

Project Management Plan

Website deployment with Kubernetes

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |

List of Figures

**No table of figures entries found.**

Table of Contents

1. [Terms & abbreviation 3](#_Toc151660047)
2. [1. Project description 4](#_Toc151660048)

[1.1 Context 4](#_Toc151660049)

[1.2 Project goal 4](#_Toc151660050)

[1.3 Project scope 4](#_Toc151660051)

[1.4 Research questions 4](#_Toc151660052)

[1.5 Products & deliverables 5](#_Toc151660053)

1. [2. Project Organisation 6](#_Toc151660054)

[2.1 Stakeholders and team roles & responsibilities 6](#_Toc151660055)

[2.2 Communication 6](#_Toc151660056)

1. [3. Activities and time plan 7](#_Toc151660057)

[3.1 Phase of the project 7](#_Toc151660058)

[3.2 Milestones 8](#_Toc151660059)

1. [4. Risk management 8](#_Toc151660060)
2. [5. Configuration management 9](#_Toc151660061)
3. [Reference 10](#_Toc151660062)

Document history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Author | Date | Description | Status |
| 0.1 | Stefan Dulgerov | 23/11/2023 | First Iteration of individual project plan. | Previous. |
| 0.2 | Stefan Dulgerov | 28/11/2023 | Second Iteration of individual project plan after feedback. | Current. |

# Terms & abbreviation

|  |  |
| --- | --- |
|  |  |
| SDD | System Design Document |
|  |  |

# Project description

## Context

The developer of a website, Mr. S. wants his game to be more fail-safe when he deploys it on a server. To do this he needs to use Kubernetes because of:

-Scalability: Kubernetes can automatically scale the number of servers up or down based on demand. This ensures that users always have access to a server, even during peak periods.

-Reliability: Kubernetes can self-heal from failures. This means that if a server crashes, Kubernetes will automatically restart it. This ensures that users are not interrupted during use.

-Cost-effectiveness: Kubernetes can help to reduce the cost of running the server by using resources more efficiently.

## Project goal

The goal of this project is to deploy a website on a Kubernetes cluster.

## Project scope

|  |  |
| --- | --- |
| **Inside scope:** | **Outside scope:** |
| Containerizing the website application. | 1. **Developing a new website** |
| Creating a Kubernetes deployment for the website application in Azure. | 1. **Creating a custom Kubernetes distribution** |
| Configuring Kubernetes to automatically scale the website application based on demand. | 1. **Integrating the website with other services** |
| Configuring Kubernetes to self-heal from server crashes. | 1. **Porting the website to a new platform** |
| Monitoring the Kubernetes cluster and website application performance. | 1. **Translating the website into a new language** |

## Research questions

**Main Research question:**

How can a website application be deployed on a Kubernetes cluster in a way that is scalable, reliable, and cost-effective?

**Sub-research questions:**

What is the best way to containerize the website application?

Action: Library/Competitive analysis

Output:

Research different containerization tools, such as Docker and Podman.

Evaluate the pros and cons of each tool.

Select the tool that is best suited for the needs of the website application.

What Kubernetes deployment strategy should be used for the website application?

Action: Library/Literature study

Output:

Review different Kubernetes deployment strategies, such as Rolling Update and Blue/Green Deployment.

Understand the pros and cons of each strategy.

Select the strategy that is best suited for the needs of the website application.

How can Kubernetes be configured to automatically scale the website application based on demand?

Action: Workshop/ Decomposition, IT architecture sketching

Output:

Design a high-level IT architecture that shows how the website application will be scaled.

Identify the components and technologies that will be used to achieve auto-scaling.

Develop a plan for implementing the auto-scaling solution.

How can Kubernetes be configured to self-heal from server crashes?

Action: Workshop/Decomposition, IT architecture sketching

Output:

Break down the website application into its components and identify the potential failure points.

Design a self-healing mechanism that can detect and recover from failures.

Implement the self-healing mechanism in Kubernetes.

What are the best tools and metrics for monitoring the Kubernetes cluster and website application performance?

Action: Library/Literature study

Output:

Research different monitoring tools, such as Prometheus and Grafana.

Identify the metrics that are relevant to monitoring the performance of the Kubernetes cluster and website application.

Develop a monitoring plan that includes the selection of tools, metrics, and dashboards.

## Products & deliverables

A containerized website application.

A Kubernetes deployment for the website application.

A configuration file for Kubernetes that defines how the website application should be scaled and healed.

A monitoring dashboard for the Kubernetes cluster and website application performance.

# Project Organisation

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Abbreviation** | **Role and functions** | **Availability** |
| *Stefan Dulgerov* | *S B D* | Project manager,  Developer,  Kubernetes engineer,  DevOps engineer,  System administrator | *When is the person available for your project (define this in the way most relevant for your project, e.g., which days are available, the amount of time, or in which phase of the project).* |
|  |  |  |  |
|  |  |  |  |

## Stakeholders and team roles & responsibilities

Defines the project scope and objectives, identifies the stakeholders and their roles and responsibilities, develops a communication plan, creates a project timeline, and hosts weekly project meetings to discuss the progress of the project.

Containerizes the website application and provides input on the best way to containerize the game application.

Creates the Kubernetes deployment for the website application and provides input on the best deployment strategy for the website application.

Configures Kubernetes to automatically scale the website application based on demand, configures Kubernetes to self-heal from server crashes, and implements a load testing plan to ensure that the website application can handle the expected demand.

