High Availability (HA) Distributed File Storage

Document: Software Requirements Specification

Version 1.1

Publication date: 25-April-2016

Group name: Gryffindor

Group Members:

- ARJITH CHALASANI
- JITENDRA NEELAM
- URMILA JYOTHULA
- CHINNA BALAJI YALLA
- NITISH NAGABHAIRAVA
- PRANEEL REDDY PADALA
- SURYA TEJA BOLLIMPALLI
- SRI GANESH SAI GUNNAM
- SRI KASYAP KAPPAGANTULA
- SIRISHA MANASWINI BHAMIDI
- RAGHUVINAYAK RAO MEDISETTI
- MAHAMMAD SUHAIL ATCHUKATLA

Contents:

1. Preface

The main concept of this project is to develop a secure file storage to the company SecuriFile in the form of a distributed file storage system with high availability to the customers.

When a user uploads a file, the file is stored in a randomly chosen server. In this we are creating replicas for the file uploaded by the user and we use File Transfer Protocol for the transfer of data.

Service Developer: Gryffindor

Customer: Dragos llie

In this document we defined the technical terms and a short note on them, system architecture, user and system requirements and references.

- **Release v1.1 on 2014-04-25**

  Preface changed

  System Architecture diagram modified

  Added replicas connection

  Modifications done in User and System requirements

- **Release v1.0 on 2014-04-18**

  Initial Release

2. Glossary and abbreviations

- **HTTP: Hyper Text Transfer Protocol**

  It is a transfer of version data formats between server and client

  EX: plain txt, hyper txt, video and sound

- **FTPS: File Transfer Protocol Security**

  It is an extension for commonly used file transfer protocol(FTP) that adds support for the transfer layer security(TLS) and secure sockets layer (SSL).

- **Message digest: SHA-1**

  IT is a crypto graphic hash function which is consider practically impossible to invert that is to recreate the input data from its hash value alone.

  SHA-1: secure Hash algorithm.SHA-1 produces a 160bit (20 byte) hash value known as a message digest. SHA-1 advancements are SHA-2 and SHA-3.

- **GUI: Graphical User Interface**

  It is a type of interface which helps in interaction with electronic devices through graphical icon and visual indicators.

- **SQL Server: Structured Query Language Server**

  SQL is used to store, query and manipulate data. It is used for manage data in a relational data base.

- **Restful API: Representation State Transfer**

  An architectural pattern to improve probability and scalability of a system.

3. System architecture

In this section of the proposal we provide the system architecture, which determines the working of the system. Initially the system can be determined in three sections they are the front end, database and the back end of the system.

