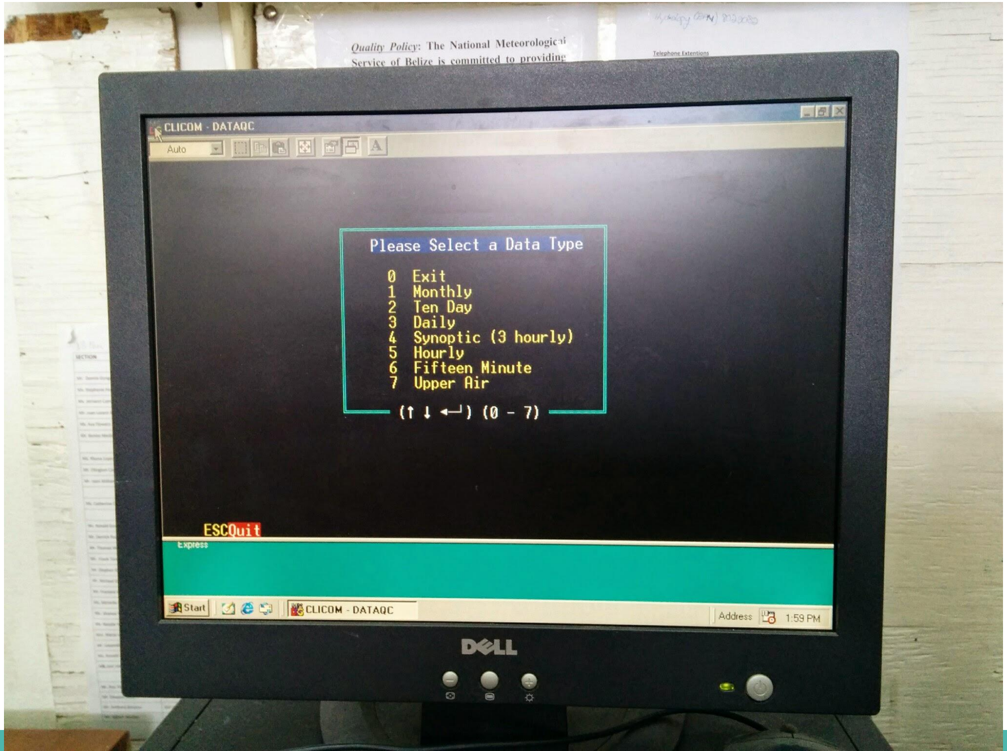
Overview of CDMS

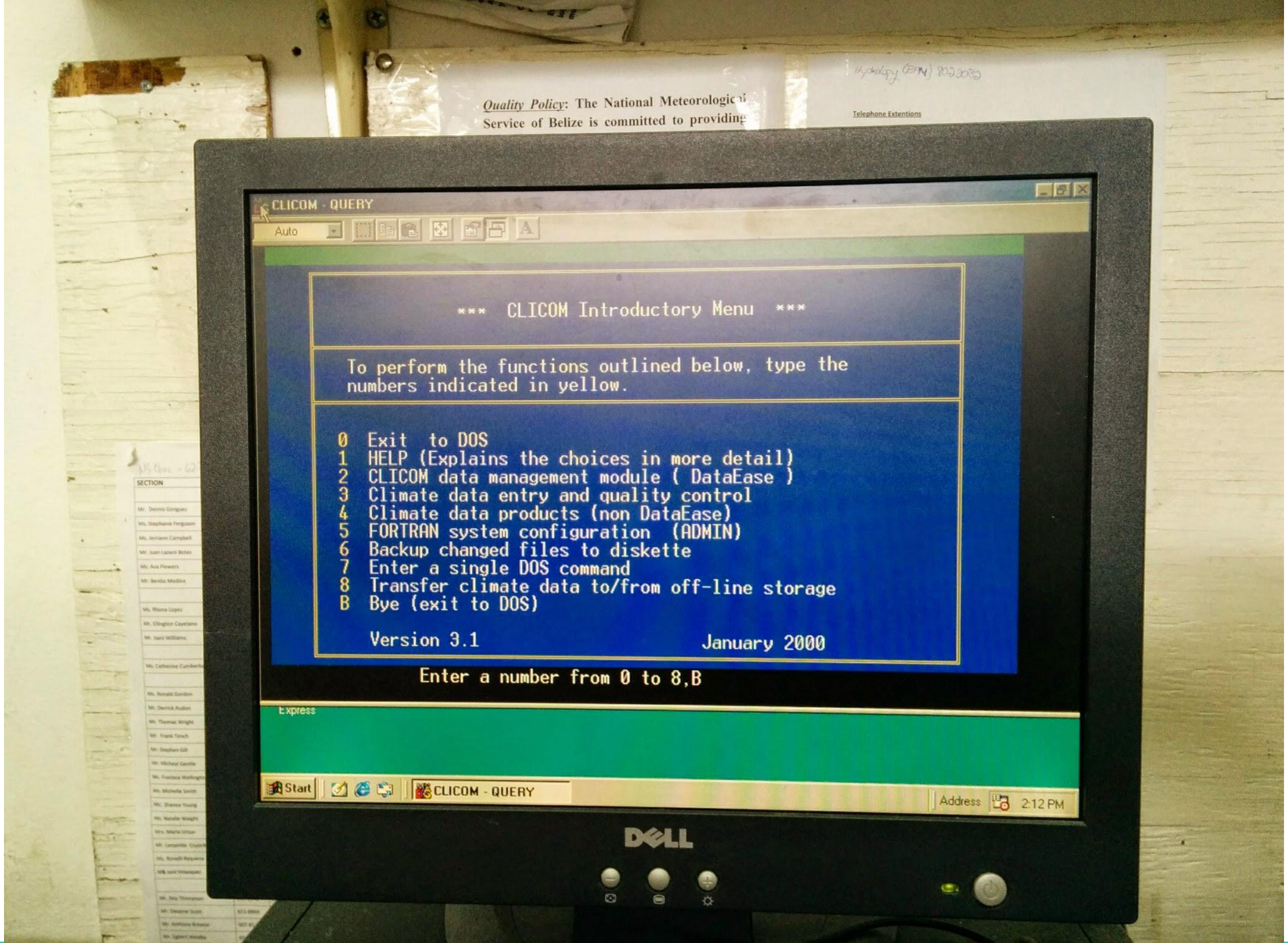
National Meteorological Service of Belize

Dwayne Scott - Technician

Climate database background

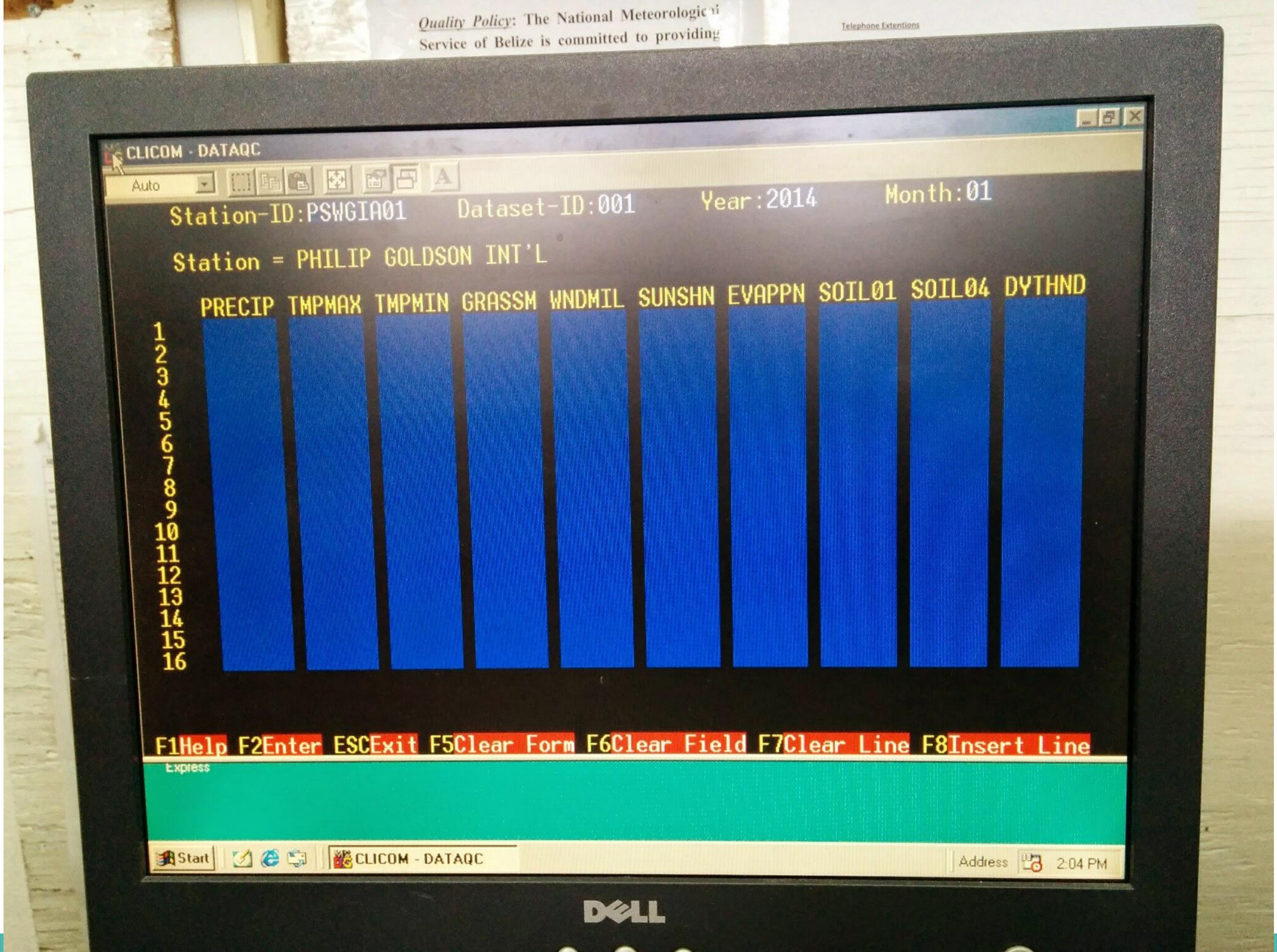
- Prior to 2010 NMS used MS-DOS based Climate database system called CLICOM





