cQube

cQube – Technical Specifications Document

April 2020

Version 1.3

Document released by:

Sreenivas Nimmagadda

Document reviewed by:

Arvind Gopalakrishnan

Document acceptance by:

Contents

1. Background and Context

EkStep and Tibil Solutions are embarking on a project to create an analytics product 'cQube' for the education system. This product can be used for monitoring the education system in a state and on a broader scale for monitoring the schools across the various levels of administration.

This document describes the product design and the specification requirements for the cQube product.

1.1 Use of this document

The use of the document is for anyone who has the permission to view documents of the completed project in order to help understand how this solution has been architected and designed.

1.2 State of this document

This document is in a final draft version and is under change control. To request a change to the technical document please contact the system architect or the project manager.

1.3 Acronyms

The following is a list of acronyms which will be used throughout this document:

Acronym	**Description**
S3	Simple Storage Service
NIFI	Apache NIFI
PK	Primary Key

FK	Foreign key
RDAC	Restricted Database Access Control

Table – 1: Acronyms

2 cQube Product setup

This section describes the prerequisites to install and configure the cQube product setup and the cQube product setup process. This section also describes the cQube network setup and the setup process

2.1 cQube product prerequisites

The cQube product installation process has a few system prerequisites that have to be followed. The system software, hardware, and security requirements have been derived and have to be adhered to before installation. The figure below gives an overview of the software requirements, security requirements, hardware requirements and the data storage locations:

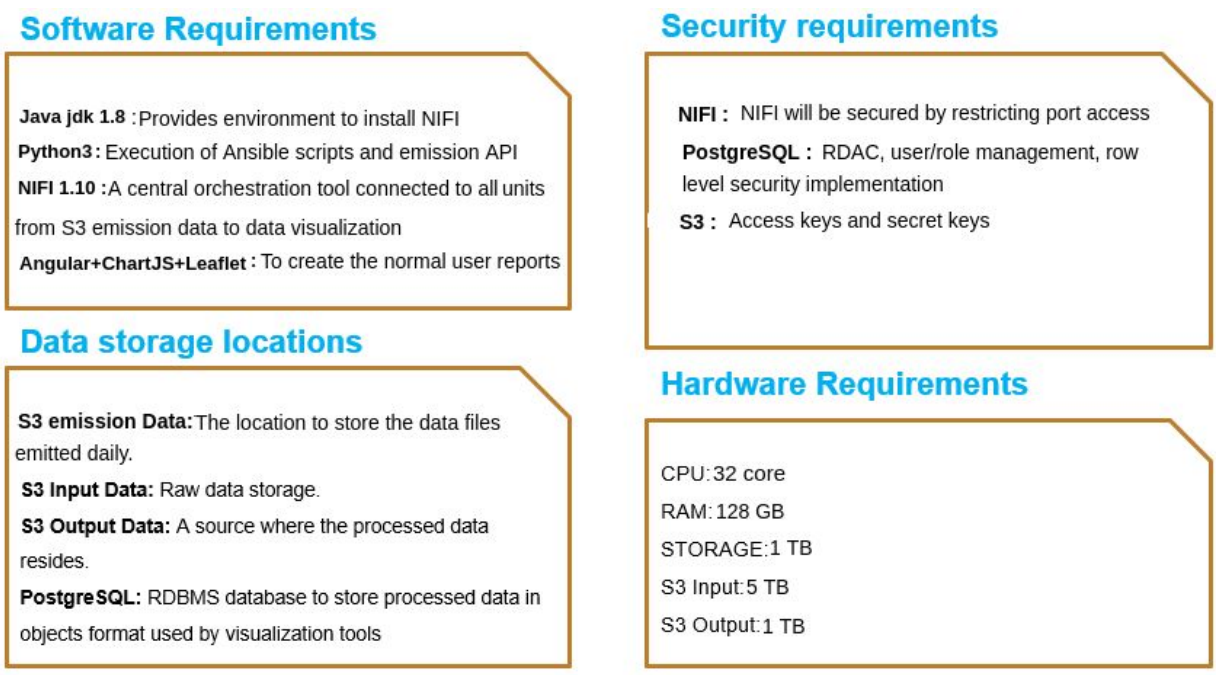

Figure – 1: System hardware, software requirements, security requirements and data storage

2.1.1 Software Requirements

- Java JDK1.8: This provides an environment for NIFI installation
- Python3: Python plays a role in the execution of Ansible scripts and data emission API using a virtual environment.
- NIFI 1.10: A central orchestration tool which is connected to all the units and all the different sections where the data flows, starting from the S3 emission data location to the data visualization stage
- PostgreSQL: An RDBMS database to keep all the processed data in relational data format. The data stored here is used by NIFI to prepare the JSON format files which can be used in the visualization charts.
- Angular + ChartJS + Leaflet: Angular, ChartJS and Leaflet are used to create dashboards/ user reports. The data stored in the S3 output bucket can directly be used to create the reports.

2.1.2 Security requirements

- Nifi - Nifi will be secured by restricting port access
- PostgreSQL - RDAC, user role management, row level security implementation
- S3 - Access keys and secret keys

2.1.3 Data Storage Locations

- S3 emission data Location: The data emitted from the state education system has to be stored until the NIFI reads the data. S3 emission storage buckets are the storage locations where the emitted data files are stored.
- S3 Input Data: This is a location where the raw data resides for all future references.
- PostgreSQL: All the transformed and aggregated data will be stored in PostgreSQL tables.
- S3 Output Data: A location where the processed data resides in JSON format.

