



SOFTWARE DESIGN SPECIFICATION

CLOUD COMPUTING  
  
MongoSafenet

|  |  |  |  |
| --- | --- | --- | --- |
| **Created By:** | Piyush Raj Kumar | **Approved By:** | Tanmai Kamat |
| **Created On:** | 22-08-2023 | **Approved On:** | 01-07-2023 |

Page left blank intentionally

**INDEX**

[**1** **PURPOSE** 2](#_Toc142418236)

[**2** **PROJECT SCOPE** 2](#_Toc142418237)

[**3** **SYSTEM OVERVIEW** 2](#_Toc142418238)

[**4** **DESIGN CONSIDERATIONS** 4](#_Toc142418239)

[4.1 Requirements 4](#_Toc142418240)

[4.2 Assumptions 5](#_Toc142418241)

[4.3 Dependencies 5](#_Toc142418242)

[**5** **SYSTEM ARCHITECTURE** 5](#_Toc142418243)

[5.1 Architectural Strategies 7](#_Toc142418244)

[5.2 Structure & Relationships 7](#_Toc142418245)

[**6** **DETAILED DESCRIPTION OF COMPONENTS** 8](#_Toc142418246)

[**7** **INTEGRATION** 9](#_Toc142418247)

[**8** **APPENDICES** 1](#_Toc142418248)

[8.1 Appendix A – Detailed Description of Components 1](#_Toc142418249)

