Database Management System

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 09/12/2021 | 1.0 | Initial release | Do Minh Vuong |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 2

1.1 Purpose 2

1.2 Scope 2

1.3 Definitions, Acronyms, and Abbreviations 2

1.4 References 2

1.5 Overview 2

2. Architectural Representation 2

3. Architectural Goals and Constraints 2

4. Use-Case View 2

4.1 Use-Case Realizations 2

5. Logical View 2

5.1 Overview 2

5.2 Architecturally Significant Design Packages 2

6. Process View 2

7. Deployment View 2

8. Implementation View 2

8.1 Overview 2

8.2 Layers 2

9. Data View (optional) 2

10. Size and Performance 2

11. Quality 2

# Introduction

“Database Management System (DMS)” is a platform to manage databases via cPanel. It makes easier for customers to manipulate MySQL databases to create websites.

## Purpose

This document provides a comprehensive architectural overview of the system, using several different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

## Scope

Under the static behavior of the system, the document discusses the class diagrams and other static architecture designs. Dynamic aspects of the system are elaborated using use case realizations and system sequence diagrams.

## Definitions, Acronyms, and Abbreviations

DMS – Database management system (project)

OOP – Object oriented programming

MVC – Model view control architecture

## References

No required.

## Overview

The report will present a detailed analysis of the architecture of DMS. Further sections cover the architectural representation of the project including architectural representation, architectural goals and constraints and use case realizations.

# Architectural Representation

No required.

# Architectural Goals and Constraints

## Server side

DMS will be hosted at web service server. Being a web-based application, this underlying client OS can be any PC operating system. (Windows, Linux, Apple). MySQL will be used as the central database server. All communication with client must comply with public HTTPS, TCP/IP communication protocol standards.

## Client Side

Users will be able to access DMS only online and restricted host.

## Security

Central security features are handled by the institute officials. They’ll be given full access features both in the application and database levels.

## Persistence

All the data will be saved in the central server. This is a rational database that implements the 3rd normal form. (MySQL).

## Reliability / Availability

The system will be subjected to several testing operations (Unit testing, integration testing, system testing) before being deployed to make sure that the system is reliable. The MYSQL database server can respond to many numbers of clients at a given moment without losing consistency and data integrity.

## Performance

The system responds to any request under standard database and web server script timeouts (3 seconds), also system performance can depend on available hardware, network, and internet connection capabilities. Especially the statistical information generation tasks may take comparatively high time. Therefore, actual performance can be determined only after the system is deployed and tested.

## Portability and reuse

DMS is now a mature system and is not scalable.

## Development tools

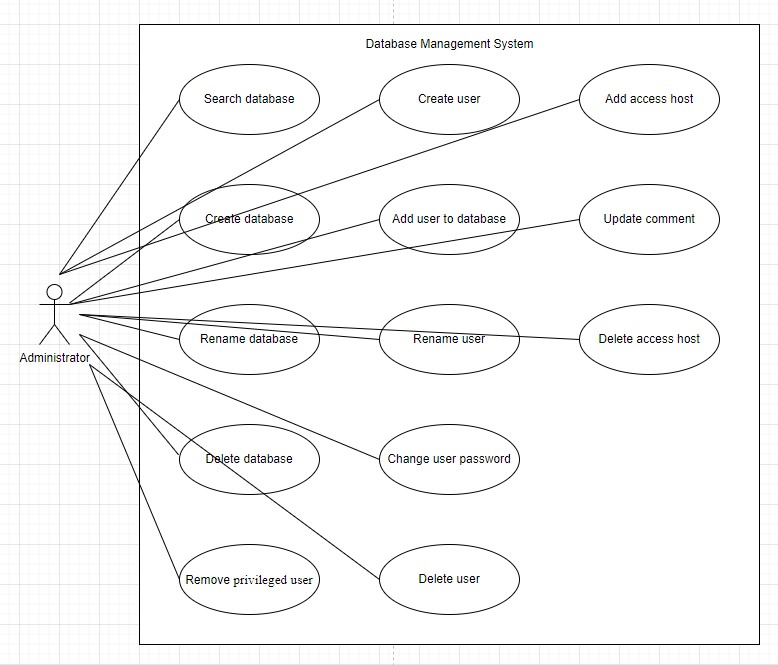
The project incorporates many development tools.