2.1.4 Hardware Requirements

Input load	

Table - Input load considerations

2.2 cQube - Product setup block diagram

The diagram below describes the cQube product installation process, the product configuration process and an overview of all the different software that have been used in the cQube tech stack.

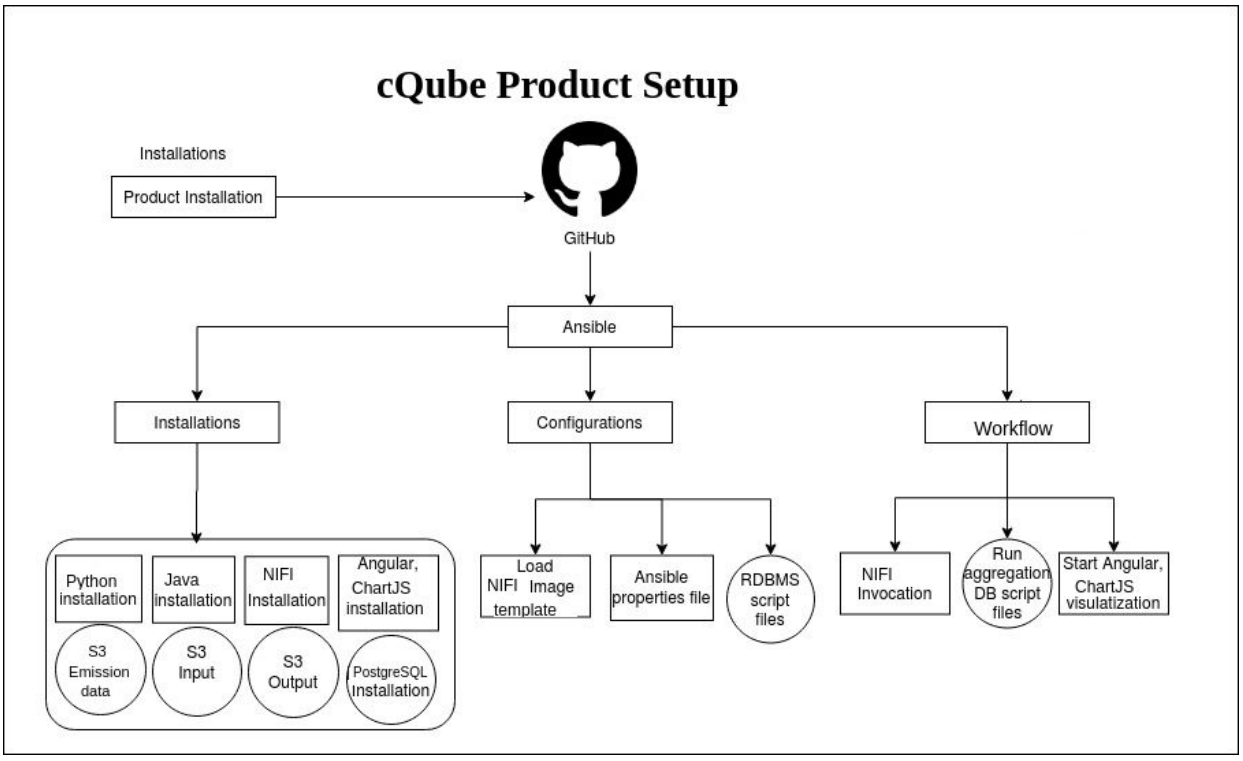

Figure – 2: Flow diagram for the cQube product setup

2.3 cQube - for Installation

The cQube product can be installed with a one-step installation process. The one-step process installs the complete cQube stack, which includes Ansible automation scripts to install Java, Python, NIFI, Angular, ChartJS, Leaflet, S3 emission data, S3 input bucket, S3 output bucket and PostgreSQL installations.

Steps for Installation:

- As a first step, clone the files from GitHub using the following command:
  $ git clone https://github.com/project-sunbird/cQube.git.
- This downloads the cQube installation files. The cQube GitHub has all the installation files required for installing cQube.
- The README.md file has all the instructions that have to be followed for the cQube product installation.
- The install.sh script installs the complete cQube stack.
- The Install.sh file calls the Ansible playbooks in the background which will complete the cQube installation setup.
- The complete installation process takes approximately 30 minutes.
- Once the installation is complete, the message “cQube successfully installed” is displayed.

2.4 cQube - Configuration process

Configurations are done as a part of the installation process by the ansible scripts. All the properties of the sensible information like URLs and passwords are saved in the ansible properties file which is encrypted by default. Configuration also is an automated process and no manual interactions are required.

With the configuration stage cQube having the flexibility to support multiple state data with minimal changes in UI code.

Steps for configuration:

- After cQube installation is completed , cQube needs to be configured depending on the emission data fields and configuration file .

- During configuration users can activate/deactivate the attributes, but not the key columns (ex : student_id,school_id cannot be deactivated). fields like subject names, infrastructure attributes for particular states can be activated/deactivated.
- Once the configuration file is emitted configuration process will be invoked
- In configuration stage fields will be activated/deactivated for that state, based on requirement
- Based on the configuration, the changes are made to handle state specific data fields
- After successful validations, data tables are created with only active and required data fields The active data fields are processed and metrics are generated
- The structure will be metrics generated will be stored in JSON files, which is used for visualization

Example : For Infrastructure table

id	cQube column	State column	Status
1	toilet	toilet	1
2	handwash	handwash	1
3	solar_panel	-	0

2.5 Workflow

NIFI is the central orchestration system which handles the complete workflow process and generates JSON files. These JSON files are used for creation/ generation of the visualization reports.

Steps for the workflow:

- Data from the state education system will be emitted into the S3 emission data folder using the cQube data emission API (emission API is described in the section 4 of this document)
- Once the data emission is complete, all the emitted data resides in delimited files in the S3 emission data bucket.