Quality Policy: The National Meteorological
Service of Belize is committed to providing
Telephone Extentions
CLICOM - QUERY
Auto
Express
Start
Address
2:12 PM
DELL

```
          ***  CLICOM Introductory Menu  ***

 To perform the functions outlined below, type the
 numbers indicated in yellow.

 0  Exit  to DOS
 1  HELP (Explains the choices in more detail)
 2  CLICOM data management module ( DataEase )
 3  Climate data entry and quality control
 4  Climate data products (non DataEase)
 5  FORTRAN system configuration  (ADMIN)
 6  Backup changed files to diskette
 7  Enter a single DOS command
 8  Transfer climate data to/from off-line storage
 B  Bye (exit to DOS)

    Version 3.1                      January 2000

          Enter a number from 0 to 8,B
```



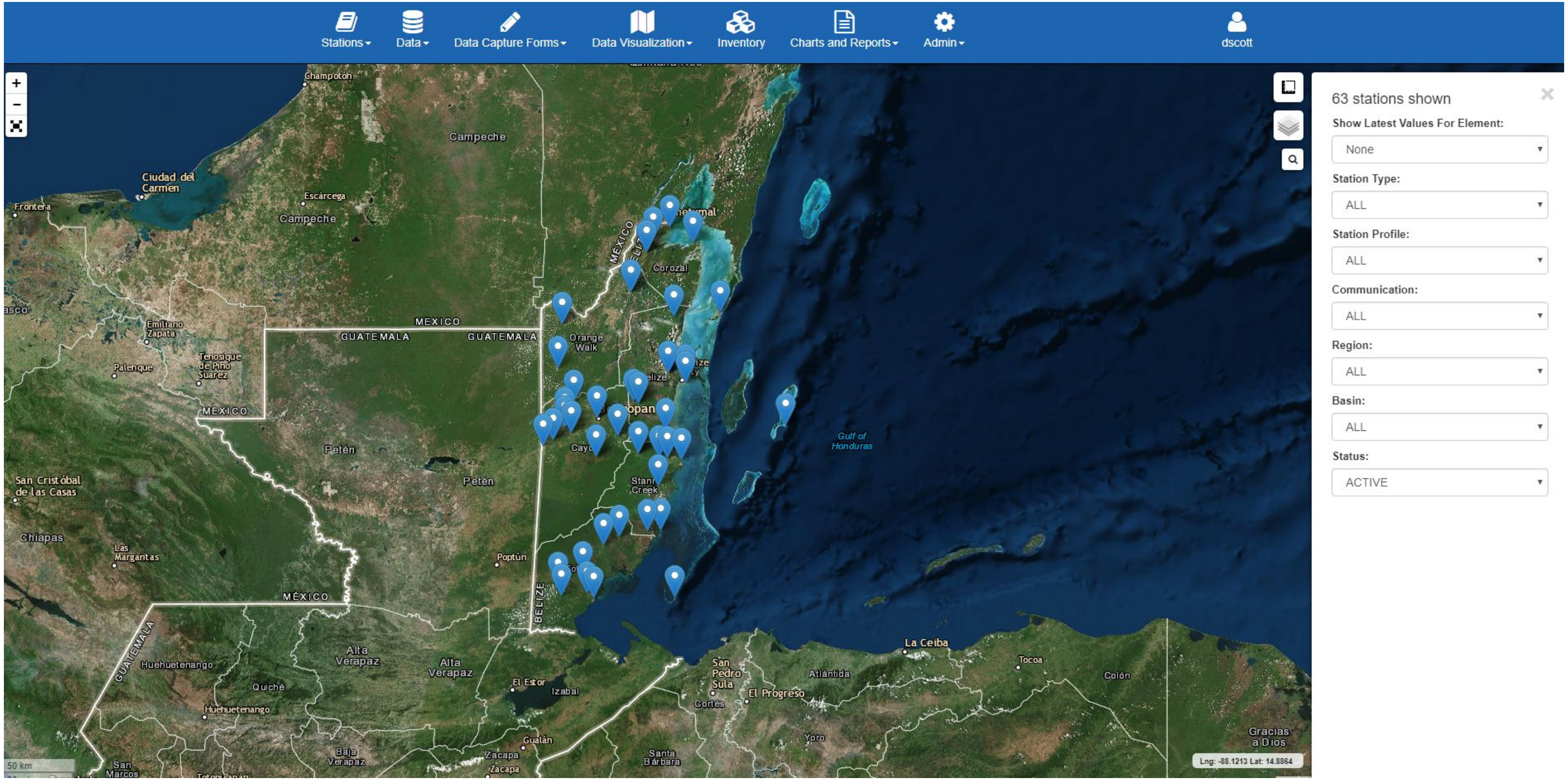
Quality Policy: The National Meteorological
Service of Belize is committed to providing
Telephone Extentions
CLICOM - DATAQC
Auto
Express
Start
CLICOM - DATAQC
Address
2:04 PM
DELL

```
  Station-ID:PSWGIA01    Dataset-ID:001     Year:2014     Month:01

  Station = PHILIP GOLDSON INT'L

       PRECIP TMPMAX TMPMIN GRASSM WNDMIL SUNSHN EVAPPN SOIL01 SOIL04 DYTHND
   1
   2
   3
   4
   5
   6
   7
   8
   9
  10
  11
  12
  13
  14
  15
  16

F1Help F2Enter ESCExit F5Clear Form F6Clear Field F7Clear Line F8Insert Line
```

Climate database background...

- 2010 CLICOM crashed and NMS started to think through development of a web-based climate data management system(CDMS)
- Between 2010 and 2015 the NMS used Central American database system called “BDCAC”
- In 2016 NMS launched its own web based system called “HydrometDB”