* Interface: cPanel
* Diagrams: Draw.io
* Database: MYSQL connector

# Use-Case View

This system is only managed by the customer as the administrator.

## Use-Case Realizations



### Search database

This use-case occurs when the user search for existing database in MySQL server. It will return a table with the databases found which match to keyword or an empty table.

### Create database

This use-case occurs when the user wants to create a new database which is not exist in MySQL server. It returns a success message and database list will be updated again.

### Rename database

This use-case occurs when the user wants to rename a database in MySQL server. It returns a success message and database list will be updated again.

### Delete database

This use-case occurs when the user wants to remove a database in MySQL server. Before starting the system will ask the user to confirm the action, then it returns a success message and database list will be updated again.

### Remove privileged user

This use-case occurs when the user wants to remove a user to from the database in MySQL server. Before starting the system will ask the user to confirm the action, then it returns a success message and database list will be updated again.

### Create user

This use-case occurs when the user wants to create a new user which is not exist in MySQL server. It returns a success message and user list will be updated again.

### Add user to database

This use-case occurs when the user wants to add a user to the database in MySQL server. It returns a success message and database with user list will be updated again.

### Rename user

This use-case occurs when the user wants to rename a user in MySQL server. It returns a success message and user list will be updated again.

### Change user password

This use-case occurs when the user wants to edit a username in MySQL server. It returns a success message and database list will be updated again.

### Delete user

This use-case occurs when the user wants to remove a user in MySQL server. Before starting the system will ask the user to confirm the action, then it returns a success message and user list will be updated again.

### Add access host

This use-case occurs when the user wants to add a new host for accessing to MySQL server. It returns a success message and access host list will be updated again.

### Update comment

This use-case occurs when the user wants to edit a comment related to the access host. It returns a success message, and the host comment will be updated again.

### Delete access host

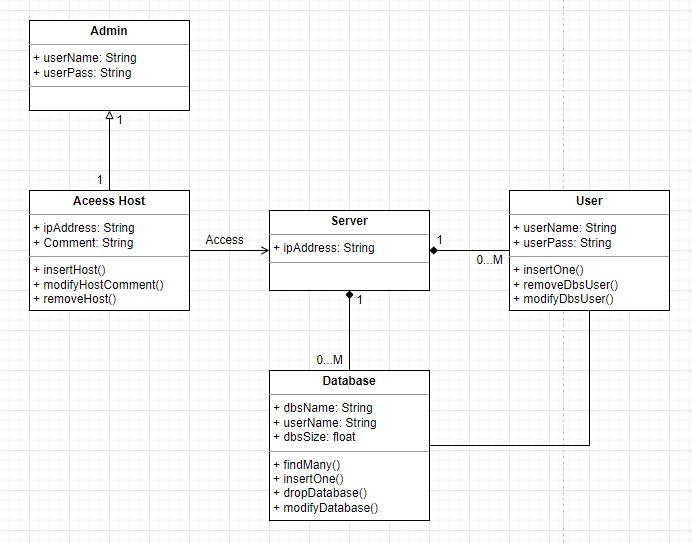
This use-case occurs when the user wants to remove an access host from accessing to MySQL server. Before starting the system will ask the user to confirm the action, then it returns a success message, and the access host list will be updated again.

# Logical View

## Overview

A description of the logical view of the architecture. Describes the overall decomposition of the design model in terms of package hierarchy and layers.

## Architecturally Significant Design Packages



# Implementation View

The implementation of the system is strictly driven from the design; therefore, the implementation view will not be considered in this document.

# Deployment View

Being a web application, this system is hosted in a remote server. The database will be hosted in MySQL service server. All the processing is done in the backend, so that the client computer doesn’t have to spend much of CPU power.

# Implementation View

## Overview

Client Layer (Browser) – Business Layer (Web server) – Data Layer (Database)

DMS is a web application that follows the MVC architecture pattern. Main reason to use this pattern is to separate functions into layers thus improve the maintainability and reusability.

## Layers

No required.

# Data View (optional)

No required.

# Size and Performance

The selected software architecture supports major timing and sizing requirements, as specified in the Supplementary Specification.

* The system will provide access to the user's database and web page for no more than 2 seconds.
* The system develops and offers two options: web version and mobile application version. For mobile apps, the minimum system requirement is 250MB of RAM and more than 20MB of available memory.

# Quality

Architectural goals and constrains topic cover the quality parameters for the OES (online examination system).