- NIFI fetches the stored data from S3 emission bucket and retains one copy of the data for future reference in S3 input bucket.

2.6. cQube network setup

The cQube network setup consists of the AWS, which encompasses the private subnet section which contain the EC2 instances Postgres, NIFI, node, angular and python, the public subnet section which comprises of the Nginx and VPN, S3 and the Load balancer. The cQube network setup process is described in the block diagram below:

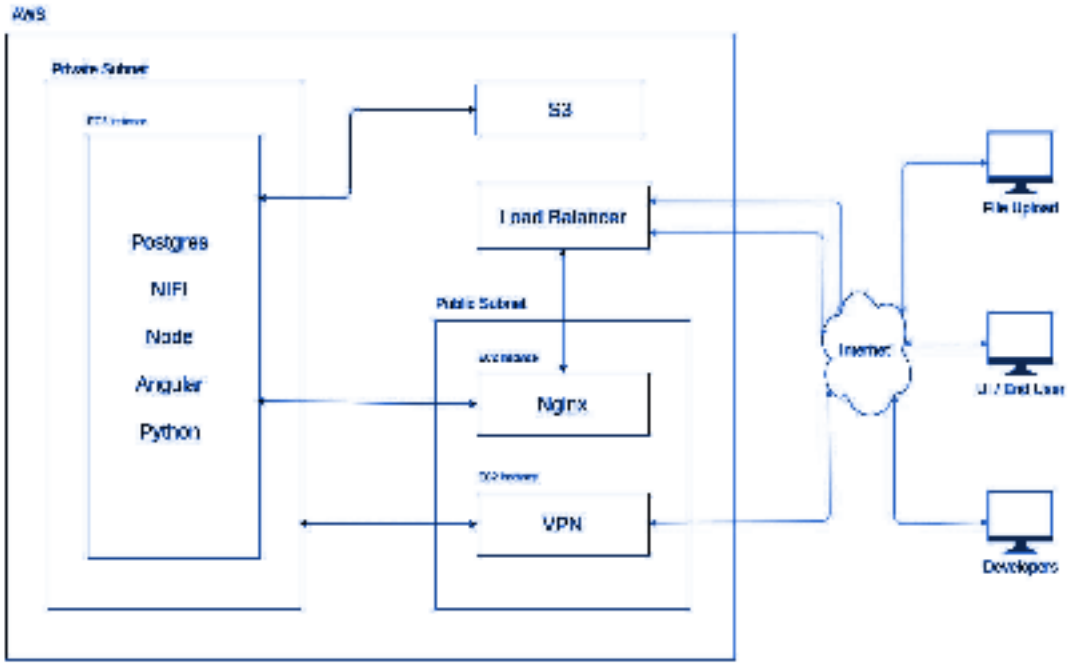

Figure – 3: cQube network setup diagram

2.6.1 End User:

- When a user accesses the cQube application through the browser, the request hits the load balancer of the AWS.
- Load balancer will forward the request to Nginx proxy server.
- Nginx will forward the request to angular application using private IP
- Angular sends the request to NodeJS server
- NodeJS will get the data from S3 bucket and respond to Angular
- Angular will process the data and display the results.

2.6.2 Emission User:

- These users call the Rest API through python client-side script to upload the files
- After user authentication, Rest API (Written in python) provides the S3 one-time URL
- Client-side python upload the files using the S3 one-time URL

2.6.3 Developers:

- Normally developers can deploy the changes through the Jenkins CI/CD pipeline.
- Developers use VPN to connect to the cQube production application if there are any direct changes, like technical changes, configuration or customizations are required.

3. Software Architecture

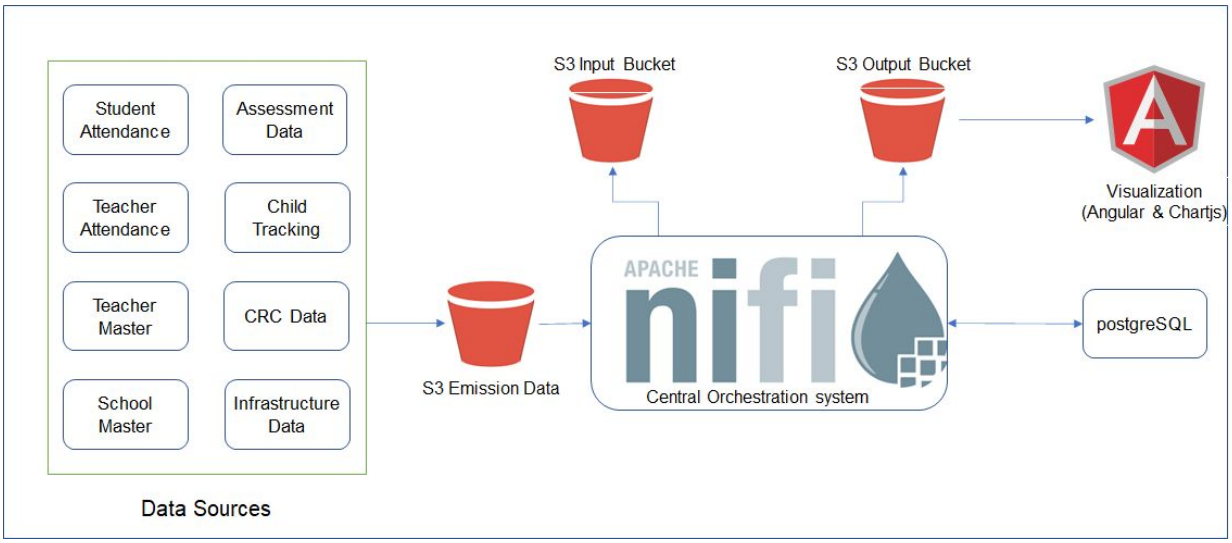

Figure – 3: cQube software architecture

3.1 Input data sources:

cQube can process the data sources like

1) Student attendance
2) Teacher Attendance
3) CRC app data

4) Semester data
5) Infrastructure data
6) Periodic assessment test data
7) Child tracking
8) Teacher master
9) School master

3.2 Emission data Storage:

Data fetched from the above mentioned data sources will be emitted to the S3 emission data bucket.