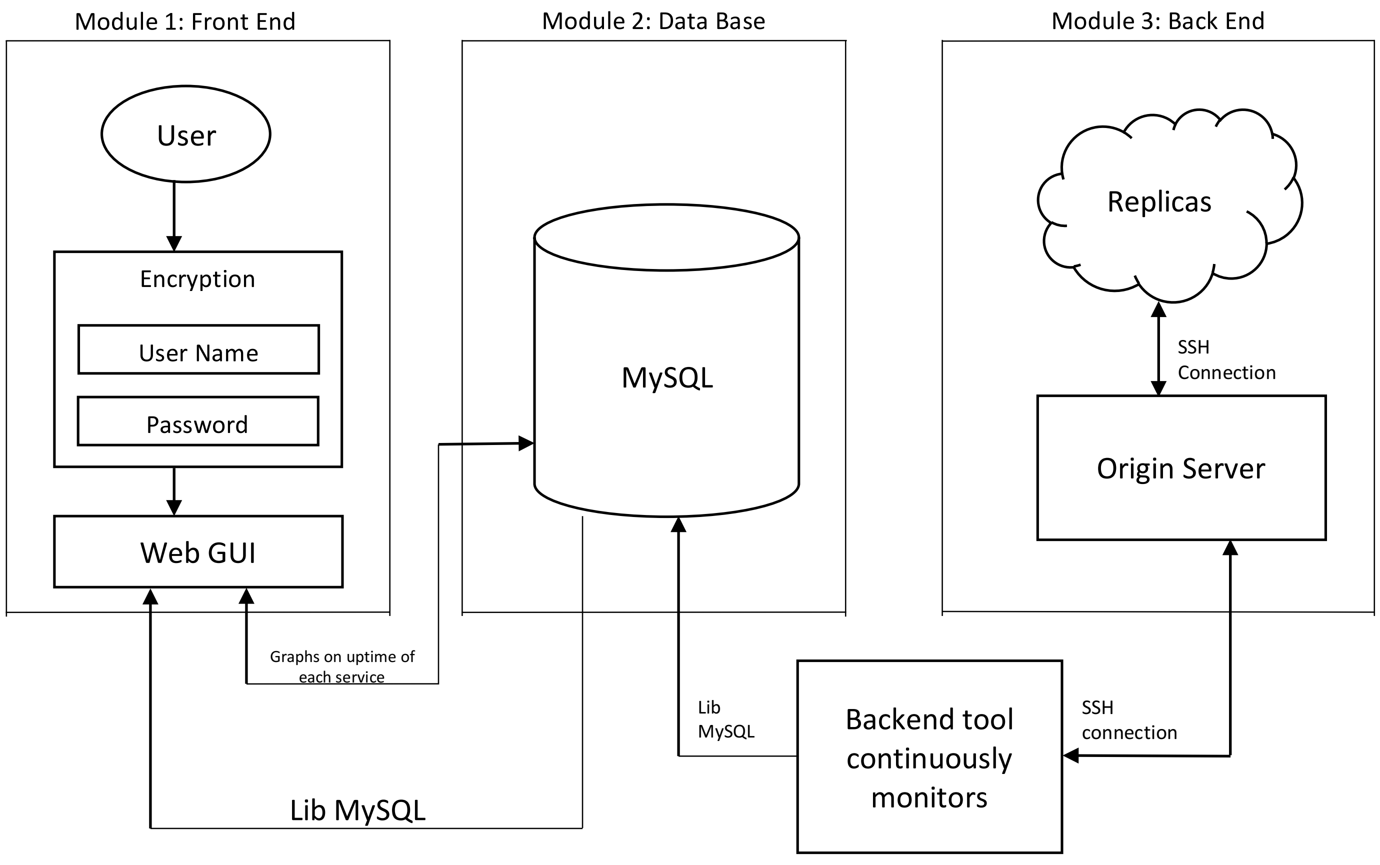
