Weighing Motion System

Cloud System Document

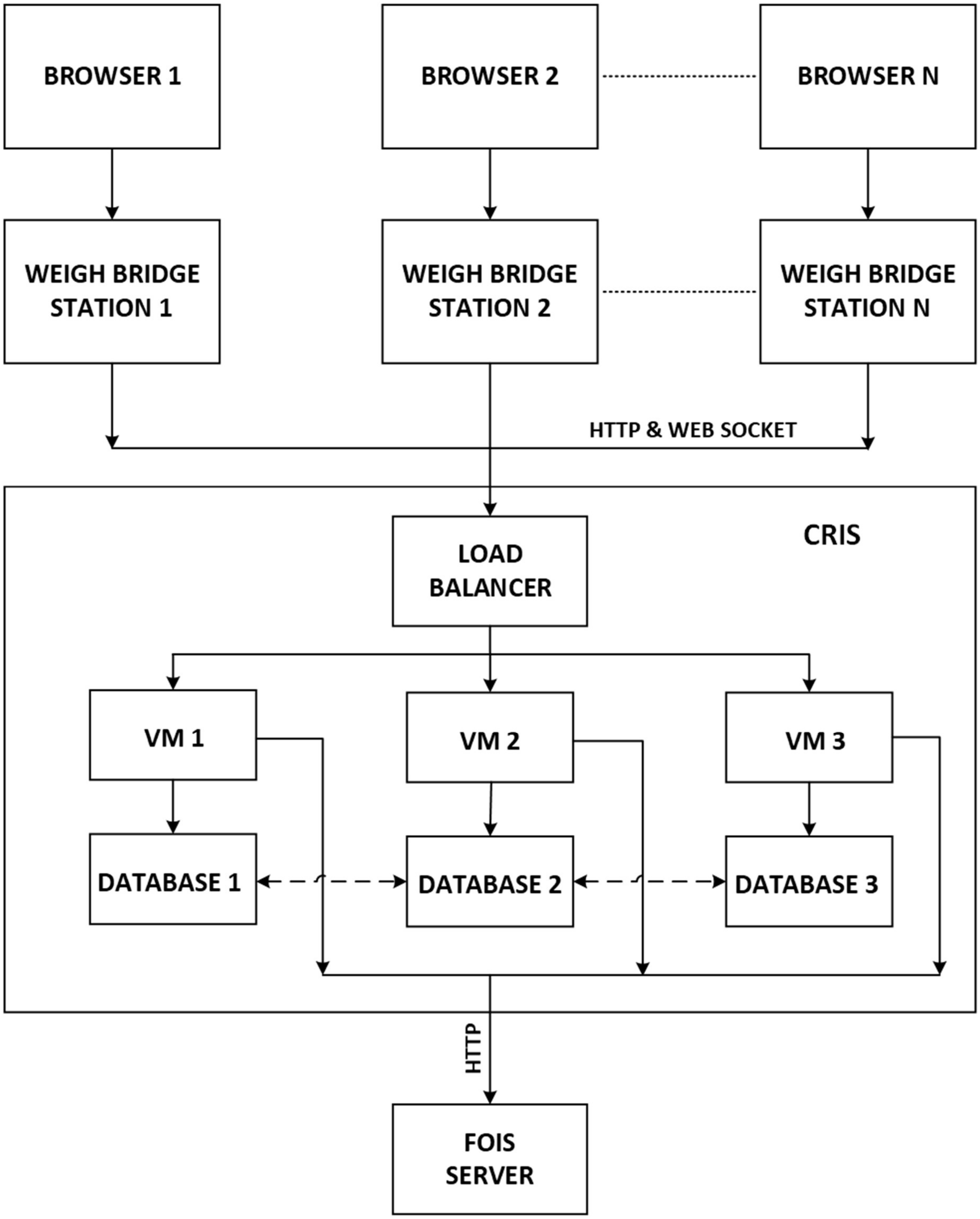
Gautham D

1 Table of Contents

2 Revision History

Version No	Date	Author	Change Log
1	10/02/2024	Gautham D	Initial

3 Cloud Architecture



4 Architecture Description

The cloud architecture consists of the following three features,

1. WeighBridge Station
2. Virtual Machine (VM)
3. FOIS Server

4.1 Details of WeighBridge Station

WeighBridge station is the location where the operator initiates the whole weighing system. WeighBridge also has the Track Logic Controller (TLC) client which communicates to the Track Weighing system and uploads the data to the virtual machine (VM). The operator has to manually download the rakes available for the specific date and choose the rake for weighing. The operator can also edit the selected rake's wagon details and initiate the weighment. After the weighment is finished, the operator has to manually terminate the weighment to upload the weighed data to the cloud. The uploaded data can be viewed on the report page and data must be uploaded to the FOIS server by the operator.

4.2 Details of Virtual Machine (VM)

The webserver is hosted in the virtual machine. The cloud application has been developed and deployed in the Virtual machine. The purpose of the virtual machine is to act as a bridge between the weighbridge station and the FOIS server. All the data has been stored in the virtual machine using the database. All three VM's database is interlinked therefore if any values are added to the one database, then the newly added data will be updated in the other VMs.