3.3 Data Validations:

- Data validations will take place at the following different levels:
  - The emitted data undergoes their first set of data validations before being copied into the S3 Input bucket
  - Second set of data validation takes place after the files are copied into the S3 Input bucket and before the Nifi process starts processing the data for cQube report creation.

- NIFI fetches the data from the S3 emission data bucket and sends it to the S3 input bucket, after performing the following validations on the data:

  - **Emitted data file size check:** NIFI gets the file size of the emitted data from the Manifest file and performs a check to see if the emitted file size matches with the original file size.
    NIFI does not allow the file into the S3 Input bucket if the file size does not match. A notification is sent through an email to the data emitter to check the file size and re-emit the data file.

- **Record count at the emitted file check:** NIFI gets the count of the number of records in the emitted data from the Manifest file. Nifi checks if the emitted file records count matches with the original file records count.
  If the file records do not match, the NIFI will not process the file into the S3 Input bucket. A notification email is sent to the data emitter to re-emit the data file.

- NIFI fetches the data from the S3 emission data bucket and sends it to the S3 input bucket, after performing the following validations:

  - **Column level validations**: Column datatype mismatch, Number of columns, Data exceeding the column size.

  - **Improper Data handling**: Missing/ null data values for mandatory fields, Empty data files, Special characters, Blank lines in data files.

  - **Duplicate records validation:** NIFI validates the duplicate records by grouping the same kind of records together.

    For the record which is having duplicate values for all fields (mirror image record) then NIFI will consider the first record and the rest of the records will be eliminated.

    For the rest of the duplicate records where the records are having the same ID (student ID/ assessment ID/ infra ID/ CRC visit ID) and different values will not be inserted into the database tables as ID is the primary key.

    For the Duplicate records with different lat long details, NIFI eliminates the records which have the same id and different lat long details and the records which have different ids and the same lat long details.

    For semester report, The records which are having the same values for fields Student ID, School ID, semester, studying class and different values for the subjects then NIFI will eliminate those records.

- **Overlapping data validation**: Overlapping data validation takes place based on the data source.

  The NIFI process for student attendance reports will check the last updated day's record from the transactional table and will process the records from the day after the last updated date. The records from all of the previous days will not be considered for NIFI processing.

  For the other data sources, duplicate records where the records are having the same ID (student ID/ assessment ID/ infra ID/ CRC visit ID) and different values will not be inserted into the database tables as ID is the primary key.

- **Other data issues**: Data handling in cases like job failures, missing data for certain days and late receipt of the data (receiving data after a few days), updating the wrong data, upon request (when issue identified at the report)

3.4 NIFI Processor Groups:

- NIFI will have processor groups to combine similar NIFI processors, i.e the processes that are related to the same functionality into batches.
- Separate processor groups enable only the required data source transformation and this supports further scaling.
- The processor groups are loosely coupled , which enables them to make specific changes required to any particular processor group without affecting all the other processor groups.

3.5 Data Process:

- NIFI acts as the central orchestration system and all the aggregations are performed in the NIFI using PostgreSQL databases.
- NIFI fetches the aggregated data from the PostgreSQL database and sends them to the S3 output bucket.

3.6 Dashboard generation:

- Angular fetches the processed data from the S3 output bucket to generate all the reports using ChartJS/ Leaflet.