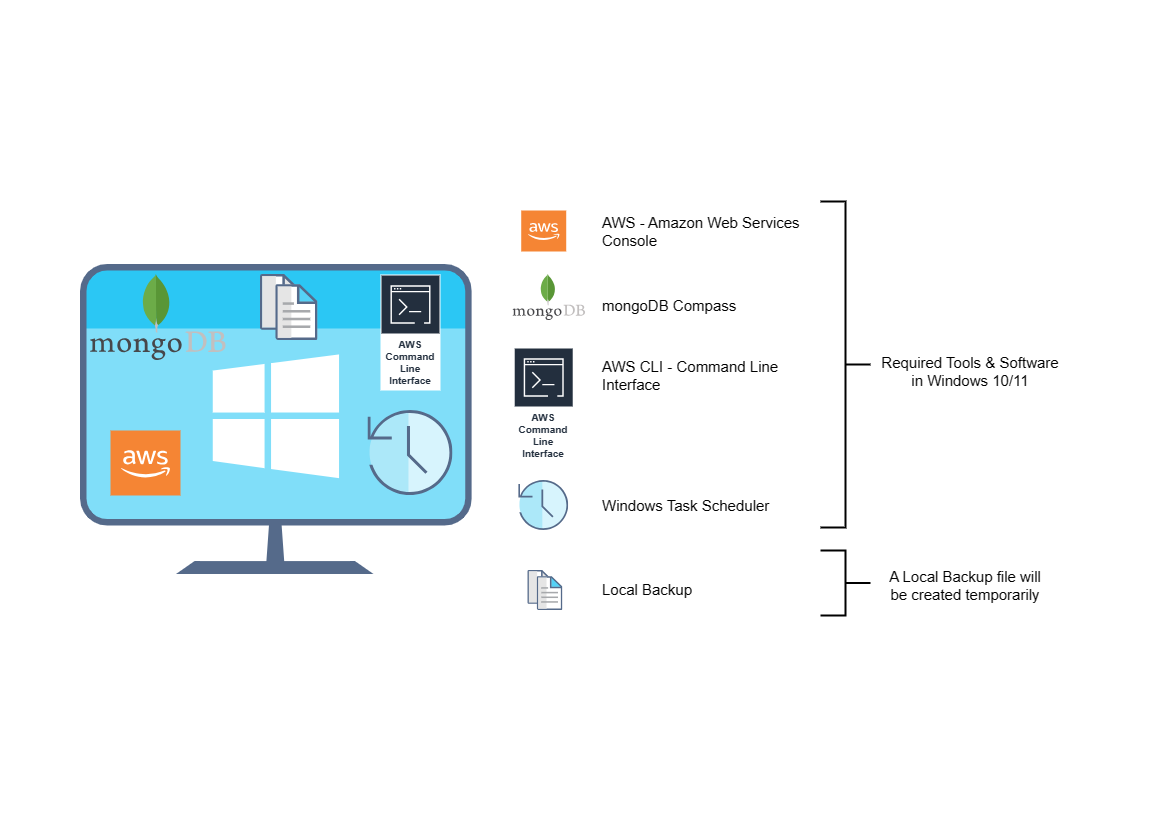
# **PURPOSE**

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 MongoSafenet includes its distinct features, its benefits, and its limitations. The system's distinct features allow it to automate the Continuous Scheduled Backup of a MongoDB database hosted on an a Windows Machine, using Windows Task Scheduler, Mongodb Compass, AWS S3 bucket for storage, and IAM for access control. The system enables the user to eliminate the necessity for manual database backups, resulting in time savings and heightened productivity.

# **SYSTEM OVERVIEW**



This section will provide an outline of the various components and subsystems of **MogoSafenet**:-

3.1 System Architecture

The MongoSafenet system is meticulously architected, boasting a modular and scalable design meticulously crafted to optimize backup and data management. The system comprises the following key components and subsystems:

3.1.1 Backup Scheduler Subsystem

* Purpose: This subsystem is responsible for scheduling and triggering automated MongoDB backups.
* Components: Windows Task Scheduler used as a scheduler.

3.1.2 Backup Process Subsystem

* Purpose: The primary function of this subsystem is to initiate and execute MongoDB backups.
* Components: MongoDB Backup Commands: Utilized for creating database backups.

3.1.3 Data Storage Subsystem

* Purpose: This subsystem is responsible for securely storing backup data.
* Components: AWS S3 Bucket: Utilized for reliable and scalable storage of backup files.

3.1.4 Access Control Subsystem

* Purpose: Ensuring secure access to backup data is the primary goal of this subsystem.
* Components: AWS IAM (Identity and Access Management): Controls and manages access to the - AWS S3 bucket.

3.2 Data Flow

The data flow within the MongoSafenet system is designed for smooth and secure backup operations:

1. The Backup Scheduler Subsystem triggers the Backup Process Subsystem based on scheduled backup intervals.

2. The Backup Process Subsystem initiates MongoDB backup commands to create backup files.

3. The Data Storage Subsystem securely stores the backup file in the designated AWS S3 bucket.

4. The Access Control Subsystem uses AWS IAM to control and manage access to the S3 bucket, ensuring only authorized users can retrieve backups.

3.3 Interfaces

The MongoSafenet system interfaces with the following external components:

* MongoDB Database: The system interacts with the MongoDB database to perform backups.
* AWS S3 Bucket: Used for storing backup files securely.
* Windows Task Scheduler: Utilized for scheduling backup tasks.
* AWS IAM: Manages access control to the S3 bucket.

3.4 Dependencies

The proper functioning of the MongoSafenet system depends on the following dependencies:

* Availability of the MongoDB database.
* Proper configuration of Windows Task Scheduler.
* AWS services, including S3 and IAM, functioning as expected.

3.5 Scalability

The system is designed to be scalable to accommodate potential future requirements, including the addition of more MongoDB databases or the need for increased storage capacity.

3.6 Security

Ensuring the security of data and access control is a priority for the system. AWS IAM is used to restrict unauthorized access to backup data stored in the S3 bucket.

3.7 Reliability

The system's architecture and components are selected for reliability and resilience, minimizing the risk of data loss during backup processes.

# **DESIGN CONSIDERATIONS**

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

## Requirements

The following requirements, as identified in the Software Requirements Specification (SRS) document, are to be considered during the design of the MongoSafenet system:

4.1.1 Automated MongoDB Backup

* The system should take care of automatically backing up the MongoDB database hosted on our Windows Machine.
* Make sure we schedule these backups to happen at times that make sense.

4.1.2 Data Storage

* Backup data shall be securely stored in an AWS S3 bucket.