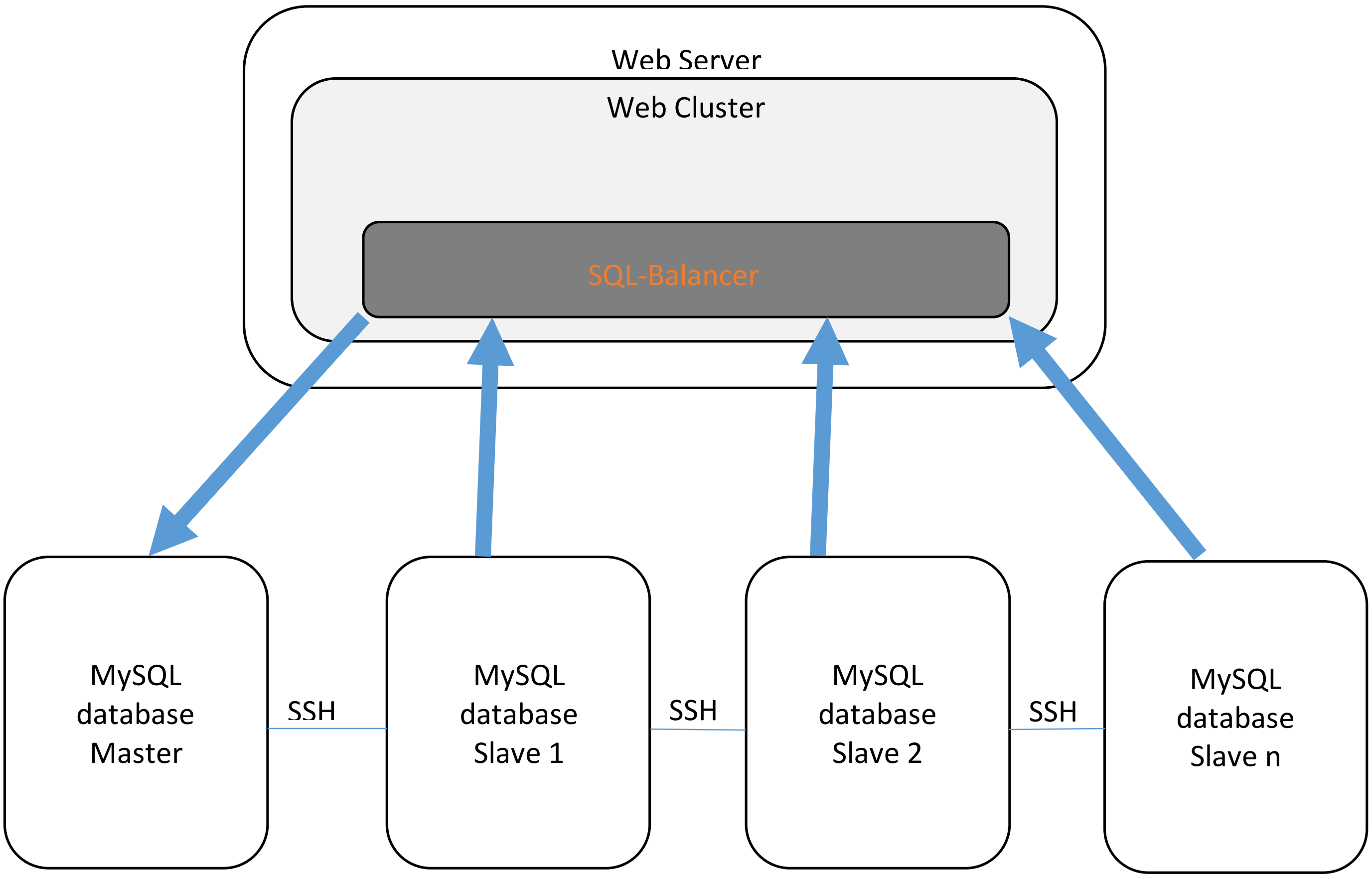