HydrometDB - web based user interface



63 stations shown

Show Latest Values For Element: None

Station Type: ALL

Station Profile: ALL

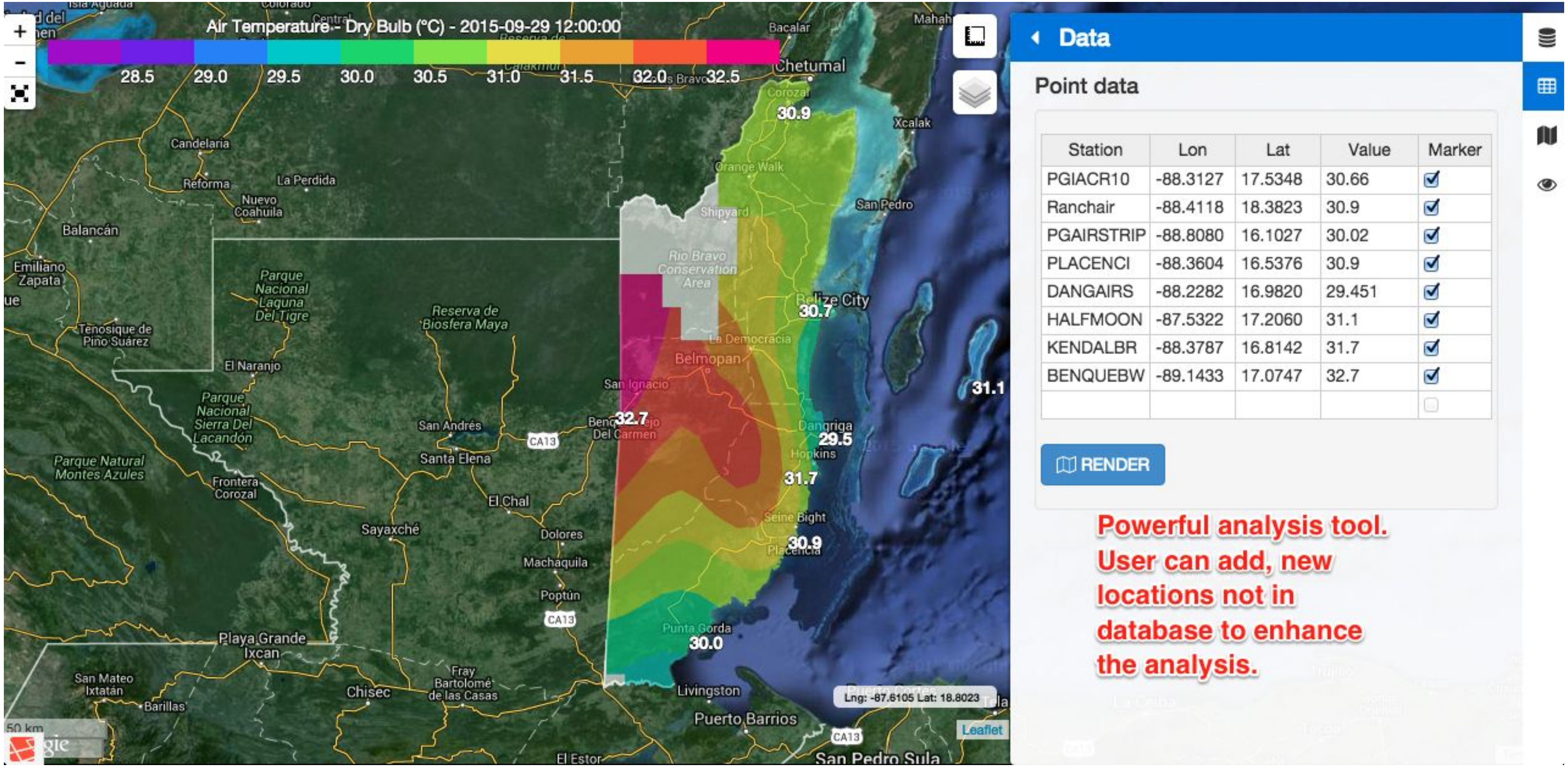
Communication: ALL

Region: ALL

Basin: ALL

Status: ACTIVE

Spatial Analysis - Custom Analysis Tool

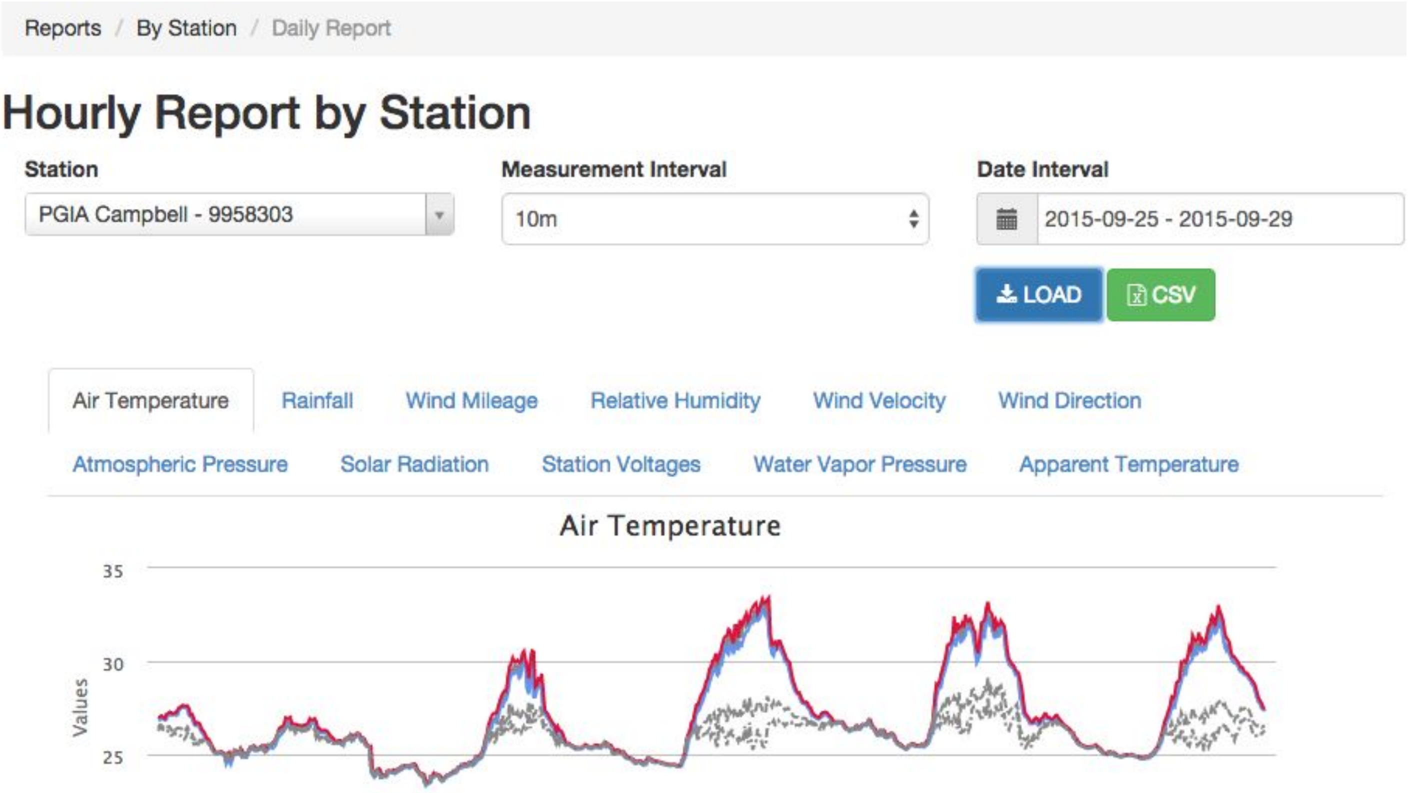


Point data

Station	Lon	Lat	Value	Marker
PGIACR10	-88.3127	17.5348	30.66	☑
Ranchair	-88.4118	18.3823	30.9	☑
PGAIRSTRIP	-88.8080	16.1027	30.02	☑
PLACENCI	-88.3604	16.5376	30.9	☑
DANGAIRS	-88.2282	16.9820	29.451	☑
HALFMOON	-87.5322	17.2060	31.1	☑
KENDALBR	-88.3787	16.8142	31.7	☑
BENQUEBW	-89.1433	17.0747	32.7	☑
				☐

Powerful analysis tool. User can add, new locations not in database to enhance the analysis.