4. cQube data flow

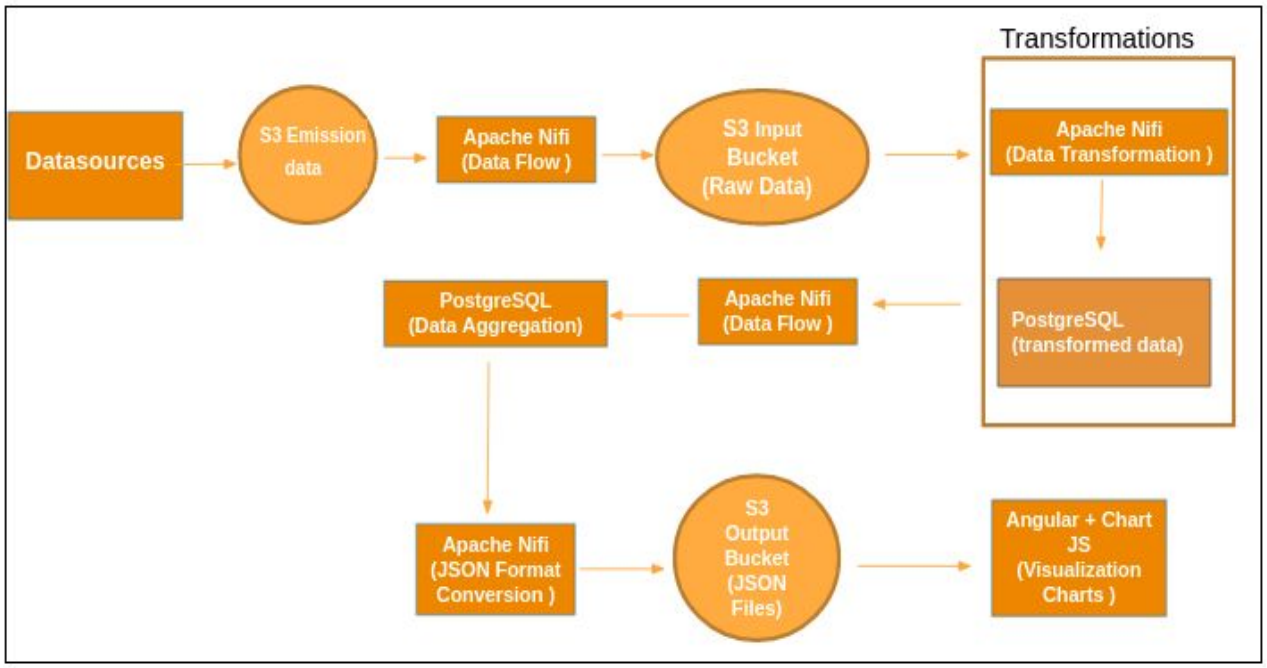

Figure – 4: cQube data process flow

Description of the data flow:

S3 emission data	Is the location where the emitted data resides. This folder is created in AWS S3 Raw data files are emitted to S3 emission data buckets periodically.
S3 emission data - NIFI	NIFI collects the raw data files from the S3 emission data location and after that removes them from the S3 emission data folders.

NIFI	NIFI sends the raw data to the S3 (input) bucket. NIFI does the validation of data fields and values. NIFI inserts the transformed data to PostgreSQL. NIFI specifies the scripts to PostgreSQL to perform the necessary aggregations for visualization reports. NIFI picks the aggregated data from PostgreSQL and places them into the S3 bucket (output) to create JSON files which can be used for the visualization reports
PostgreSQL	PostgreSQL base tables are populated by NIFI PostgreSQL transformed tables are populated by NIFI PostgreSQL aggregate tables are populated by NIFI
S3 Output Bucket	S3 Output bucket contains the files in JSON format Angular reads the data from S3 output bucket by using Node API
Angular, ChartJS/ Leaflet	Angular fetches the data using Node JS from S3 to create reports using ChartJS/ Leaflet

Table – 2: cQube data flow tables

5. Security Implementations

- **EC2**: A pair of public and private keys are generated, and the public key is stored in the EC2 server. The client with the private key gets authenticated with the server during login only if the keys match.
- **S3 emission data bucket**: cQube provides a data-ingestion API to emit the data. The API will have different secured endpoints like:
    1. User authentication
    2. API call to emit the data files using the https protocol into cQube.
    3. API takes the data file as a parameter.

4. The API will provide acknowledgement on successful emission.

- **S3 Input & Output buckets**: A pair of access keys and secret keys are generated to secure the S3 location.
- **NIFI**: Only authorized users can gain access to the NIFI Dashboard. The confidential keys such as username and password will be encrypted.

- **PostgreSQL**: PostgreSQL will be secured as follows:
    1. User Access Control (RDAC)
    2. Server configuration
    3. User and role management
    4. Logging
    5. PostgreSQL audit extension (pgAudit)
    6. Security patches
- **Angular**: Role based authentication will be provided to prevent access to unauthorized users. A reverse proxy server has been used that usually stays behind the firewall of a private network. Reverse proxies are also used as a means of caching common content and compressing inbound and outbound data, resulting in a faster and smoother flow of traffic between the clients and servers.
- Role based authentication: Will be provided to admin users, and based on the user roles, relevant access will be provided to users at district levels, block levels, cluster levels and school levels.
- Normal/ regular users can access public reports without any authentication.

6. cQube data processing

6.1 Data emission process

cQube provides an API for the data ingestion. Authenticated API call provides onetime S3 URL and emits the data into the S3 emission bucket.

