App for Microgrid Demonstration

Team: Gabe Rueger, Micheal Thai, Mickey Doyle, Patrick Shirazi, William Bronson
Client: ISU Electric Power Research Center
Advisor: Anne Kimber

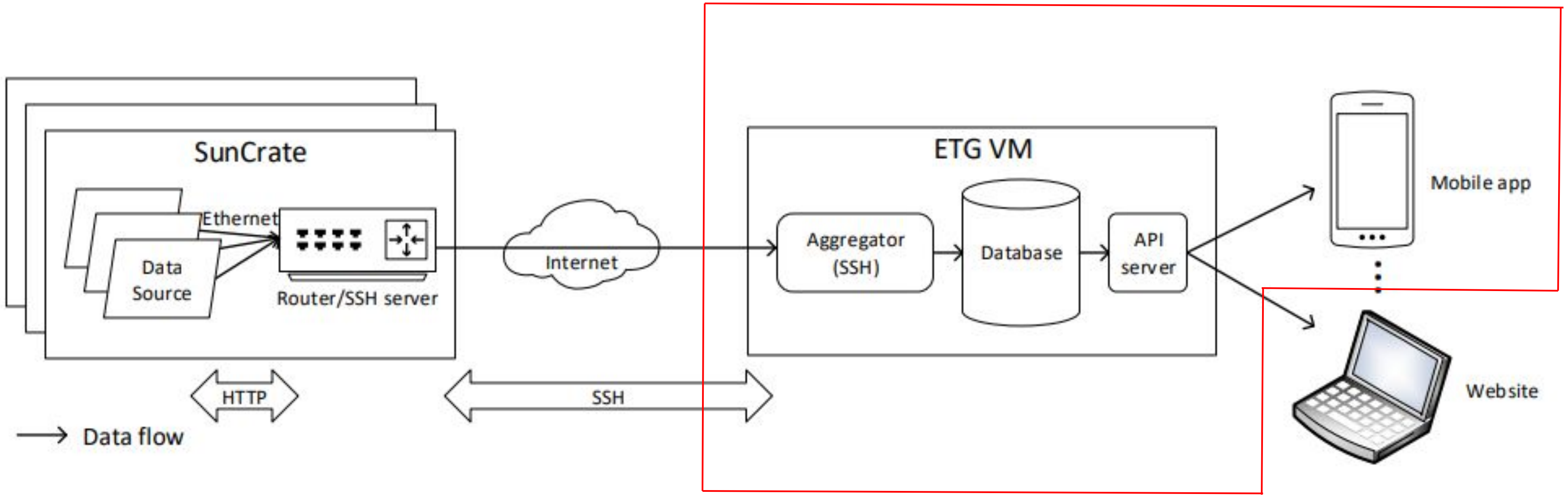
Project Plan

Problem Statement

- The ISU Electric Power Research Center has been researching on a solar crate with microgrid data, but there isn’t an effective way of accessing this data.
- Our solution was to create a mobile application to retrieve and present the energy data and serve as an efficient way for users to analyze the efficiency of the power collection.

NOTICE
DO NOT ENTER

Conceptual Sketch



Functional Requirements

- Ability to add additional crates
- Ability to add additional data sources
- Query and search different subsets of the data
- Configurable data collection interval
- Support automatic archiving of data