Charts and Reports Samples



Reports / By Station / Daily Report

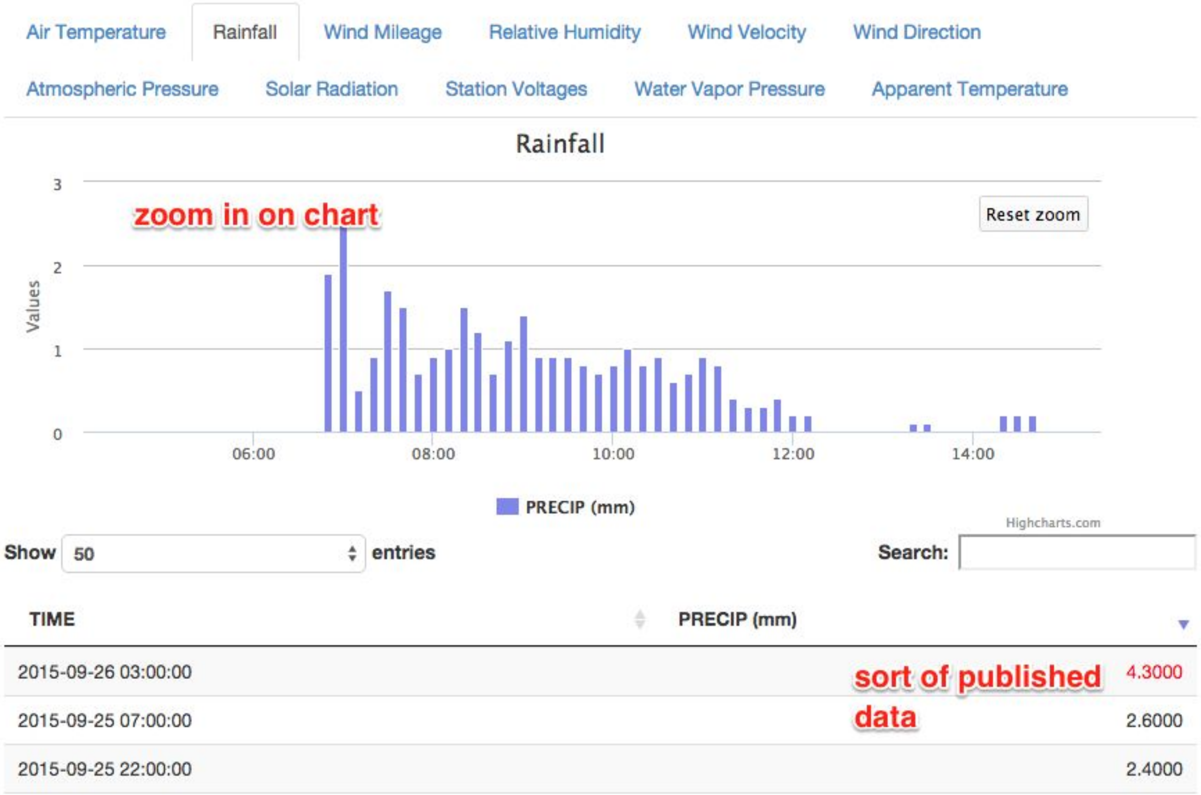
Hourly Report by Station

Station: PGIA Campbell - 9958303

Measurement Interval: 10m

Date Interval: 2015-09-25 - 2015-09-29

Charts and Reports Samples



zoom in on chart

TIME	PRECIP (mm)
2015-09-26 03:00:00	4.3000
2015-09-25 07:00:00	2.6000
2015-09-25 22:00:00	2.4000

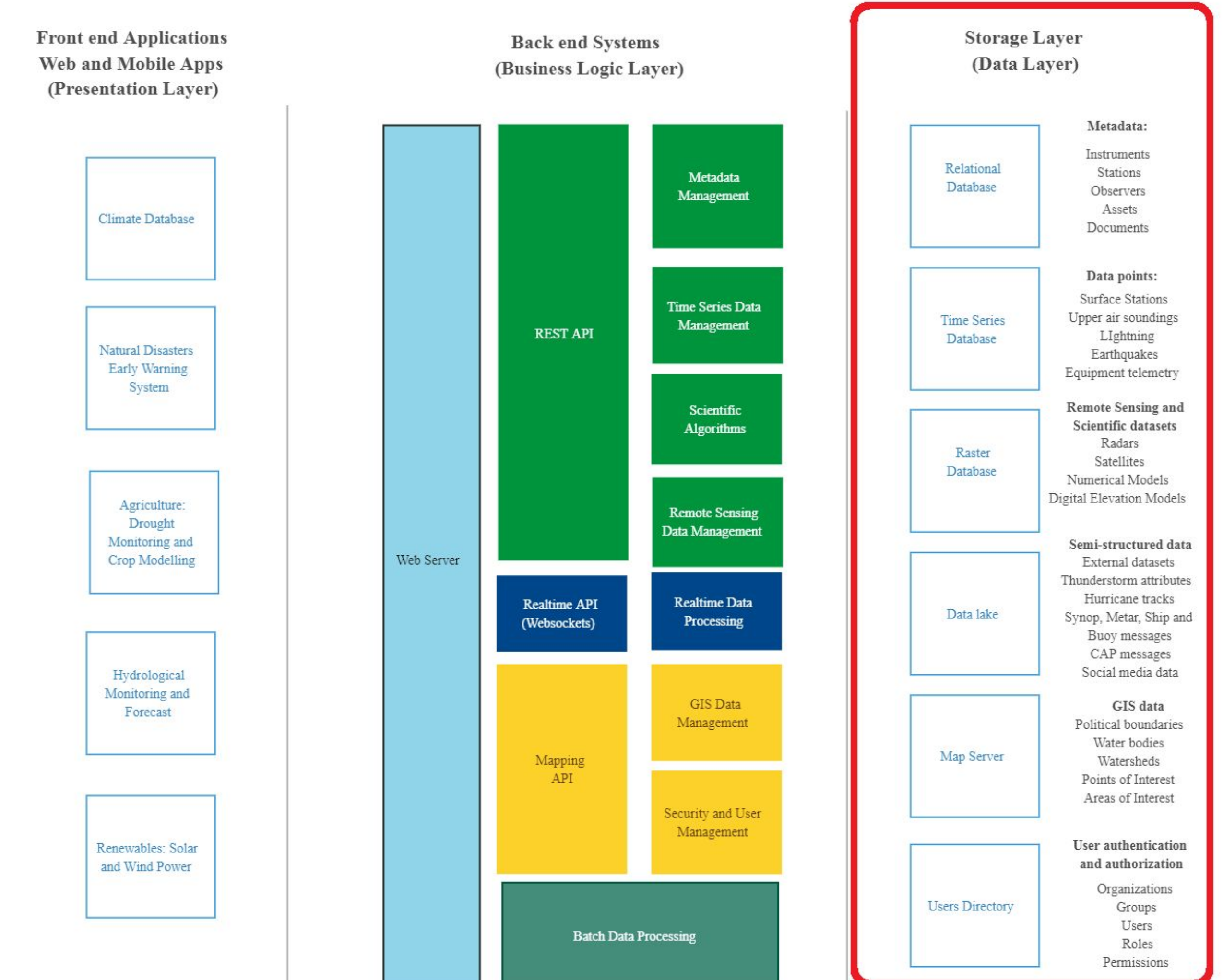
sort of published data

Climate database background...

- Our idea all along was to have a system/application that could store a wide range of observational data e.g:
    - Weather Stations
    - Radar
    - Weather Satellites - Geostationary and Polar Orbiting
    - lightning detection Network
- From this data set, Weather and Climate patterns could be derived and compared to short term and long term forecasts.

Climate database background...

- Over a period of time the NMS as well as its stakeholders would have large data sets that could be used in conjunction with stakeholder specific algorithms to create tools that can assist with decision making
- Our Goal is to help the various sectors(in Belize) affected by weather and climate to develop tools to help them make daily, weekly, monthly decisions.
- Visually this would look like this...



Front end Applications
Web and Mobile Apps
(Presentation Layer)

Climate Database

Natural Disasters Early Warning System

Agriculture: Drought Monitoring and Crop Modelling

Hydrological Monitoring and Forecast

Renewables: Solar and Wind Power

Back end Systems
(Business Logic Layer)

Web Server

REST API

Metadata Management

Time Series Data Management

Scientific Algorithms

Remote Sensing Data Management

Realtime API (Websockets)

Realtime Data Processing

Mapping API

GIS Data Management

Security and User Management

Batch Data Processing

Storage Layer
(Data Layer)

Relational Database

Metadata:
Instruments
Stations
Observers
Assets
Documents

Time Series Database

Data points:
Surface Stations
Upper air soundings
LIghtning
Earthquakes
Equipment telemetry

Raster Database

Remote Sensing and Scientific datasets
Radars
Satellites
Numerical Models
Digital Elevation Models

Data lake

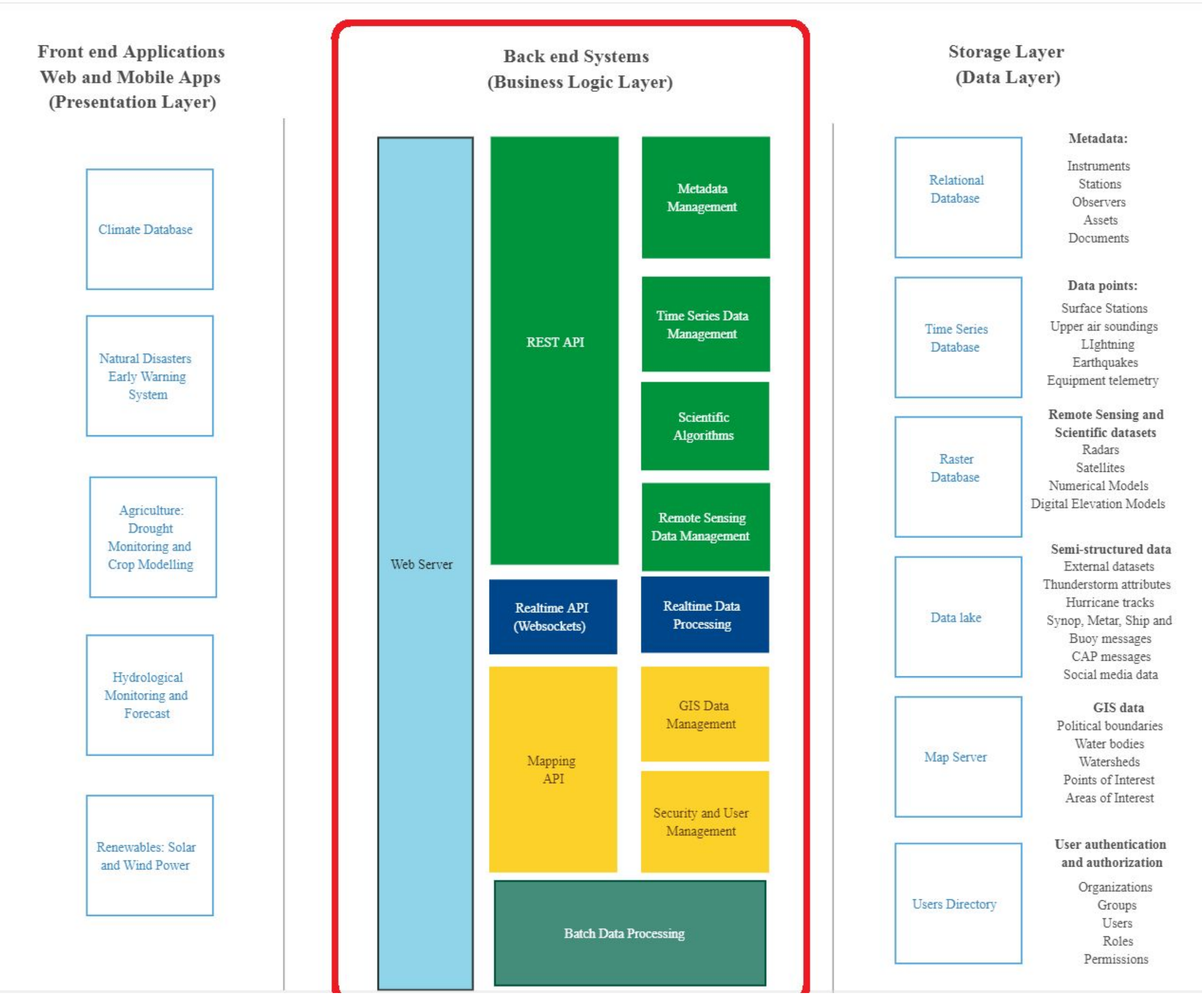
Semi-structured data
External datasets
Thunderstorm attributes
Hurricane tracks
Synop, Metar, Ship and Buoy messages
CAP messages
Social media data

Map Server

GIS data
Political boundaries
Water bodies
Watersheds
Points of Interest
Areas of Interest

Users Directory

User authentication and authorization
Organizations
Groups
Users
Roles
Permissions



Front end Applications Web and Mobile Apps (Presentation Layer)

Climate Database

Natural Disasters Early Warning System

Agriculture: Drought Monitoring and Crop Modelling

Hydrological Monitoring and Forecast

Renewables: Solar and Wind Power

Back end Systems (Business Logic Layer)

Web Server

REST API

Metadata Management

Time Series Data Management

Scientific Algorithms

Remote Sensing Data Management

Realtime API (Websockets)

Realtime Data Processing

Mapping API

GIS Data Management

Security and User Management

Batch Data Processing

Storage Layer (Data Layer)

Relational Database

Metadata:
Instruments
Stations
Observers
Assets
Documents

Time Series Database

Data points:
Surface Stations
Upper air soundings
LIghtning
Earthquakes
Equipment telemetry

Raster Database

Remote Sensing and Scientific datasets
Radars
Satellites
Numerical Models
Digital Elevation Models

