Deployment Module 1](#_juh5bwayulks)

[3. Monitoring and Logging 1](#_dvrnk8rfsnoq)

[4. User Interface 1](#_eor4e23kpr8c)

[**4 DESIGN CONSIDERATIONS 2**](#_3znysh7)

[4.1 Requirements 2](#_2et92p0)

[4.2 Assumptions 2](#_tyjcwt)

[4.3 Dependencies 2](#_3dy6vkm)

[**5 SYSTEM ARCHITECTURE 2**](#_1t3h5sf)

[5.1 Architectural Strategies 4](#_4d34og8)

[1. CI/CD Pipeline Component 4](#_5xy825jg8x48)

[2. Deployment Module 4](#_9qumtvcmn5e5)

[3. Monitoring and Logging Component 4](#_46x34rv4epk5)

[4. User Interface Layer 4](#_ksoek15686aj)

[6. Security Component 4](#_l1zixnrkqmot)

[5.2 Structure & Relationships 5](#_2s8eyo1)

[**6 DETAILED DESCRIPTION OF COMPONENTS 6**](#_17dp8vu)

[**7 INTEGRATIONS 7**](#_3rdcrjn)

[1. GitHub Actions 7](#_5r8iag73uo2m)

[2. Vercel 7](#_kim062roigyy)

[**8 APPENDICES 1**](#_26in1rg)

[8.1 Appendix A – Detailed Description of Components 1](#_lnxbz9)

# **PURPOSE**

This document is created based on the requirement specification document. The purpose of this Software Design Specification (SDS) Document is to break down the project into components to describe in detail what the purpose of each component is and how it will be implemented. The SDS will also serve as a tool for verification and validation of the final product.

# **PROJECT SCOPE**

The scope of the Deploy Fusion includes its distinct features, its benefits, and its limitations. The system's distinct features allow it to automate the deployment process and streamline CI/CD workflows by using GitHub Actions for automation and Vercel for deployment hosting.

The system enables the user to reduce manual errors in the deployment process and accelerate release cycles, thereby enhancing overall development efficiency.

# **SYSTEM OVERVIEW**

This section will provide an outline of the various components and subsystems of Deploy Fusion.

### 1. CI/CD Pipeline

* **GitHub Actions:**
  + Automates the entire software development workflow, including building, testing, and deployment.
  + Triggers workflows based on events such as code commits, pull requests, and issue updates.

### 2. Deployment Module

* **Vercel Integration:**
  + Facilitates automatic deployment of web applications directly to the Vercel platform.
  + Supports multiple environments (development, staging, production) for better testing and rollout.
* **Rollback Mechanism:**
  + Allows for immediate restoration to the last stable version in case of deployment issues.
  + Reduces downtime and ensures application reliability.

### 3. Monitoring and Logging

* **Monitoring Tools:**
  + Provides real-time tracking of application performance, uptime, and error rates.
  + Sends alerts for any deployment failures or critical performance issues.
* **Logging System:**
  + Captures detailed logs during deployment processes and application runtime.

### 4. User Interface

* **Dashboard:**
  + A centralized web-based interface that displays deployment statuses, logs, and performance metrics.
  + Provides intuitive navigation for users to manage deployments and monitor system health.
* **Configuration Management:**
  + Enables users to adjust settings related to CI/CD pipelines, deployment parameters, and monitoring preferences.
  + Simplifies the customization of workflows according to project needs.

# **DESIGN CONSIDERATIONS**

This section describes requirements, assumptions and dependencies to be addressed to devise a complete design solution.

## Requirements

**Automated Deployment:** The system must support automated deployment processes to Vercel.

**CI/CD Integration:** Integration with GitHub Actions for automated builds and tests.

**Monitoring and Alerts:** Real-time monitoring of application performance with alert notifications for failures.

**User Interface:** A user-friendly dashboard for managing deployments and monitoring status.

**Rollback Capability:** Ability to quickly revert to the last stable deployment in case of issues.

## Assumptions

* Users have a basic understanding of version control and CI/CD concepts.
* The system will be used primarily by development teams familiar with GitHub and Vercel.
* Network connectivity is stable to ensure seamless integration with cloud services.
* Required permissions and access rights for GitHub and Vercel will be managed adequately.

## Dependencies

**GitHub Actions:** Dependence on GitHub for version control and CI/CD automation.

**Vercel:** The deployment module is dependent on Vercel's services for hosting applications.

**Monitoring Tools:** Reliance on third-party monitoring tools for application performance tracking.

**Team Collaboration Tools:** Integration with tools for communication and project management to facilitate teamwork.

# **SYSTEM ARCHITECTURE**

The software system architecture refers to the logical organization of a distributed system into software components. It defines how components of a software system are assembled, their relationship and communication between them. It serves as a blueprint for software application and development basis for the developer team. An effective architecture serves as the conceptual glue that holds every phase of the project together for all of its stakeholders, enabling agility, time and cost savings, and early identification of design risks.