Non-functional Requirements

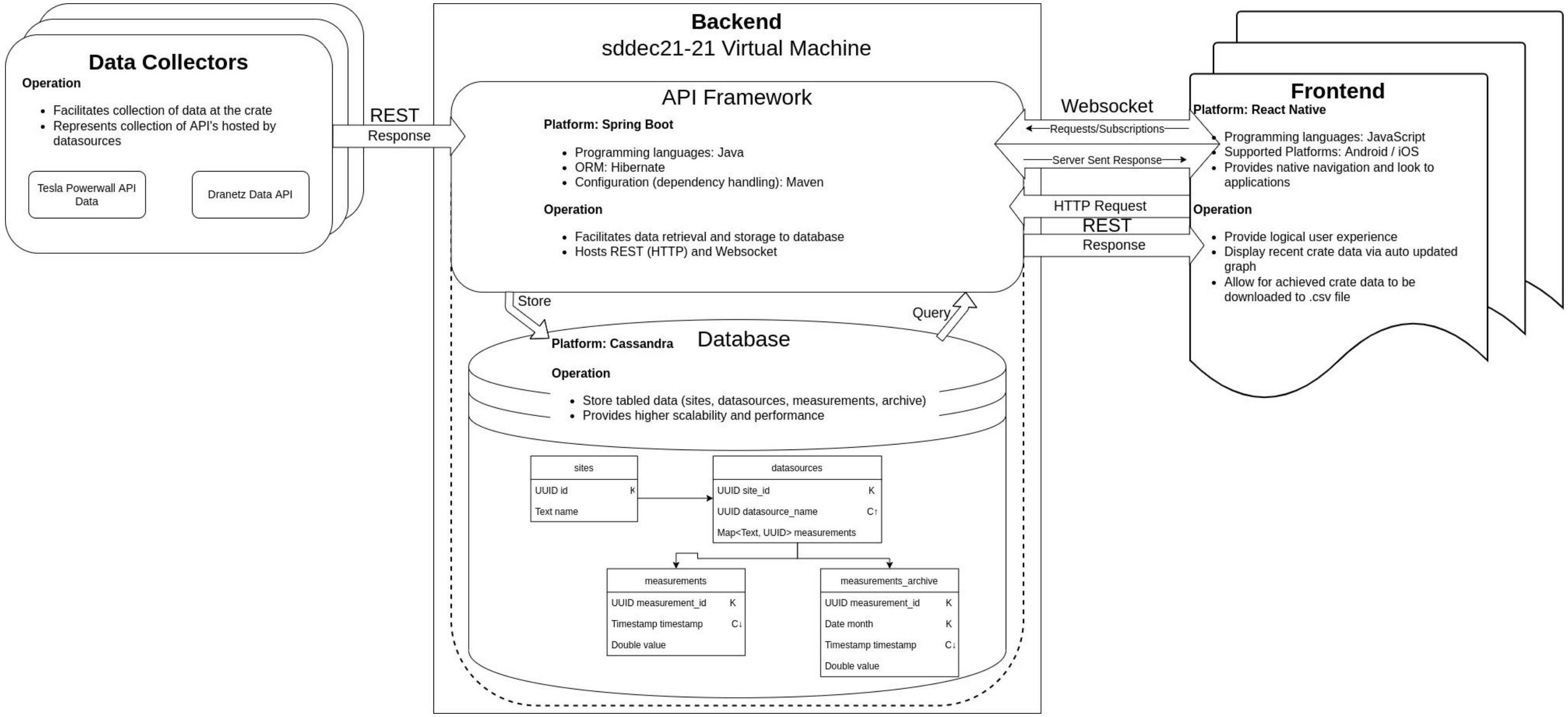
- Data must be displayed within a minute of collection
- Database size scales linearly with number of data sources and with time
- Old data reduced to 5 minute samples
- Frameworks, libraries, etc. must be well-supported and maintained
- Must use open and well-supported communication standards
- All decisions made throughout this project must be well documented

Constraints / Considerations

- Scalability - currently only one site, hopefully hundreds in the future
- Performance - want timely data
- Adaptability - new data sources can be added to sites
- Usability:
    - need to be user friendly for the public
    - researchers should be able to access voltage, current, and frequency data readings
    - cross platform

Design & Implementation

Functional Decomposition



Data Collectors

Operation

- Facilitates collection of data at the crate
- Represents collection of API's hosted by datasources

Tesla Powerwall API Data

Dranetz Data API

Backend

sddec21-21 Virtual Machine

API Framework

Platform: Spring Boot

- Programming languages: Java
- ORM: Hibernate
- Configuration (dependency handling): Maven

Operation

- Facilitates data retrieval and storage to database
- Hosts REST (HTTP) and Websocket

Database

Platform: Cassandra

Operation

- Store tabled data (sites, datasources, measurements, archive)
- Provides higher scalability and performance

sites	
UUID id	K
Text name	

datasources	
UUID site_id	K
UUID datasource_name	C↑
Map<Text, UUID> measurements	

measurements	
UUID measurement_id	K
Timestamp timestamp	C↓
Double value	

measurements_archive	
UUID measurement_id	K
Date month	K
Timestamp timestamp	C↓
Double value	