Data lake

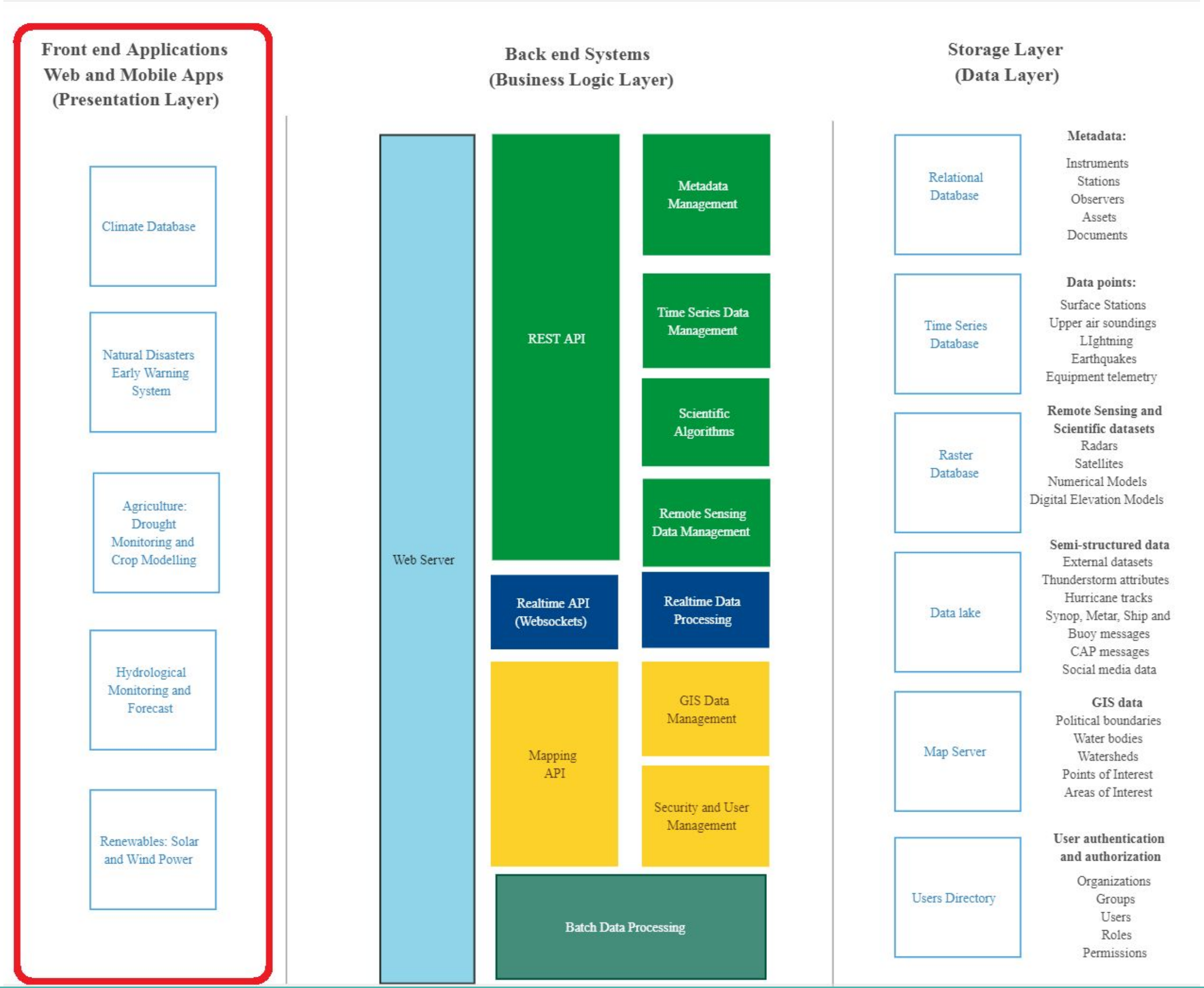
Semi-structured data
External datasets
Thunderstorm attributes
Hurricane tracks
Synop, Metar, Ship and Buoy messages
CAP messages
Social media data

Map Server

GIS data
Political boundaries
Water bodies
Watersheds
Points of Interest
Areas of Interest

Users Directory

User authentication and authorization
Organizations
Groups
Users
Roles
Permissions



Front end Applications Web and Mobile Apps (Presentation Layer)

Climate Database

Natural Disasters Early Warning System

Agriculture: Drought Monitoring and Crop Modelling

Hydrological Monitoring and Forecast

Renewables: Solar and Wind Power

Back end Systems (Business Logic Layer)

Web Server

REST API

Metadata Management

Time Series Data Management

Scientific Algorithms

Remote Sensing Data Management

Realtime API (Websockets)

Realtime Data Processing

Mapping API

GIS Data Management

Security and User Management

Batch Data Processing

Storage Layer (Data Layer)

Relational Database

Metadata:
Instruments
Stations
Observers
Assets
Documents

Time Series Database

Data points:
Surface Stations
Upper air soundings
LIghtning
Earthquakes
Equipment telemetry

Raster Database

Remote Sensing and Scientific datasets
Radars
Satellites
Numerical Models
Digital Elevation Models

Data lake

Semi-structured data
External datasets
Thunderstorm attributes
Hurricane tracks
Synop, Metar, Ship and Buoy messages
CAP messages
Social media data

Map Server

GIS data
Political boundaries
Water bodies
Watersheds
Points of Interest
Areas of Interest

Users Directory

User authentication and authorization
Organizations
Groups
Users
Roles
Permissions

Limitations of HydrometDB

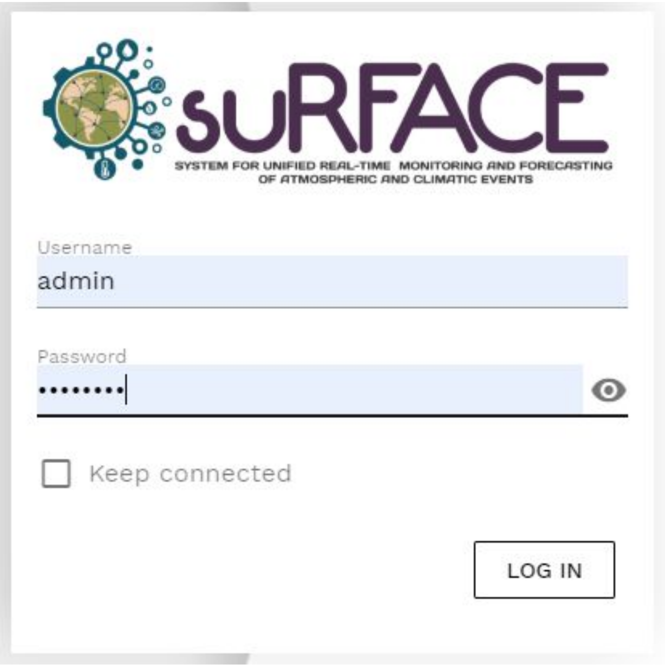
- Timeseries data storage and access ineffiecient(using relational database)
- Background processing in PHP(lack scientific libraries)
- PHP lack ability to process large amount of data quickly
- Lack robust API for easy interface with stakeholders
- Quality Control module inoperable

JCCCP improvements made in 2019

- Improvements to system architecture, reliability and data security by leveraging reliable, stable and well supported open source technologies
- Refactoring of the storage layer to use a time series database to store Hydrological and Meteorological datasets
- Refactoring of the current source code to a modular design using Python/django framework/REST API ;
- Improvements to the Quality Control module to include manual validation procedures;
- Support processing of hydrological measurements and automatic computation of stream flows;
- New Rack mount Server

CDMS Version 2 called “SURFACE”

System for Unified Real-time monitoring and Forcasting of Atmospheric and Climatic Events



sURFACE

SYSTEM FOR UNIFIED REAL-TIME MONITORING AND FORECASTING OF ATMOSPHERIC AND CLIMATIC EVENTS

Username: admin

Password: ••••••••

☐ Keep connected

System Architecture and Components

Five Basic System Components

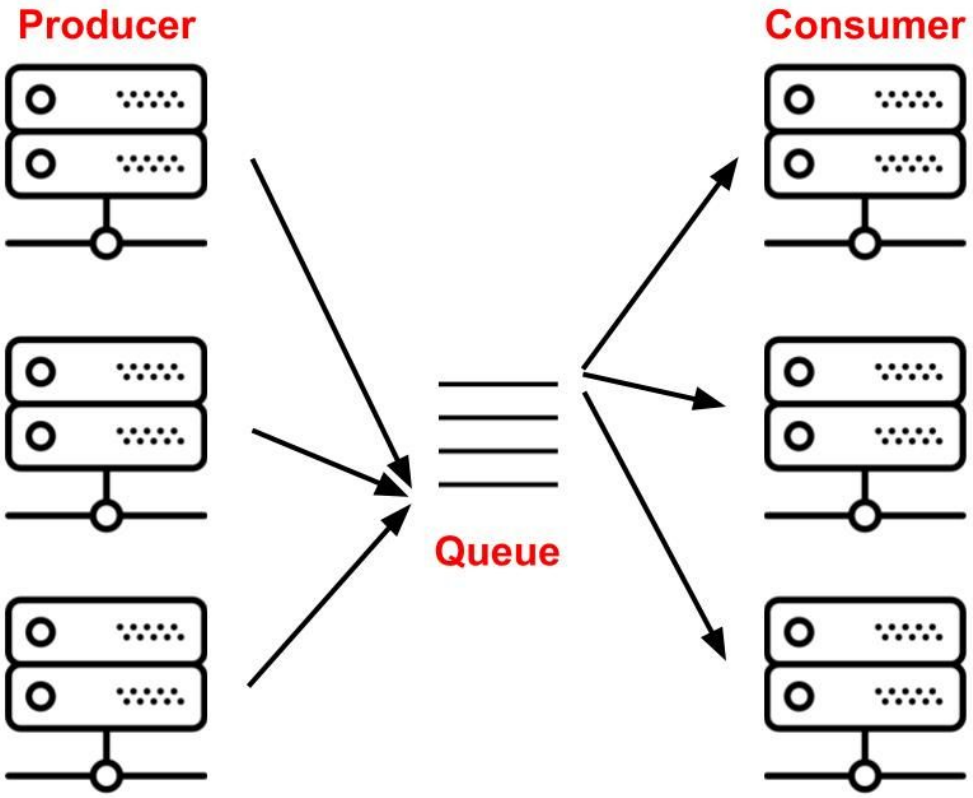
- Database - Stores Data - **PostgreSQL**
- Workers - Processes Data - **Celery**
- Message Broker - Arranges a list of processes - **RabbitMQ**
- Web API - Making available backend data - **Python/Django/RESTFUL**
- Front End Interface - Displays Information from backend on browser - **Vue.js --Vuetify**