4.1.3 Access Control

* Access to backup data in the S3 bucket shall be controlled and managed using AWS IAM.

4.1.4 Reliability

* The system shall be designed for reliability, minimizing the risk of data loss during backup processes.

4.1.5 Scalability

* Our system's architecture will be built to grow with us, accommodating future needs like more MongoDB databases or increased storage space.

4.1.6 Security

* Ensuring the security of our data is a top concern. We'll put measures in place to keep unauthorized access away from our backup data.

## Assumptions

The following assumptions, as listed in the Software Requirements Specification (SRS) document, are considered during the design of the MongoSafenet system:

4.2.1 MongoDB Database

* It is assumed that the MongoDB database is installed on Windows otherwise hosted on an AWS EC2 instance if going totally on aws cloud.

4.2.2 Windows Task Scheduler

* Windows Task Scheduler is assumed to be available and properly configured for scheduling backup tasks.

4.2.3 AWS Services

* AWS services, including S3 and IAM, are assumed to be functional and accessible.

## Dependencies

The following dependencies, as listed in the Software Requirements Specification (SRS) document, must be considered during the design of the MongoSafenet system:

4.3.1 MongoDB Database

* The system is dependent on the availability and proper functioning of the MongoDB database.

4.3.2 Windows Task Scheduler

* Proper configuration of Windows Task Scheduler is a dependency for scheduling backup tasks.

4.3.3 AWS Services

* AWS services, specifically S3 for storage and IAM for access control, are dependencies for the system.

# **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 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

The following are the major components of the system architecture for the MongoSafenet project:

5.1.1 Backup Scheduler Module

* What it does: Responsible for scheduling and triggering automated MongoDB backups.
* How it works: Utilizes Windows Task Scheduler.

5.1.2 Backup Process Module

* What it does: Initiates and executes MongoDB backups.
* How it works: Utilizes MongoDB backup commands.

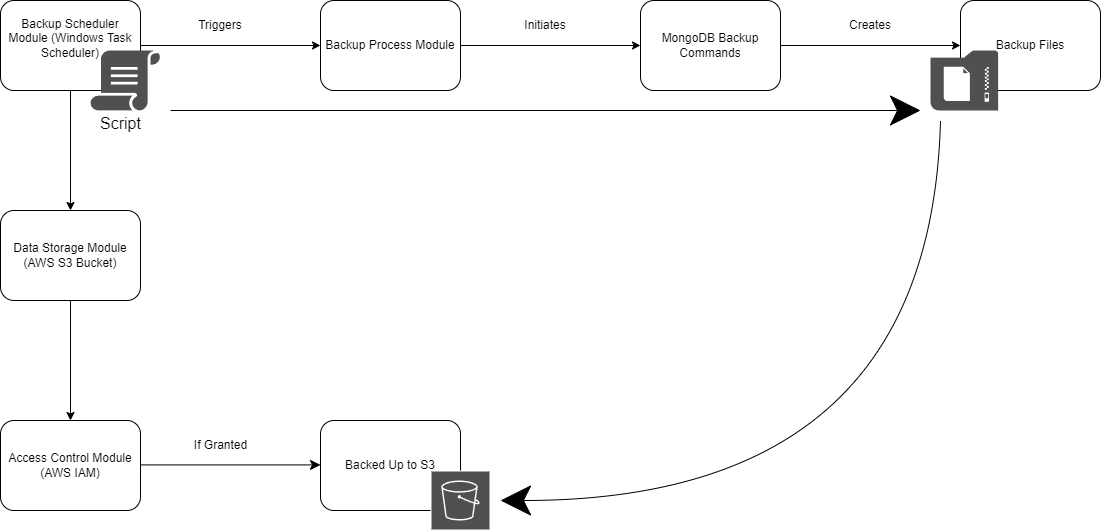
5.1.3 Data Storage Module

* What it does: Handles the storage of backup data.
* How it works: Utilizes AWS S3 bucket for storage.

5.1.4 Access Control Module

* What it does: Ensures secure access to backup data.
* How it works: Utilizes AWS IAM for access control.