Monitors the Kubernetes cluster and website application performance and implements a monitoring plan for the Kubernetes cluster and website application performance.

Stakeholders other than the developer:

Teachers, coach, clients.

## Communication

Email: Email will be used to communicate with stakeholders on an ad hoc basis.

Teams: Teams will be used to communicate with stakeholders.

In-person meetings with coaches and teachers:

Every Monday there will be a coach meeting and Wednesday, Thursday and Friday I will ask for feedback from my other teachers.

# Activities and time plan

## Phase of the project

Phase 1: Planning (1 week (week 11))

Goal: Define the project scope, objectives, deliverables, and timeline.

Activities:

Conduct a kickoff meeting with all stakeholders to align on website goals and expectations.

Clearly define the project scope, including what is included and excluded from the project.

Identify the specific deliverables that the project will produce, such as a containerized website application, a Kubernetes deployment configuration, and a monitoring dashboard.

Create a detailed project timeline with milestones and deadlines for each task.

Phase 2: Design (1 week (week 12))

Goal: Design the containerization strategy, Kubernetes deployment, and monitoring solution for the website application.

Activities:

Determine the best approach for containerizing the website application.

Design a Kubernetes deployment strategy that can handle the expected demand and scalability requirements of the website.

Select appropriate monitoring tools and metrics to track the performance and health of the Kubernetes cluster and website application.

Document the design decisions and specifications for future reference.

Phase 3: Implementation (2 weeks (week 13&14))

Goal: Containerize the website application, create the Kubernetes deployment, and implement the monitoring solution.

Activities:

Containerize the website application using the chosen containerization tool and methodology.

Create the Kubernetes deployment configuration based on the design specifications.

Implement the selected monitoring tools and dashboards to collect and visualize performance data.

Phase 4: Testing and Deployment (1 week (week 15))

Goal: Perform comprehensive testing and be ready to deploy the website application to the Kubernetes cluster in production.

Activities:

Perform load testing to simulate high user demand and ensure the website application can scale effectively.

Conduct stress testing to identify potential bottlenecks and performance issues under extreme conditions.

Address any bugs or issues discovered during testing and refine the deployment configuration as needed.

Be ready to deploy the website application to the production Kubernetes cluster following a carefully planned rollout strategy.

Phase 5: Handover and Reflection (1 week (week 16))

Goal: Document the project and reflect on lessons learned.

Activities:

Create comprehensive project documentation that includes design decisions, implementation details, and operational procedures.

Create a handover document in case there is a need for handover.

Facilitate a project retrospective session to gather feedback, identify lessons learned, and suggest improvements for future projects.

## Milestones

Milestone 1: Project kickoff and scope definition.

Milestone 2: Deliverable - Project plan with scope, objectives, deliverables, and timeline.

Milestone 3: Deliverable - Containerization strategy design document.

Milestone 4: Deliverable - Kubernetes deployment design document.

Milestone 5: Deliverable - Monitoring solution design document.

Milestone 6: Containerization of the website application completed.

Milestone 7: Kubernetes deployment configuration finalized.

Milestone 8: Monitoring solution implemented and integrated.

Milestone 9: Load testing completed, and performance issues addressed.

Milestone 10: Stress testing completed, and scalability issues identified.

Milestone 11: Deployment plan finalized, and rollout strategy defined.

Milestone 12: Deliverable - Project documentation finalized.

Milestone 13: Project retrospective session conducted, and lessons learned documented.

# Risk management

|  |  |  |  |
| --- | --- | --- | --- |
| Risk | Probability | Impact | Countermeasures |
| Website application may not be easy to containerize | Medium | High | Determine best approach for containerizing the website application |
| Kubernetes deployment may not be able to handle the high-volume, low-latency demands of the website application | Medium | High | Determine best deployment strategy for the website application |
| Website application may not be able to scale to meet demand | Medium | High | Implement load testing plan to ensure website application can handle expected demand |
| Website application may not be able to self-heal from failures | Medium | High | Implement self-healing mechanism for website application |
| Kubernetes cluster may not be able to be monitored effectively | Medium | Medium | Implement monitoring plan for Kubernetes cluster and website application performance |

Risk scores:

|  |  |  |
| --- | --- | --- |
| Website application may not be easy to containerize | 3 | 6 |
| Kubernetes deployment may not be able to handle the high-volume, low latency demands of the game application | 3 | 6 |
| Website application may not be able to scale to meet demand | 3 | 6 |
| Website application may not be able to self-heal from failures | 3 | 6 |
| Kubernetes cluster may not be able to be monitored effectively | 2 | 4 |

**Total Risk Score:** 30

The total risk score is 30, which is considered a high risk. This means that the project is at a high risk of failure if the risks are not properly mitigated. The highest risk areas are containerization, deployment, scalability, self-healing, and monitoring.

# Configuration management

(This will probably change in near future after discussions with teachers and more research.)

The following tools and strategies will be used for configuration management in this project:

Kubectl: Kubectl will be used to configure the Kubernetes cluster.

AKS: AKS will be used to manage Kubernetes deployments.

Azure monitor: Azure monitor will be used to monitor the Kubernetes cluster and game application performance.

GitHub will be used so there is a place to save all of the necessary documentation and deployment files and app configurations.

# Reference

Kubernetes documentation: <https://kubernetes.io/docs/>