System Component - Database

- A newer version of **PostgreSQL** (version 10) was adopted together with PostGIS and TimescaleDB extensions.
- PostGIS is a popular extension that adds spatial capabilities to PostgreSQL, enabling the database to store and query spatial datasets.
- TimescaleDB is an open-source project that was recently released by MIT researchers and adds special features to PostgreSQL to manage time series data.

System Component - Message and Worker

- To enable parallel processing for the heavier tasks, such as importing and exporting large datasets of measurement data, RabbitMQ and Celery where adopted for the development of the message broker subsystem.
- RabbitMQ is an open source message queue based on Erlang, a technology widely adopted for the development of fault-tolerant distributed systems infrastructure.


Producer
Consumer
Queue

System Component - Worker(Celery)

- Celery is an asynchronous task/job framework based on RabbitMQ and Python that integrates natively with Django.
- All tasks that needs to be processed in the backend were developed in the new system based on Celery workers.

System Component - Web API

- For the backend Django was adopted as the option for the development of the Web API.
- Django is a mature and popular web framework based on Python Programming language, and is considered one of the most productive frameworks available for backend web development.
- To support the development the REST based web API, an extension called "Django REST Framework" was added to Django.
- Django backend interface used as to administer application

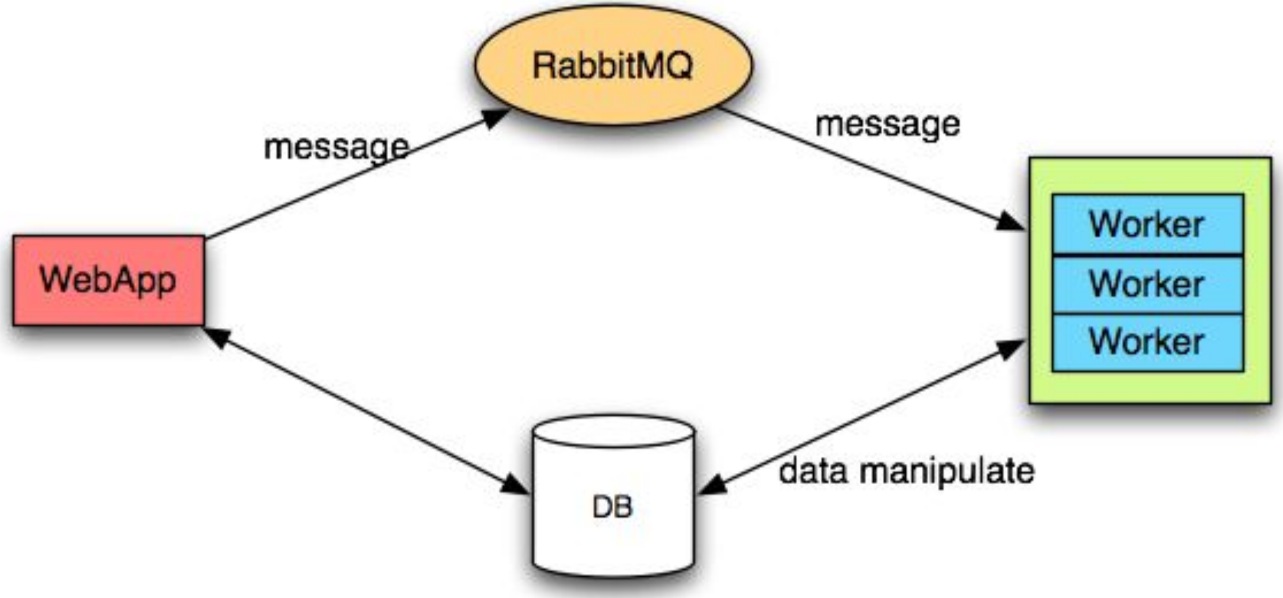
System Component - Frontend

- Vue.js is a modern Javascript framework for creating modern Single Page Applications (applications like Gmail) and enables the development of user interfaces that deliver better user experience and usability
- Vuetify is a library for Vue.js with a set of high quality user interface graphic components based on Google's Material Design standard.
- Vuetify also provides important features for data input validation and presenting user interfaces in devices with very different screen resolutions (smartphones, tablets, desktop computers).

Detailed System Overview

Vue.js

Vue + Vuetify



Installation

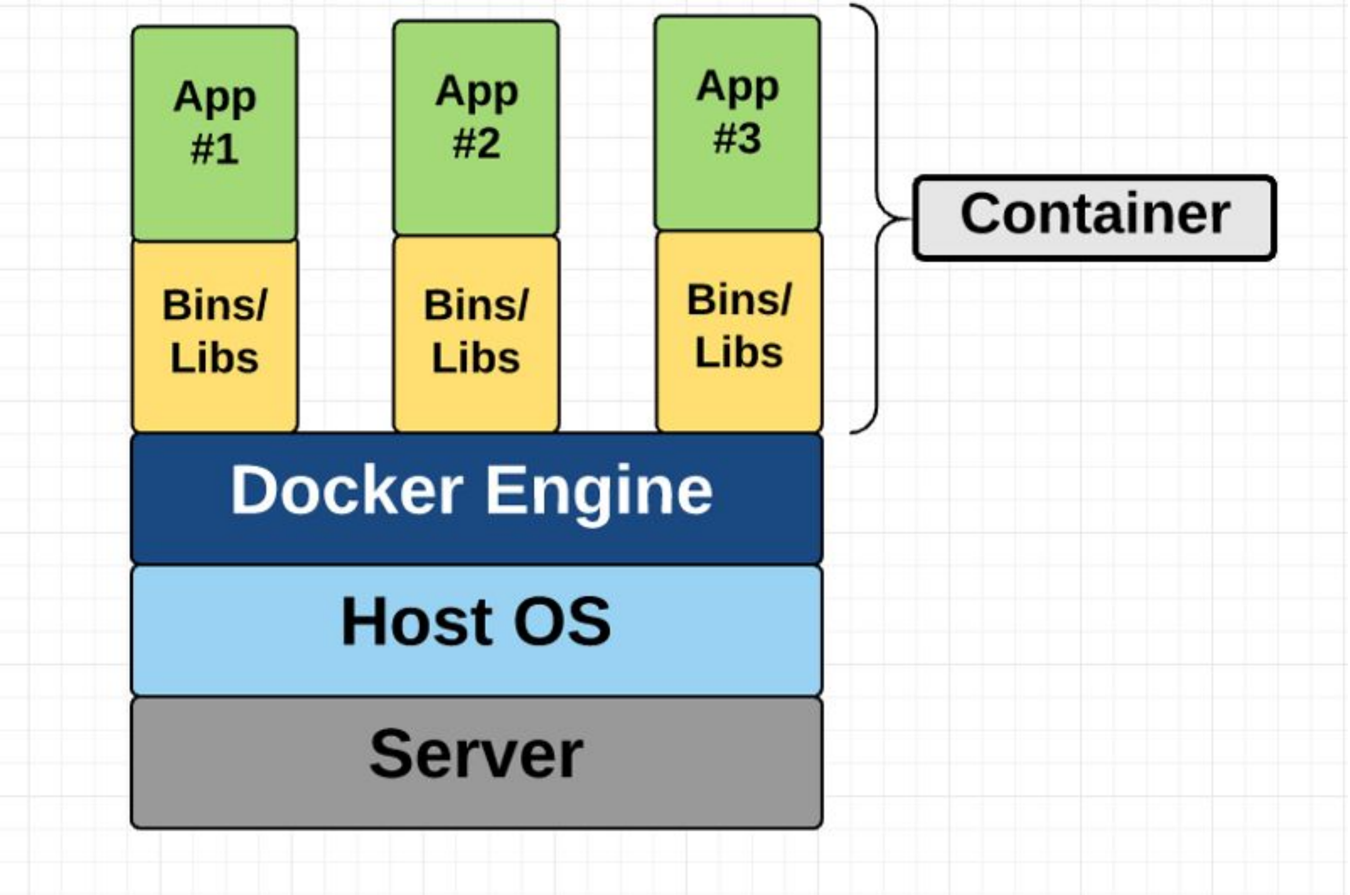
- A totally new approach was selected for the deployment of this version of the system into production.
- One of the main complaints with the old system was the complexity involved in all the steps required for building and deploying the CDMS into a new hardware.
- To overcome these problems, a container-based solution was structured based on Docker containers.