## Structure & Relationships



# 

# **DETAILED DESCRIPTION OF COMPONENTS**

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

The below template will be used to specify the details of all the components

**Table 1: Detailed Design Specification Template**

|  |  |
| --- | --- |
| **Identification** | The unique name for the component and the location of the component in the system. |
| **Type** | A module, a subprogram, a form, a data file, a control procedure, a class, etc. |
| **Purpose** | Function and performance requirements implemented by the design component, including derived requirements. Derived requirements are not explicitly stated in the SRS - but are implied or adjunct to formally stated SDS requirements. |
| **Subordinates** | The internal structure of the component, the constituents of the component, and the functional requirements satisfied by each part. |
| **Dependencies** | How the component’s function and performance relate to other components. How this component is used by other components. The other components that use this component. Interaction details such as timing, interaction conditions (such as order of execution and data sharing), and responsibility for creation, duplication, use, storage, and elimination of components. |
| **Interfaces** | Detailed description of all external or internal interfaces as well as of any mechanism for communicating through messages, parameters, or common data areas. All error messages and error codes should be identified. All screen formats, interactive messages, and other user interface components (originally defined in the SRS) should be given here. |
| **Resources** | A complete description of all resources (hardware or software) external to the component but required to carry out its functions. |
| **Processing** | A full description of the functions presented in the Function subsection. Pseudocode can be used to document algorithms, equations, and logic. |
| **Data** | For the data internal to the component, describes the representation method, initial values, use, semantics, and format. |

# **INTEGRATIONS**

The MongoSafenet system establishes crucial collaborations with various components, applications, and tools to achieve its objectives. Below are the details of the integrations required:

7.1 MongoDB Integration

* Objective: The MongoSafenet system integrates with the MongoDB database hosted on a Windows Machine to perform automated backups.
* Implementation: MongoDB backup commands are used to interact with the MongoDB database and create backup files.

7.2 AWS S3 Integration

* Objective: To securely store backup data, the system integrates with AWS S3, an object storage service.
* Implementation: Backup files are transferred and stored in an AWS S3 bucket.

7.3 AWS IAM Integration

* Objective: To ensure access control and security of backup data in the S3 bucket, the system integrates with AWS IAM.
* Implementation: IAM policies and roles are defined to manage access to the S3 bucket.

7.4 Windows Task Scheduler Integration

* Objective: The Backup Scheduler Module utilizes Windows Task Scheduler to trigger automated MongoDB backups at scheduled intervals.
* Implementation: Scheduled backup tasks are managed and triggered through Windows Task Scheduler.

7.5 Configuring Aws Command Line Interface

* Objective: To connect Windows Machine remotely to access Aws S3 Bucket to upload backups.
* Implementation: Installing Aws Cli & then configuring it with IAM credentials and ultimately gets access to the S3 bucket.

# **APPENDICES**

## Appendix A – Detailed Description of Components

|  |  |
| --- | --- |
| **Identification** | **Backup Scheduler** |
| **Type** | Module |
| **Purpose** | The Backup Scheduler Module is responsible for scheduling and triggering automated backups. |
| **Subordinates** | N/A |
| **Dependencies** | Windows Task Scheduler for Windows,  Or Crons for Ubuntu. |
| **Interfaces** | Windows Task Scheduler,  Aws Command Line Interface,  MongoDb Compass |
| **Resources** | N/A |
| **Processing** | Manages Scheduled tasks to backup according the script given to it, which in this case commands to upload backups on AWS S3 bucket. |
| **Data** | Connection String Added already in Script |

|  |  |
| --- | --- |
| **Identification** | **Backup Process** |
| **Type** | Module |
| **Purpose** | The Backup Process Module is responsible for initiating and executing MongoDB backups. |
| **Subordinates** | None |
| **Dependencies** | Relied on MongoDB backup commands for the database backup process. |
| **Interfaces** | This module interact with MongoDB to create backup files. |
| **Resources** | None |
| **Processing** | The module triggers MongoDB backup commands and oversees the entire backup process. |
| **Data** | The module processes data, including the backup files generated during the backup operation. |