The Software architecture:

* Defines structure of a system
* Defines behaviour of a system
* Defines component relationship
* Defines communication structure
* Balances stakeholder’s needs
* Influences team structure
* Focuses on significant elements
* Captures early design decisions

Below some important characteristics which are commonly considered are explained.

**Operational Architecture Characteristics:**

* Availability
* Performance
* Reliability
* Low fault tolerance
* Scalability

**Structural Architecture Characteristics:**

* Configurability
* Extensibility
* Supportability
* Portability
* Maintainability

**Cross-Cutting Architecture Characteristics:**

* Accessibility
* Security
* Usability
* Privacy
* Feasibility

## Architectural Strategies

### 1. CI/CD Pipeline Component

* **GitHub Actions:**
  + Responsible for automating the workflow from code commit to deployment.
  + Configured to run tests, build the application, and trigger deployments based on specific events.

### 2. Deployment Module

* **Vercel Integration:**
  + Handles the deployment of web applications to the Vercel platform.
  + Manages different environments (development, staging, production) to ensure proper testing and rollout.

### 3. Monitoring and Logging Component

* **Monitoring Tools:**
  + Continuously tracks application performance metrics, such as uptime, response time, and error rates.
  + Sends alerts and notifications for any issues detected during deployment or application runtime.
* **Logging System:**
  + Captures logs related to deployment activities and application events.
  + Provides a central repository for troubleshooting and performance analysis.