4.3 Details of FOIS Server

FOIS server is fully maintained by RAILWAYS. The Rakes assigned for weighing have been downloaded from the FOIS server and after weighing, the weighed data will be uploaded to it. The FOIS server URLs used are as follows,

"http://10.60.200.54:50014/foisCustomer/FoisCustomerWS"

5 Application Layer

- FrontEnd
- BackEnd

5.1 FrontEnd

The frontend has been developed using ReactJs.

5.1.1 ReactJS

- ReactJS is a JavaScript library for building user interfaces.
- ReactJS allows us to create reusable UI components
- ReactJS version 18 is used.
- React Documentation – https://react.dev/

5.2 BackEnd

The backend has been developed with Django and for storing data MySQL database is used.

5.2.1 Django

- Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design.
- Django makes it easier to build better web apps more quickly and with less code.
- Django version 4.2 is used.
- Django Documentation - https://docs.djangoproject.com/en/4.2/

5.2.2 MySQL

- MySQL version 8 is used
- Database name is **Weighing.**
- Tables inside database are as follows
  - weighing_permrakedetails – Stores all selected/weighed rake details.
  - weighing_permwagondetails – Stores all selected/weighed wagon details.
  - weighing_temprakedetails – Stores all the downloaded rake details.
  - weighing_tempwagondetails – Stores all the downloaded wagon details.
  - weighing_weighbridge – Stores all the weighebridge station details.
  - weighing_wgidsettings – Stores all the weighbridge's default settings values.

5.2.3 Nginx

- HTTP protocol has been handled by the nginx webserver.
- Requests reaching the Nginx server have been sent to the Django application and handled.

5.2.4 Gunicorn

- Gunicorn is a WSGI HTTP server.
- Gunicron is used behind the HTTP proxy server(ngnix).

5.2.5 Mosquitto

- Mosquitto is an open-source message broker that implements the MQTT protocol.
- Mosquitto is lightweight and is suitable to use on all devices.
- MQTT protocol has been handled by mosquito.

6 Protocol and Ports

6.1 Protocol Used

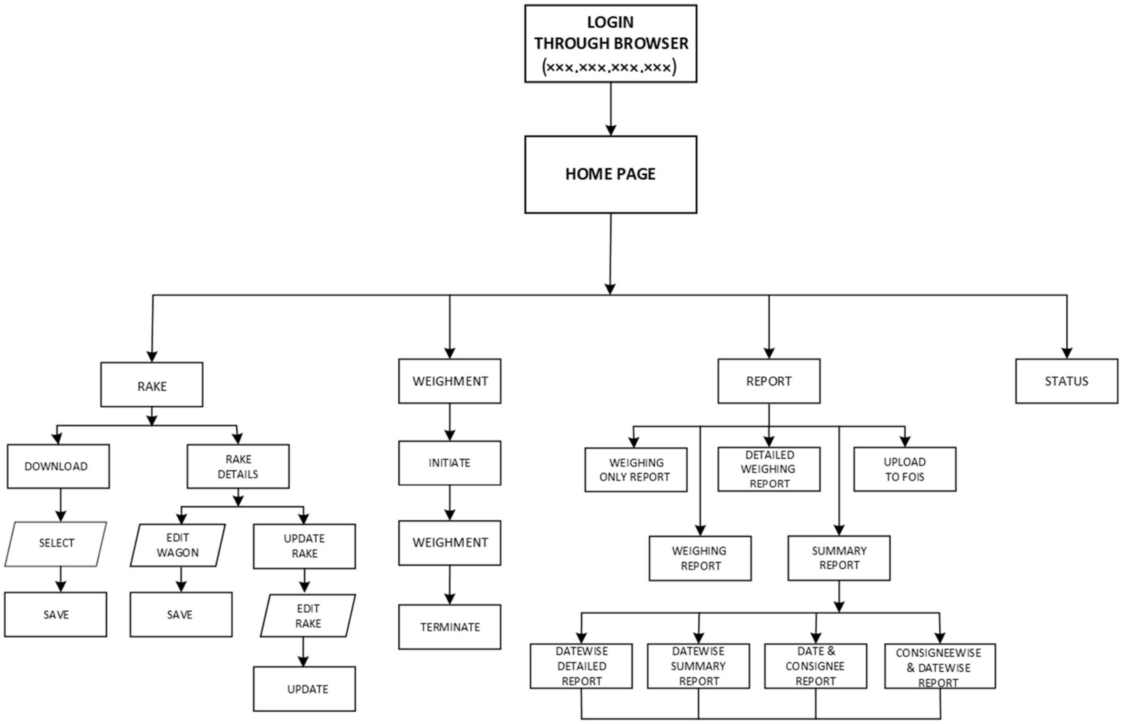
- HTTP
- WebSocket Over MQTT

6.2 Ports Used

- 443 – Used for HTTP requests and responses.
- 80 – Used for MQTT Communications.

Note: - Cris didn't permit to use MySQL port number for database replication. Therefore, the database replication was not implemented on all VMs.

7 Cloud System Flowchart



8 References