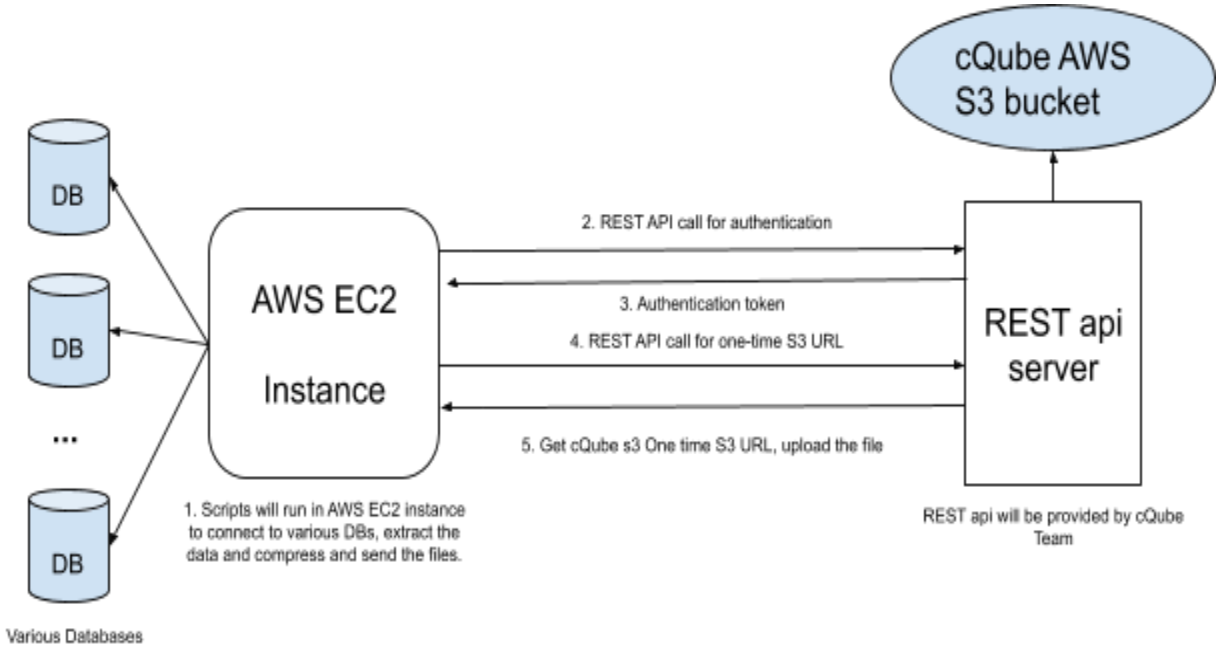

Figure – 5: Data emission process

The steps involved in the data Emission Process

- The data emitters will work from the data source location.
- Data emission will be performed periodically as per the specified time interval from different data sources with the help of an automated extraction process.
- Once the emission process extracts the data fields which are used by cQube, it is converted into csv formatted, pipe delimited files.
- The CSV data files will then be placed into the state data center by the automated process.
- Emission automated processes will invoke cQube data-ingestion APIs to emit the data.
- The API will have different end points as mentioned below:
    - Data emission users will request the cQube admin for the emission API token.
    - Data emission users incorporate the API token into the emission process code.
    - The Emission process makes an API call to generate AWS S3, one time presigned URL.
    - API calls to emit the data files using the https protocol into cQube.
    - API takes the data file as a parameter.
    - The API sends an acknowledgement on successful emission.

6.2 S3 bucket partitioning

S3 buckets will contain partitions for the data files to store. The partitions are created at the S3 input bucket & the S3 Output bucket.

6.2.1 S3 emission bucket partitions:

- All the files in the S3 emission bucket will be in the CSV format.
- S3 emission bucket follows the folder hierarchy based on the data sources.
  S3 -> Bucket name -> Data source -> emitted zip files with timestamp
- The emitted zip file contains the CSV data files with timestamp and a manifest file with a timestamp.
- The folders and the files will be removed from the S3 emission bucket once NIFI copies the data.
- Unprocessed data files will remain in the S3 emission bucket for one week and then they will be deleted automatically at the end of the week.

6.2.2 S3 input bucket partitions:

- All the files in the S3 input bucket will be in the CSV format.
- S3 input bucket follows a hierarchical partitioning based on the Data source, Year, Month, date and timestamp
  S3 -> Bucket name -> Data source -> Year -> Year - Month -> date_Source name

  Example for the S3 input bucket:

  S3/cqube-gj-input/student_attendance/2020/2020-05/2020-05-29_student_attendance

6.2.3 S3 output bucket partitions:

- All the files in the S3 Output bucket will be in the JSON format.
- S3 output bucket follows the hierarchical partitioning based on the data source, Year, Month, date and timestamp, similar to the partitioning that the S3 input bucket follows.
- Metadata files will have information of the latest updated output files which helps cQube to consider the latest output file during the visualization stage.

6.3 NIFI data pipeline

cQube uses Apache NIFI to automate the data flow

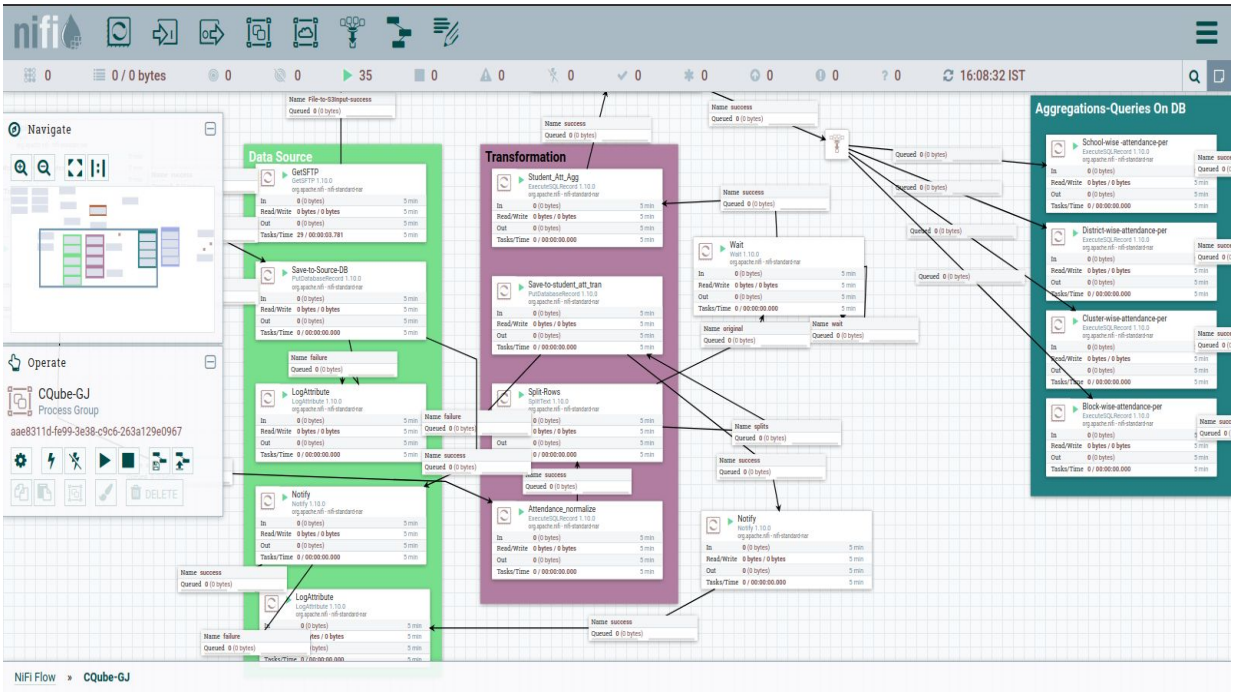

Figure – 6: NIFI flow screenshot

- Apache NIFI Data Source Processor fetches the data from S3 emission data bucket.
- Stores raw data into S3 Input bucket.
- NIFI processes perform the data validations.
- Transformation Processor: Data is transformed using the queries in PostgreSQL and the transformed data is sent to the PostgreSQL.
- Aggregation Processors: Data aggregations are performed on the transformed data.
- NIFI stores the aggregated data and static data into S3 output buckets in JSON format.