Installation and update - Docker

- Docker is an open source software platform that allows packaging of applications into portable containers that can be easily deployed into Windows, Linux and MacOS operating systems.
- Using this approach, each subsystem of the new CDMS was packaged into individual containers where all the files required for building and integrating the Docker images of the database, message broker, web api, workers, and frontend are located in the project's source code repository.

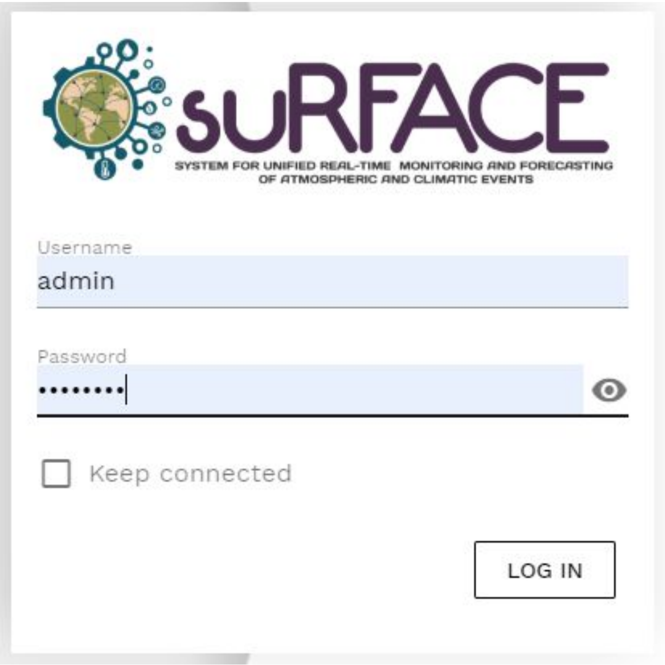
Installation and update - Docker

- With the source code, creating a new development environment to maintain or extend the system, and deploying it to a new hardware is a trivial task achieved by running a single docker-compose command in the terminal.
- This approach also allows synchronizing updates in the source code from the repository to the server using regular git commands, making it easier to apply bug fixes to the system.


App #1
App #2
App #3
Container
Bins/ Libs
Bins/ Libs
Bins/ Libs
Docker Engine
Host OS
Server

System Screen Shots - login Screen

- Ability to ingest data from multiple sources
  - Stations, Radar, lightning detection network
  - Handles AWS data aggregations
- Monitors Station performance & data flow
- Metadata for Stations & Instruments
- Uses Restful API(Open Source)
- Python/Django Backend(Open Source)
- Modular design using Docker(Open source)