Fig 1: System Architecture

Replication Connection:



Cluster:

1) Cluster and users are connected and registered with SSL connection.
2) Cluster connects the master users to the master database and for the slave users slave database is randomly selected.
3) The cluster is connected to master and slave using SSH connection.

Master and Slave:

1) Master database can be accessed only by the master users.
2) Master database also consists the files present in the slave database.
3) Slaves and masters are connected with SSH connection.

3.1 Frontend

At first the user is asked to login, he gets his access from database and this is done through a series of process through HTML and CSS. We create a login page to login into the server but first he needs to register into the admin server, the admin server provides a verification mail to the user and later conformation of the mail is done and the connections for the web pages are done through PHP. Now a separate account is allocated to the customer through which user can store files. Through the login page the customer logins into the user's account and user can upload the file and make modifications to the existing file. The storage capacity is limited to the customer and users can't exceed the given storage capacity, later he can logout from the page.

3.2 Database

The MySQL database contains user's information, data status and uptime information inserted into their respective service tables
In the MySQL database we create a data slot for each customer. Through front end the user can access into user's login account and through backend the user can access user's files.

3.3 Backend

The third section of the architecture is used to connect the entire architecture. For backend programming we use Python, which is used to connect the servers. Backend is also use to ping the servers constantly, if the ping is lost or server is down it sends a message to database and in the database it is stored if the connection is lost.
It retrieves the status and stores it in the database in MySQL. Separate tables are allocated for each server.