Aggregation files

7. cQube Data Model

This section describes the student attendance, student assessment, infrastructure data and the teacher attendance data structure.

7.1 Tables classification

All the entities are classified into four types

- Static tables
- Metadata tables
- Hierarchical tables
- Dynamic tables
- Aggregated tables

Static tables: The tables which contain the static information of the entities like geo master, school master, student master, subject master

Metadata tables: The tables which are used to store the processing information of the emission process like incoming files and process status. To store the information of the days when the student attendance data is processed.

Hierarchical tables: These tables contain the rarely updated values or dimensions that change rarely such as student class, academic year.

Dynamic tables: These tables contain the daily or frequently changing values such as the student attendance, Teacher attendance, CRC data, Semester assessment.. These tables are updated more frequently.

Aggregated tables: These tables will hold the school-wise aggregated values which will be ready to be converted into JSON files. Multiple aggregation tables will be created based on the visualization report requirements.

Initialization stage for Infrastructure dataset: After the configuration stage is completed, the infrastructure master tables are initialized with the infrastructure columns of that state, based

on the columns present in the infrastructure transaction table columns. Once initialization is completed the infrastructure weights can be configured through the emission API.

Click on the link below for the data dictionary:

https://docs.google.com/spreadsheets/d/1OM5jClFb3shyk0KKqXk0jtiThcwHntUXdzTyam_IwDo/edit?usp=sharing

Click on the link below for the latest ERD:

https://drive.google.com/file/d/1s1TJUs5c_IZ5zlQvpRWEnb4jWDVMgkI8/view?usp=sharing

8. cQube users and user roles

The cQube product can be used by a variety of users and each user will have different functions that he/she can perform and different user privileges The cQUbe users can be divided into the following categories:

1. Admin
2. Report Viewer
3. Report Creator
4. Ad-hoc Analyst

The various roles and the functions that each role can perform have been described in the table below.

8.1 Admin

cQube has two different interfaces, an interface for the cQube administrator and another interface/ dashboard for the cQube reports:

- Administrator pages: Administrator activities will run separately in the VPN and admin must login through the 2 factor authentication process to perform the admin tasks.Change password functionality will be included in this login.
- cQube dashboard for report: This is a normal login which will have the cQube insights of the metrics.

8.1.1. Admin login process:

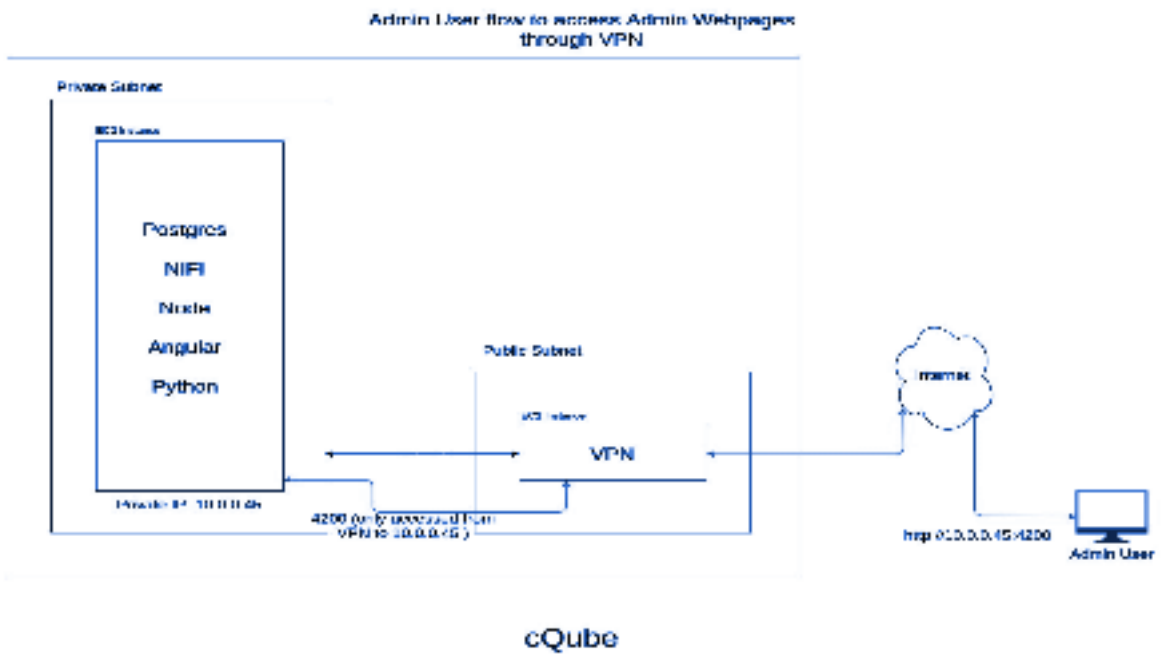

Figure - 5: Admin user flow through VPN

Admin must follow the 2 factor authentication process to perform the admin tasks. The Admin user should connect to the VPN to avail the 2 factor authentication by following the steps mentioned below:

- Admin has to request the devops team for OpenVPN access details.
- Admin has to download the google authenticator app to his phone and register the app with the QR code showing in the OPENVPN Access server page.
- By providing the credentials & Google authenticator code to download the user-locked profile(client.ovpn) file.
- Admin has to install the client openvpn-connect application
- Admin has to create the OpenVPN profile from the client.ovpn file
- Admin has to validate the user authentication and Google authenticator code to login to the cQube VPN.
- With the successful authentication admin can access the cQube admin features by opening the local ip in the browser.