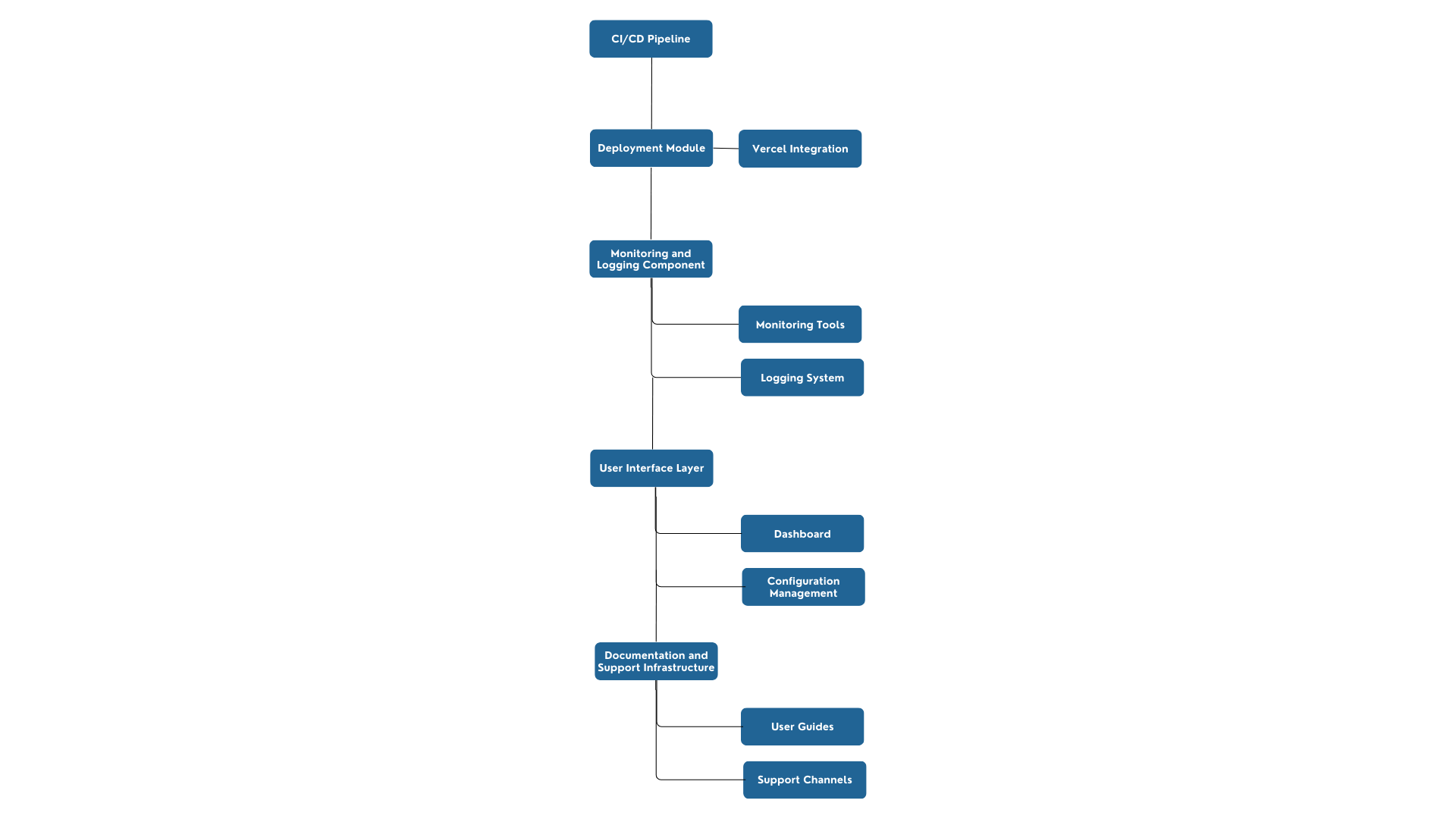
### 4. User Interface Layer

* **Dashboard:**
  + Web-based interface for users to monitor deployment statuses and application health.
  + Provides real-time insights into logs, performance metrics, and system alerts.
* **Configuration Management:**
  + Allows users to customize settings for the CI/CD pipeline, deployment processes, and monitoring preferences.

### 6. Security Component

* **Authentication and Authorization:**
  + Ensures secure access to the CI/CD pipeline and deployment processes.
  + Manages user roles and permissions for GitHub and Vercel integrations.

## Structure & Relationships



# **DETAILED DESCRIPTION OF COMPONENTS**

For detailed description of the components, please refer **Appendix A – Detailed Description of Components**

# **INTEGRATIONS**

### 1. GitHub Actions

* **Purpose:** Automates the CI/CD pipeline for the application.
* **Integration Details:**
  + **Workflow Automation:** GitHub Actions will be configured to automate the build, test, and deployment processes. This includes defining workflows triggered by events such as code commits and pull requests.
  + **Actions:** Utilize pre-built actions for common tasks (e.g., running tests, linting code) to streamline the development process.
  + **Secrets Management:** Sensitive information (like API keys) will be stored in GitHub Secrets for secure access during the build and deployment processes.

### 2. Vercel

* **Purpose:** Hosts and deploys the web applications.
* **Integration Details:**
  + **Deployment Management:** The deployment module will use Vercel's API to manage deployments directly from the GitHub repository. This includes deploying to different environments (development, staging, production).
  + **Real-time Updates:** Vercel provides automatic updates for the application when new code is pushed to the specified branches in GitHub, ensuring the latest version is always live.
  + **Environment Variables:** Vercel allows configuration of environment variables to manage settings for different deployment stages without exposing sensitive information.

# **APPENDICES**

## Appendix A – Detailed Description of Components

| **Identification** | **CI/CD Pipeline** |
| --- | --- |
| **Type** | Class/Form/ |
| **Purpose** | Automates the integration and deployment of code changes to ensure continuous delivery of the web application. This component manages the entire workflow from code commit to production deployment. |
| **Subordinates** | GitHub Actions |
| **Dependencies** | Access to the GitHub repository for version control and workflow definitions.  Configuration files (.yml) for defining workflows. |
| **Interfaces** | Interfaces with the GitHub API for triggering workflows and managing deployments. |
| **Resources** | Requires access to environment variables for secure data handling (e.g., API keys, secrets). |
| **Processing** | Handles the automation of:   * Building the application. * Running tests (unit, integration). * Deploying the application to the hosting environment (Vercel). |
| **Data** | Code changes (commits).  Test results.  Deployment logs and status updates. |

| **Identification** | **Deployment Module** |
| --- | --- |
| **Type** | Class/Form/ |
| **Purpose** | Manages the deployment of web applications to the Vercel platform, facilitating seamless updates and rollbacks as needed. |
| **Subordinates** | Vercel Integration |
| **Dependencies** | Requires Vercel account and project setup for deployment management.  Access to environment configurations for different deployment environments (development, staging, production). |
| **Interfaces** | Interfaces with the Vercel API to handle deployment processes, including triggering new deployments and managing project settings. |
| **Resources** | Requires configuration settings for environment variables and deployment parameters specific to each environment. |
| **Processing** | Handles the following processes:   * Deploying the application based on triggers from the CI/CD pipeline. * Managing environment settings for different deployment stages. * Performing rollbacks if a deployment fails. |
| **Data** | * Deployment configurations (e.g., branch to deploy, environment settings). * Deployment logs and status reports from Vercel. |