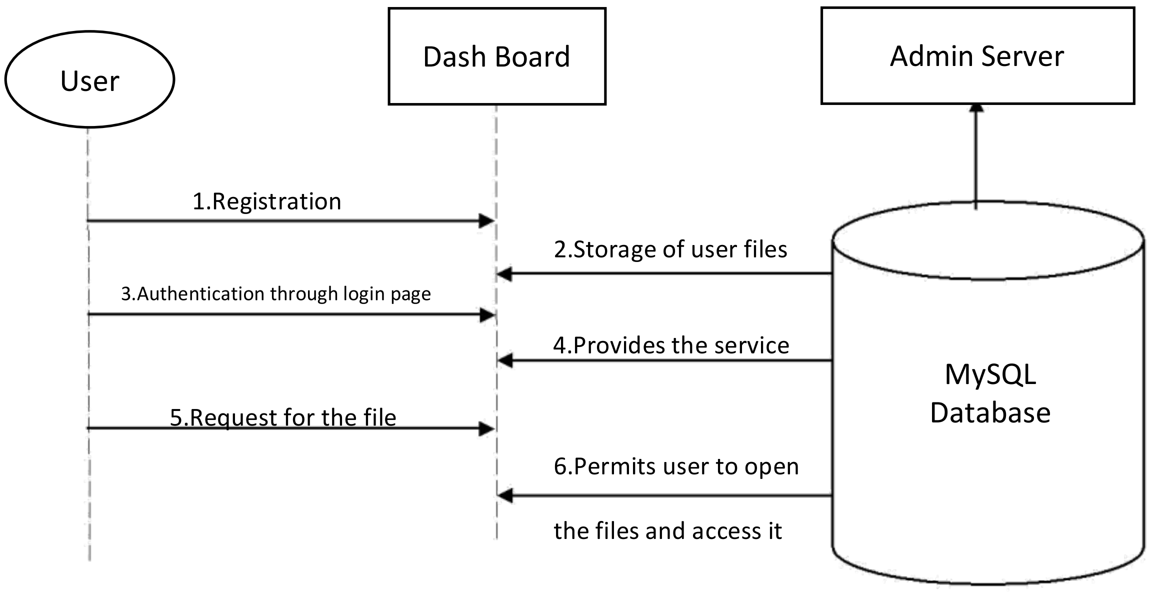


Fig 2: Front End

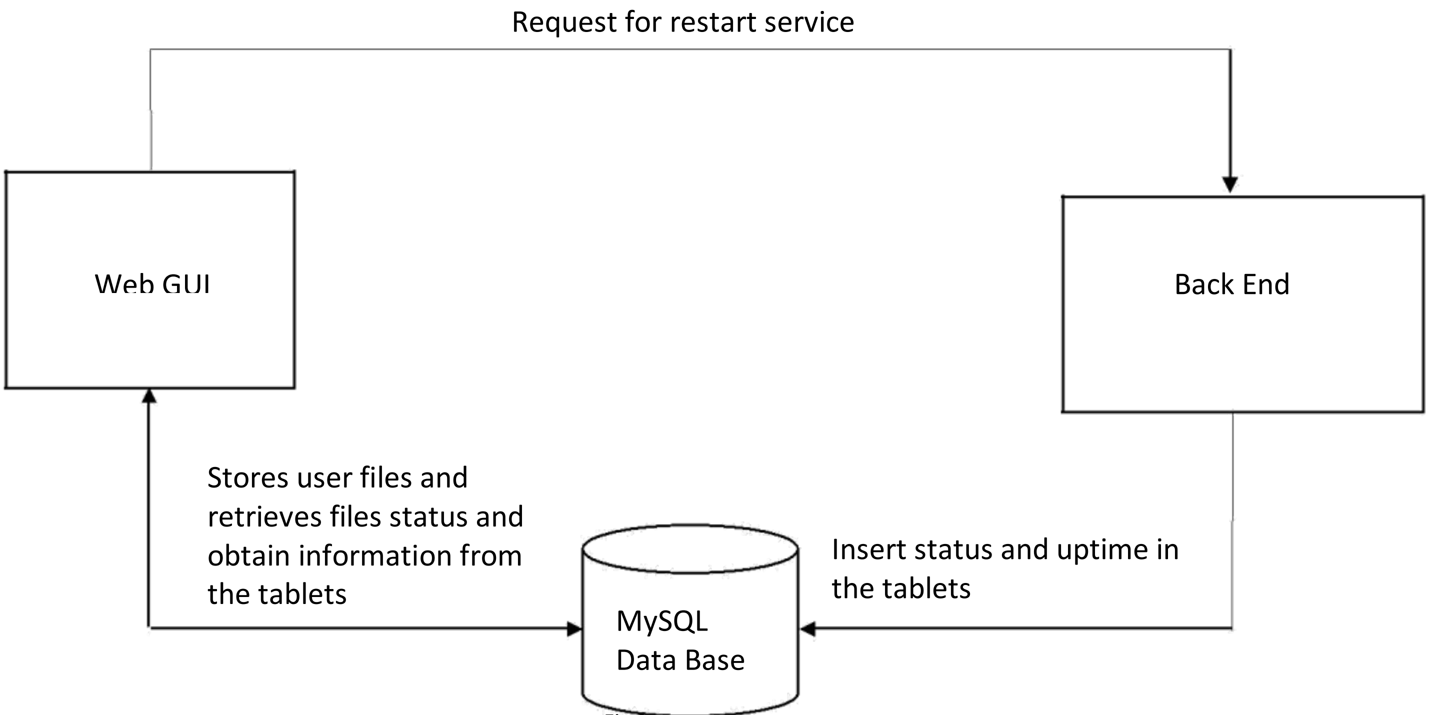


Fig 3: Data Base

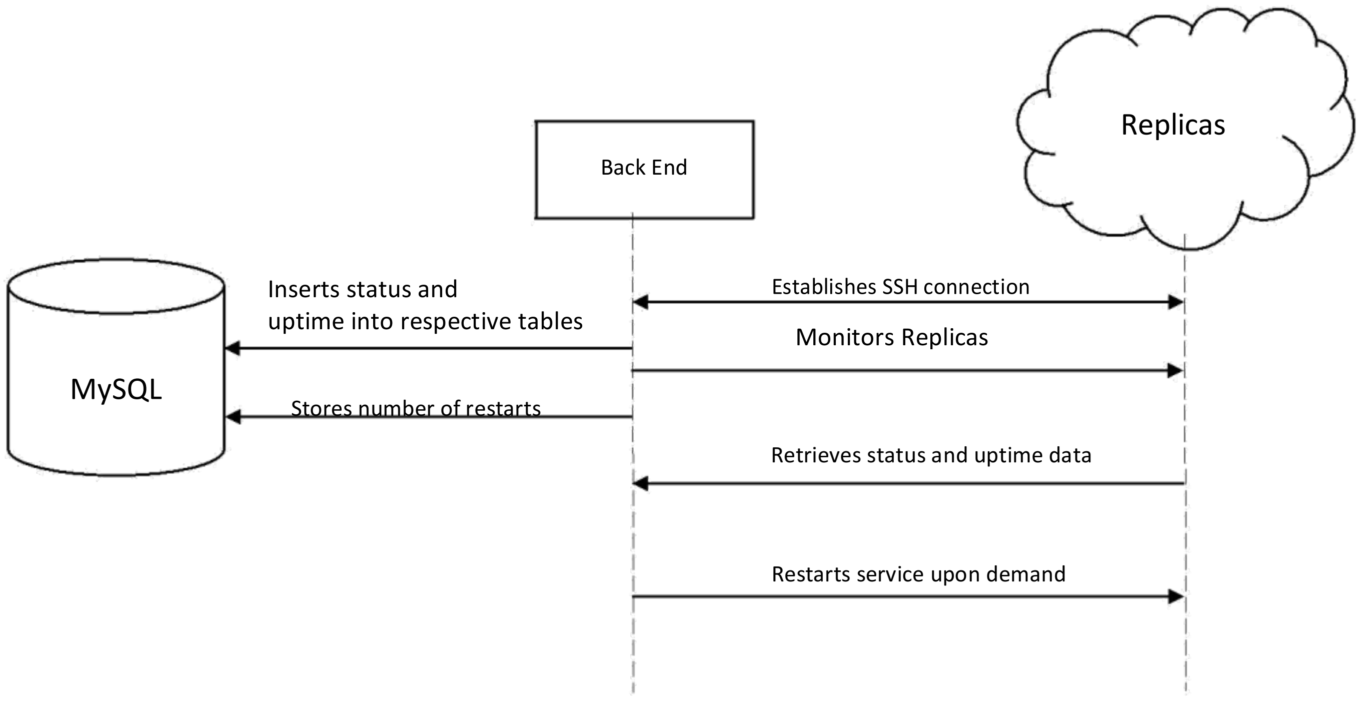

Back End
Replicas
MySQL
Inserts status and
uptime into respective tables
Establishes SSH connection
Monitors Replicas
Stores number of restarts
Retrieves status and uptime data
Restarts service upon demand

Fig 4: Back End

4. Requirements

The project requirements define the basic lists of requirements for Distributed File Storage system. Requirements are classified in to user and system Requirements

4.1 User Requirements:

The User requirements describes what the services provided for the customer, hear we describe the function and non-functional requirements.

Requirement	Identification	Module	Type	Dependencies	Description
Simple GUI	REQ-USR_1	Frontend	Functional	REQ-SYS_1 REQ-NFSYS_1	A web-based GUI shall be provided to enable users to control the database and manage files
Login Authentication	REQ-USR_2	Frontend	Functional	REQ-SYS_1 REQ-SYS_2 REQ-SYS_3	To prevent unauthorised users to access the files and to provide login only to the customers with proper user name and password
File Availability	REQ-USR_3	Database	Functional	REQ-SYS_2 REQ-SYS_3	To provide access to the files who login to the data base
File Sharing	REQ-USR_4	Backend	Functional	REQ-SYS_3	To provide sharing and transfer of files between user to user
Security	REQ-USR_5	Backend	Functional		To provide ssh encryption between node to node and ssl encryption between server and user