8.1.2 Admin - Features

1. **Activate, Deactivate and Create users:**
   - Admin can create new cQube users from the user interface page.
   - Admin can search for the user
   - Admin is able to activate and deactivate the users by updating the start date / end date in the calendar field from the user interface page.

2. **S3 buckets** - Admin can perform the files cleanup, files Download actions in the S3 buckets by the following steps:
   - Admin will request the Devops team to create the AWS IAM(Identity and Access Management) user permissions to access the S3 buckets.
   - Admin can enter the AWS console and perform the files cleanup, files Download activities
   - Raw data will be downloaded in CSV format and the metrics data will be downloaded in JSON format.

3. **Database Controls** - Admin can perform the database actions mentioned below:
   - Delete data
   - Truncate tables
   - Partitions handling operations
   - SQL dump
   - Restoring the Dump
   - Data Archive

The steps mentioned below describe how the admin user can perform the above mentioned operations:

- After logging in, the admin will be provided with a user interface page to perform these operations.
- Admin can select the desired operation from the options provided.
- Admin has to select the database schema and choose the tables to perform the operations.

- Admin should get the options to kill the active database sessions when required.
- All operations can be executed as mentioned below,
    - Admin will select the table which he wants to perform operations like Archiving data, Downloading the data and Dumping the data.
    - Admin will click on the required action button
    - An API will be executed which is called a shell script to execute the desired functionality.

4. **Application Controls:** Admin can perform the operations mentioned below using the buttons provided:

- Emission API start, stop and restart
- AngularJS & NodeJS start, stop and restart
- PostgreSQL start, stop and restart
- Nifi start, stop and restart

5. **View Logs:**

- All the log files will be consolidated into a common folder using soft links.
- New log files will be created every day and the old logs are retained for 7 days. At any point of time, there are 7 log files in the folder, old files are deleted when new files are created.
- Admin will have an option to view the last 500 lines on the browser with the help of the user interface provided. There will be a download option provided to download the entire log file.
- Log files will be downloaded in .log format.
- The list of log files provided to the admin are as follows:
    - System logs
    - NIFI logs
    - Data emission process logs
    - Database logs
    - User Interface logs (Angular and Node logs)
- S3 logs can be accessed using AWS cloud watch by having the AWS IAM (Identity and Access Management) user permissions.

Grafana Dashboard Connectivity: Admin can use the Prometheus APIs to integrate with Grafana for viewing the cQube application health. The following details are displayed in the grafana dashboard:

- RAM usage
- CPU Usage
- Data storage disk usage
- Nifi heap memory usage
- Data source file metrics
- JVM usage

8.2 Report Viewer

- They are defined in a hierarchical manner,
  For example: If the users belong to the State administrative office, they can view the insight statistics of the districts that belong only to that state.
  If the users belong to the district administrative office, they can view the statistics of the blocks which belong to the district.

  The hierarchy is as shown below:

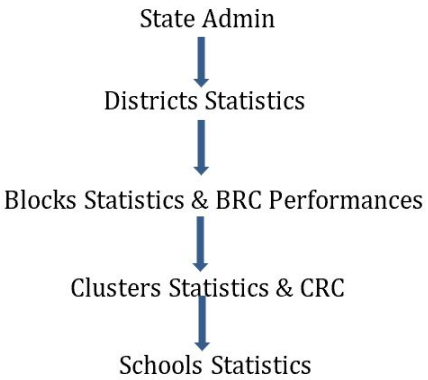

Figure – 6: Report viewer hierarchy

- They can view static dashboards according to their access scope. The selection option dropdown or drilldown options will be provided according to the access scope.
- They will not be able to access any other dashboards which are outside their scope.

- They can have insights into the performance and identify the low and high performers based on the benchmark in the dashboard.
- They can communicate with the users at lower hierarchical levels and provide them with the actionable items or exceptional performance via email (functionality provided in the dashboard)
    - They can alert the low performing district/ block/ cluster/ school based on their performance
    - They can notify the average performing district/ block/ cluster/ school based on their designation, to improve the performance
    - They can recognize the high performing district/ block/ cluster/ school based on their designation, to maintain the performance

8.3 Report Creator

- Report creators can create new dashboards / reports by using the existing reports functionality and metrics.
- These users will be able to call the cQube API to download the metrics files in JSON format.
- They can use the same download API to download the transactional data files and aggregated data files in CSV format from the S3 output bucket.
- Report creators can login into the cQube application to view the existing reports for their understanding of the existing functionalities.
- They can create new reports by using the AngularJS, Node JS, Chart JS, and leaflet maps using the existing code from GitHub repository.

  Note: These users should have the knowledge of the above-mentioned technologies and tools to alter or create the reports.

8.4 Ad-hoc analyst

- They are the dynamic report creators of cQube.

- The ad hoc analyst can make use of third-party visualization tools like Tableau or any other visualization tool to create dynamic dashboards and develop dynamic reports by directly accessing the database.