Frontend

Platform: React Native

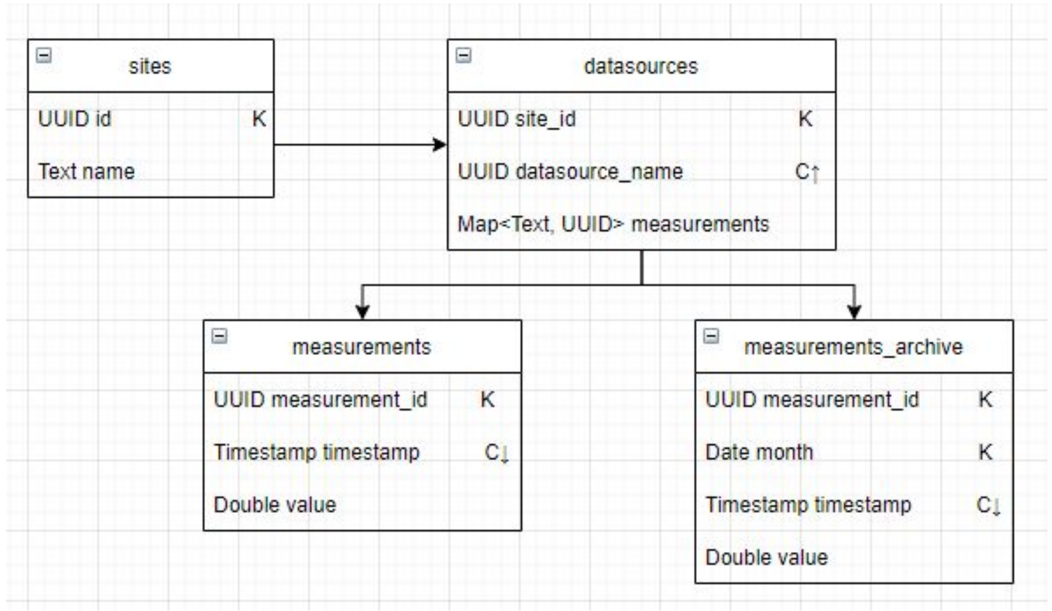
- Programming languages: JavaScript
- Supported Platforms: Android / iOS
- Provides native navigation and look to applications

Operation

- Provide logical user experience
- Display recent crate data via auto updated graph
- Allow for achieved crate data to be downloaded to .csv file

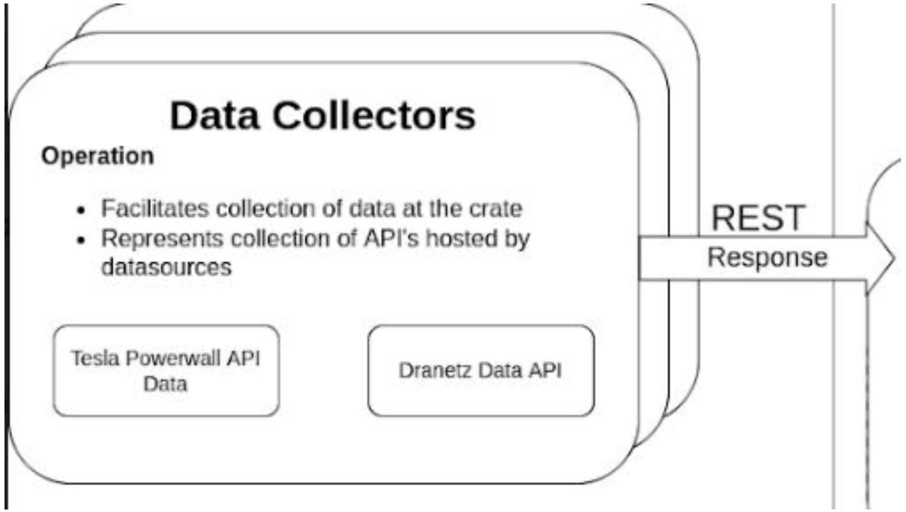
Database

- Apache Cassandra
- Frequent writes
- Lots of data
- Regular datasource and measurement changes
- Archiving system