Table 1: User Requirements

4.2 System requirements

System requirements describes the technical details of Distributed File Storage system. These are classified into function and non-functional requirements.

Requirement	Identification	Created Date	Module	Type	Description
Web Server	REQ-SYS_1	2016-4-21	Frontend	Functional	It is used to provide web pages requested by client's computers. The web server that is requested is Apache2 Local server
Data Base	REQ-SYS_2	2016-4-21	Database	Functional	We use MySQL to store the user data, status, login details. Through MySQL we create tables where information is stored
Replicas	REQ-SYS_3	2016-4-21	Backend	Functional	duplication of files is performed. these files are stored in rest of the servers
Operating System	REQ-NFSYS_1	2016-4-21		Non Functional	The system requirement is Ubuntu 14.04 LTS
Languages: HTML CSS PYTHON JAVA SCRIPT	REQ-NFSYS_2	2016-4-21		Non Functional	HTML, CSS, Java Script are used to building web pages, Python is used in replication and MySQL is used for storage
Compatibility	REQ-NFSYS_3	2016-4-21		Non Functional	The developed product must be compatible with previous and going versions
Testability	REQ-SYS_4	2016-4-21		Non Functional	The product must support software testing to high extent in order to detect bugs

Table 2: System Requirements

5. References:

- Sommerville, Ian. Software Engineering, 9th ed. Addison-Wesley, 2011