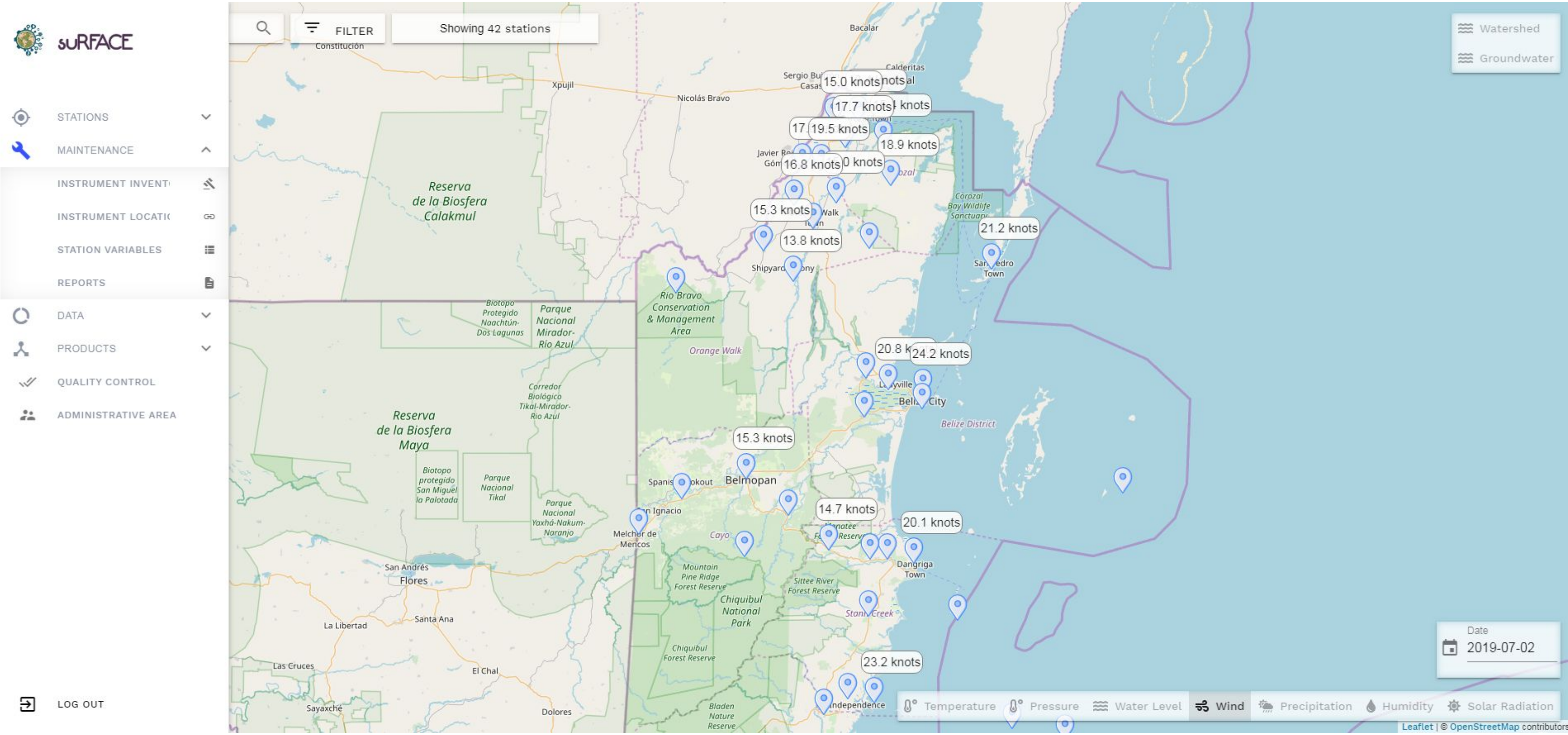

sURFACE
SYSTEM FOR UNIFIED REAL-TIME MONITORING AND FORECASTING OF ATMOSPHERIC AND CLIMATIC EVENTS
LOG IN

Username
admin

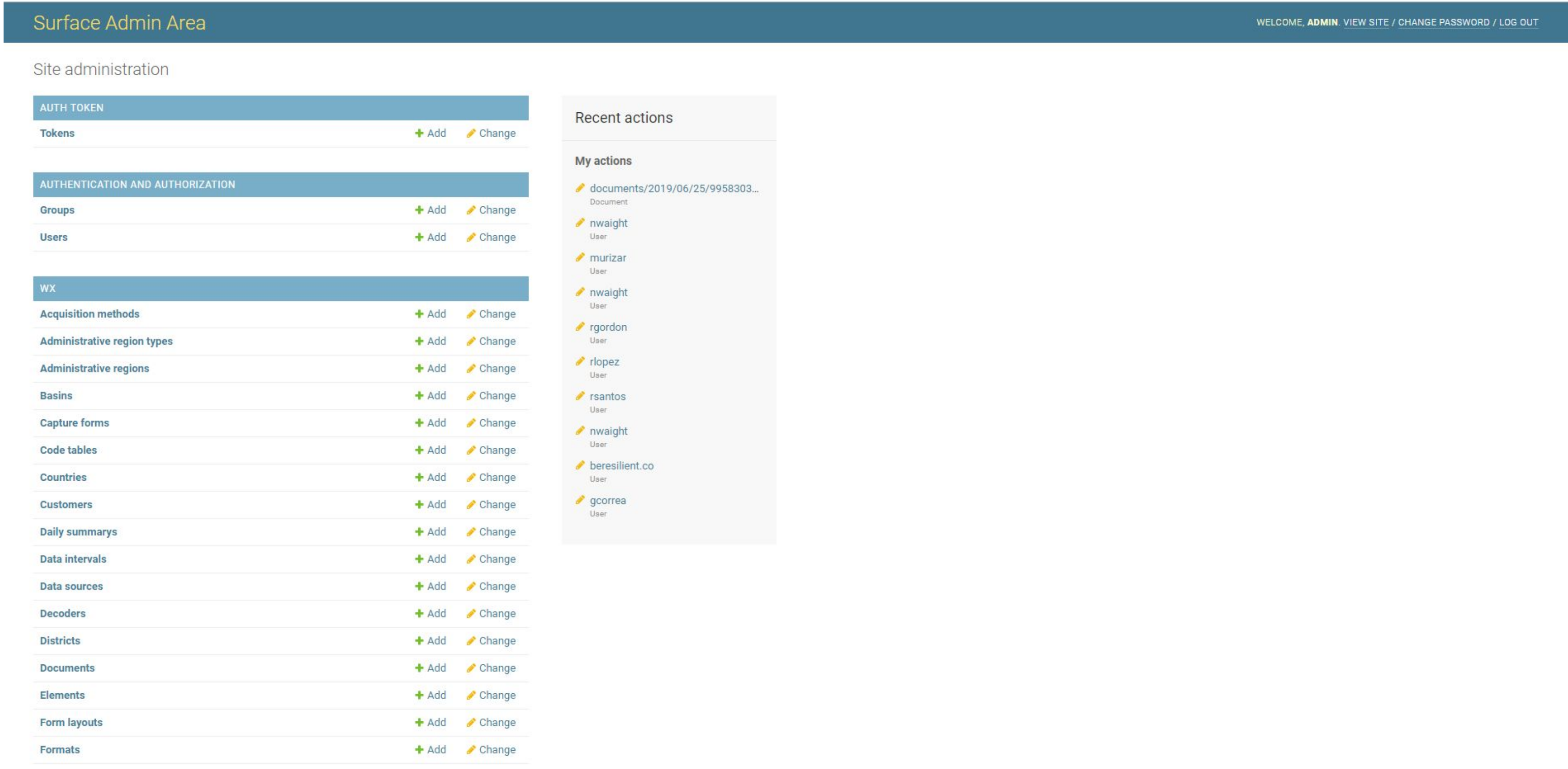
Password
•••••••••

□ Keep connected

Map Display


sURFACE
STATIONS
MAINTENANCE
INSTRUMENT INVENT
INSTRUMENT LOCATI
STATION VARIABLES
REPORTS
DATA
PRODUCTS
QUALITY CONTROL
ADMINISTRATIVE AREA
LOG OUT
FILTER
Showing 42 stations
Watershed
Groundwater
Date
2019-07-02
Temperature
Pressure
Water Level
Wind
Precipitation
Humidity
Solar Radiation
15.0 knots
17.7 knots
19.5 knots
18.9 knots
16.8 knots
15.3 knots
13.8 knots
21.2 knots
20.8 k
24.2 knots
15.3 knots
14.7 knots
20.1 knots
23.2 knots
Bacalar
Xpujil
Nicolás Bravo
Reserva de la Biosfera Calakmul
Reserva de la Biosfera Maya
Rio Bravo Conservation & Management Area
Orange Walk
Belmopan
Belize District
Cayo
Dangriga Town
Stann Creek
Chiquibul National Park
Flores
Leaflet | © OpenStreetMap contributors

django Backend Interface



Surface Admin Area

WELCOME, **ADMIN**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

AUTH TOKEN		
Tokens	✚ Add	✎ Change

AUTHENTICATION AND AUTHORIZATION		
Groups	✚ Add	✎ Change
Users	✚ Add	✎ Change

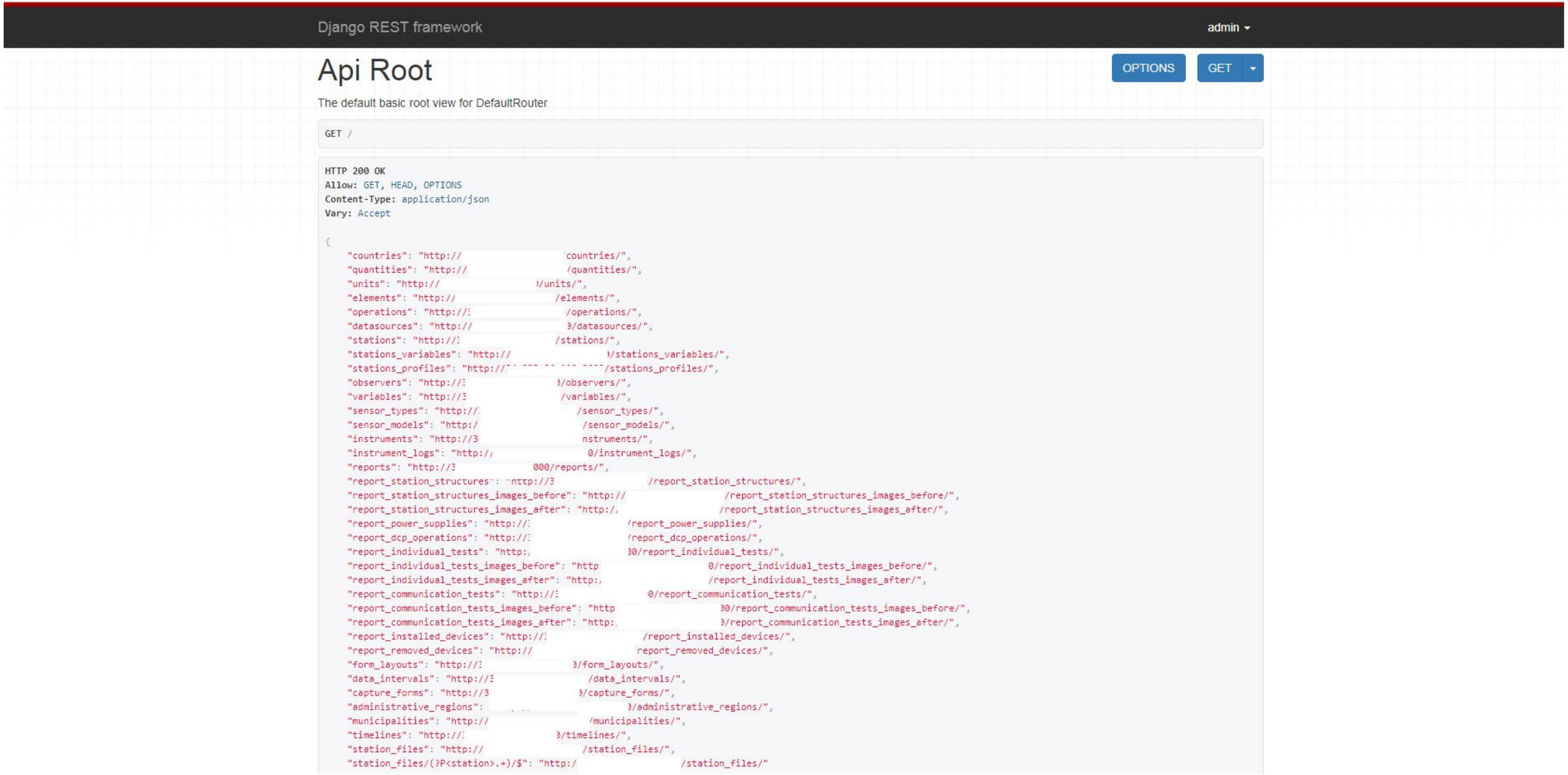
WX		
Acquisition methods	✚ Add	✎ Change
Administrative region types	✚ Add	✎ Change
Administrative regions	✚ Add	✎ Change
Basins	✚ Add	✎ Change
Capture forms	✚ Add	✎ Change
Code tables	✚ Add	✎ Change
Countries	✚ Add	✎ Change
Customers	✚ Add	✎ Change
Daily summarys	✚ Add	✎ Change
Data intervals	✚ Add	✎ Change
Data sources	✚ Add	✎ Change
Decoders	✚ Add	✎ Change
Districts	✚ Add	✎ Change
Documents	✚ Add	✎ Change
Elements	✚ Add	✎ Change
Form layouts	✚ Add	✎ Change
Formats	✚ Add	✎ Change

Recent actions

My actions

- ✎ documents/2019/06/25/9958303…
  Document
- ✎ nwaight
  User
- ✎ murizar
  User
- ✎ nwaight
  User
- ✎ rgordon
  User
- ✎ rlopez
  User
- ✎ rsantos
  User
- ✎ nwaight
  User
- ✎ beresilient.co
  User
- ✎ gcorrea
  User

Rest API end points



Api Root

The default basic root view for DefaultRouter

```
GET /
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "countries": "http://[redacted]countries/",
    "quantities": "http://[redacted]/quantities/",
    "units": "http://[redacted]/units/",
    "elements": "http://[redacted]/elements/",
    "operations": "http://[redacted]/operations/",
    "datasources": "http://[redacted]/datasources/",
    "stations": "http://[redacted]/stations/",
    "stations_variables": "http://[redacted]/stations_variables/",
    "stations_profiles": "http://[redacted]/stations_profiles/",
    "observers": "http://[redacted]/observers/",
    "variables": "http://[redacted]/variables/",
    "sensor_types": "http://[redacted]/sensor_types/",
    "sensor_models": "http:/[redacted]/sensor_models/",
    "instruments": "http://[redacted]nstruments/",
    "instrument_logs": "http:/[redacted]0/instrument_logs/",
    "reports": "http://[redacted]000/reports/",
    "report_station_structures": "http://[redacted]/report_station_structures/",
    "report_station_structures_images_before": "http://[redacted]/report_station_structures_images_before/",
    "report_station_structures_images_after": "http:/[redacted]/report_station_structures_images_after/",
    "report_power_supplies": "http://[redacted]/report_power_supplies/",
    "report_dcp_operations": "http://[redacted]/report_dcp_operations/",
    "report_individual_tests": "http:[redacted]0/report_individual_tests/",
    "report_individual_tests_images_before": "http[redacted]0/report_individual_tests_images_before/",
    "report_individual_tests_images_after": "http:[redacted]/report_individual_tests_images_after/",
    "report_communication_tests": "http://[redacted]0/report_communication_tests/",
    "report_communication_tests_images_before": "http[redacted]0/report_communication_tests_images_before/",
    "report_communication_tests_images_after": "http:[redacted]/report_communication_tests_images_after/",
    "report_installed_devices": "http://[redacted]/report_installed_devices/",
    "report_removed_devices": "http://[redacted]report_removed_devices/",
    "form_layouts": "http://[redacted]/form_layouts/",
    "data_intervals": "http://[redacted]/data_intervals/",
    "capture_forms": "http://[redacted]/capture_forms/",
    "administrative_regions": [redacted]/administrative_regions/",
    "municipalities": "http://[redacted]municipalities/",
    "timelines": "http://[redacted]/timelines/",
    "station_files": "http://[redacted]/station_files/",
    "station_files/(?P<station>.+)/$": "http:/[redacted]/station_files/"
```

Thank You