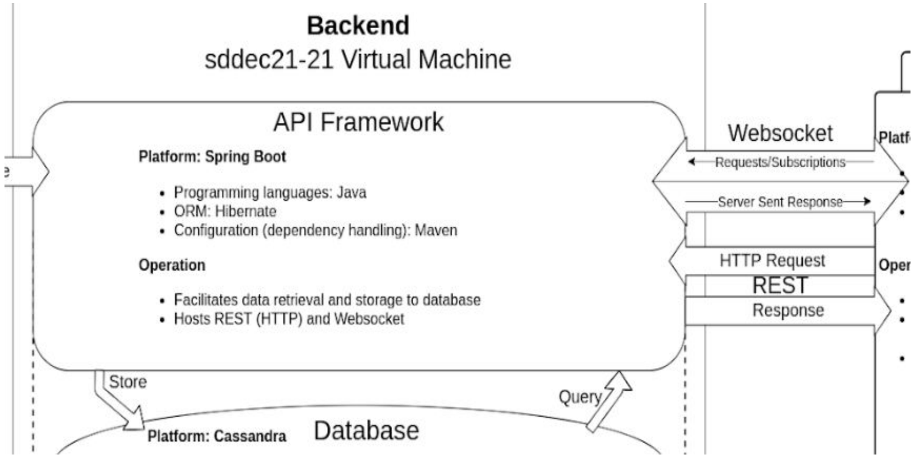
Data Collectors

- Individual python scripts
- Access datasource through APIs or other means
- Parses and packages data to be sent to server
- Run and go design


Data Collectors
Operation
• Facilitates collection of data at the crate
• Represents collection of API's hosted by datasources
Tesla Powerwall API Data
Dranetz Data API
REST
Response

Spring Boot Server

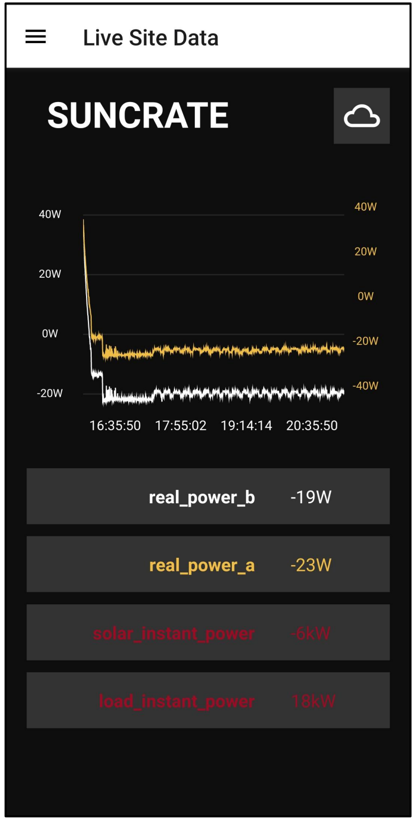
- Establish communication between modules
    - Hosts REST API and Websocket
- Objectify data for easy use



Backend
sddec21-21 Virtual Machine
API Framework
Platform: Spring Boot
Programming languages: Java
ORM: Hibernate
Configuration (dependency handling): Maven
Operation
Facilitates data retrieval and storage to database
Hosts REST (HTTP) and Websocket
Websocket
Requests/Subscriptions
Server Sent Response
HTTP Request
REST
Response
Store
Query
Platform: Cassandra
Database

Frontend

- Using React Native
  - Cross-platform
  - Good support
  - Free
- Allow users to visualize live data from a site
  - Lots of data to process
  - Should work cross-platform
  - < 1 min turnaround for displaying live data
- Allow users to access past data from a site
  - Should be able to select data over longer time period (several weeks)
- Everything needs to scale to additional sites



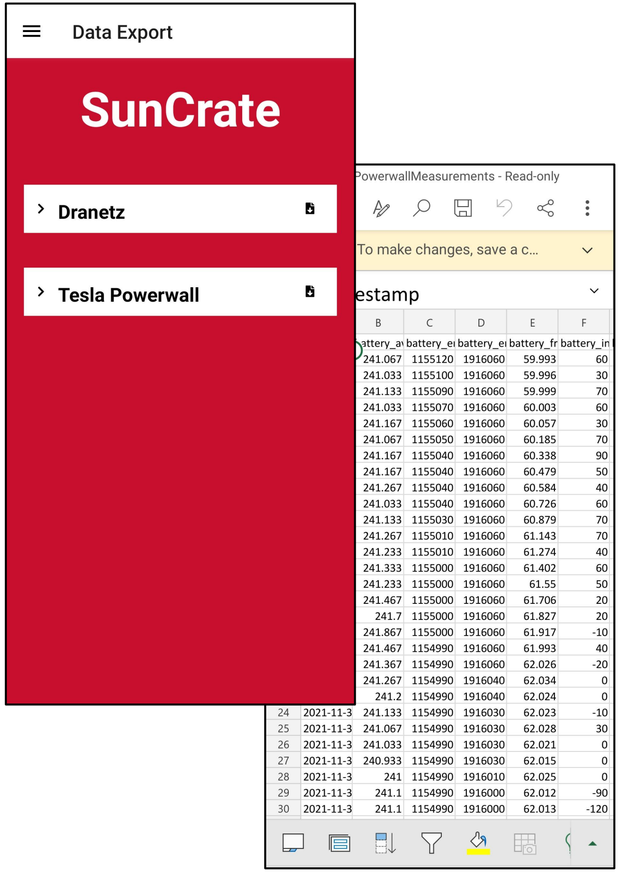
Live Site Data

SUNCRATE

real_power_b	-19W
real_power_a	-23W
solar_instant_power	-6kW
load_instant_power	18kW

Frontend Implementation

- Websocket to stream live data from the server
  - Displayed on a graph
  - Very short turnaround time
- HTTP requests
  - Get available sites
  - Get available data sources
  - Access recent data from a site (several hours)
  - Access archived data from a site (several weeks)
    - Saved as a file
- Location data
  - Select site by location
  - Weather info
  - Solar potential


Data Export
SunCrate
Dranetz
Tesla Powerwall
PowerwallMeasurements - Read-only
To make changes, save a c...
estamp

		B	C	D	E	F
		attery_a	battery_e	battery_e	battery_fr	battery_in
		241.067	1155120	1916060	59.993	60
		241.033	1155100	1916060	59.996	30
		241.133	1155090	1916060	59.999	70
		241.033	1155070	1916060	60.003	60
		241.167	1155060	1916060	60.057	30
		241.067	1155050	1916060	60.185	70
		241.167	1155040	1916060	60.338	90
		241.167	1155040	1916060	60.479	50
		241.267	1155040	1916060	60.584	40
		241.033	1155040	1916060	60.726	60
		241.133	1155030	1916060	60.879	70
		241.267	1155010	1916060	61.143	70
		241.233	1155010	1916060	61.274	40
		241.333	1155000	1916060	61.402	60
		241.233	1155000	1916060	61.55	50
		241.467	1155000	1916060	61.706	20
		241.7	1155000	1916060	61.827	20
		241.867	1155000	1916060	61.917	-10
		241.467	1154990	1916060	61.993	40
		241.367	1154990	1916060	62.026	-20
		241.267	1154990	1916040	62.034	0
		241.2	1154990	1916040	62.024	0
24	2021-11-3	241.133	1154990	1916030	62.023	-10
25	2021-11-3	241.067	1154990	1916030	62.028	30
26	2021-11-3	241.033	1154990	1916030	62.021	0
27	2021-11-3	240.933	1154990	1916030	62.015	0
28	2021-11-3	241	1154990	1916010	62.025	0
29	2021-11-3	241.1	1154990	1916000	62.012	-90
30	2021-11-3	241.1	1154990	1916000	62.013	-120

Testing

- Unit and Integration Testing
  - JUnit, Mockito
  - Jest, Enzyme
- Acceptance Testing
  - Client Demos
- 1 minute data pass through requirement
  - Easily achieved by the system
  - At most ~10 seconds

Demo

Questions?