Selena: a Serverless Energy Management System

Florian Huber Nikolai Koerber Markus Mock

Motivation

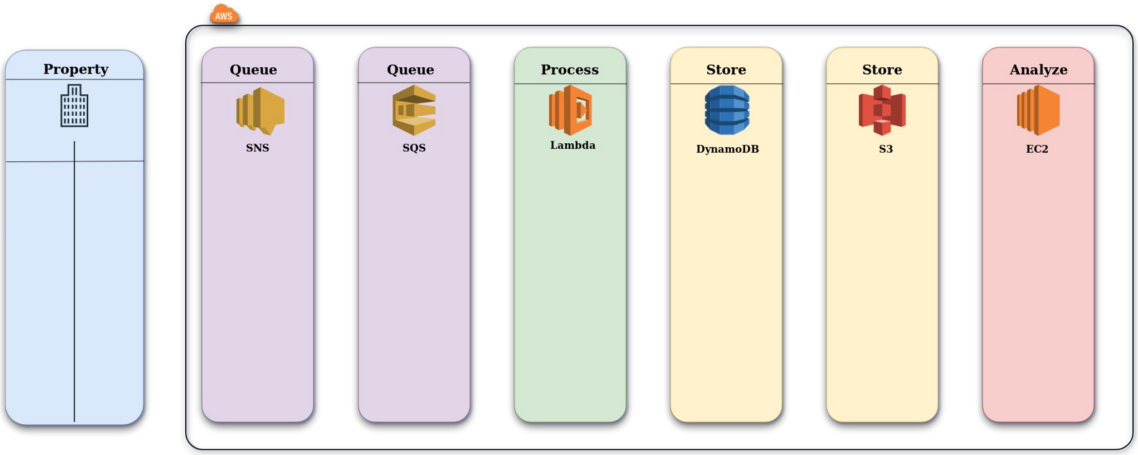
- Reduction of CO2
- One third due to buildings
- Energy management system(EMS) quickly become mandatory

Motivation

- **Extensibility**
- **Scalability**
- **Maintainability**

Key Components:

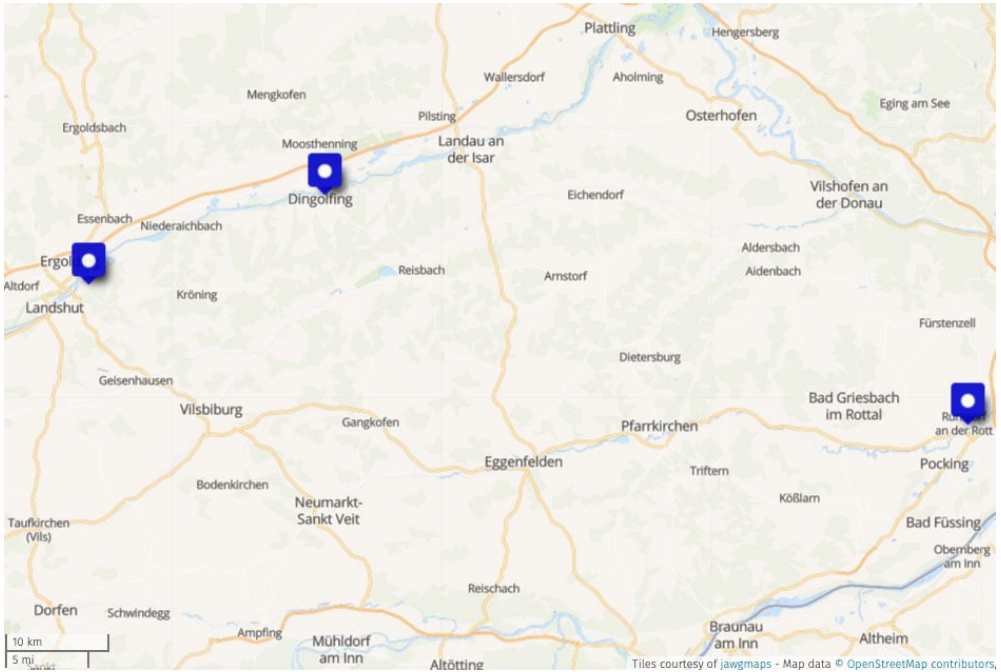
- **Property**
  - Physical site
  - Data-producer
- **AWS**
  - Queue
  - Process
  - Store
  - Analyze



Selena architecture overview.

Property

- Resource
  - Electricity
  - Water
  - Weather
  - PV
  - Heat



Overview of sites whose energy-related data is currently managed in Selena

Data Representation

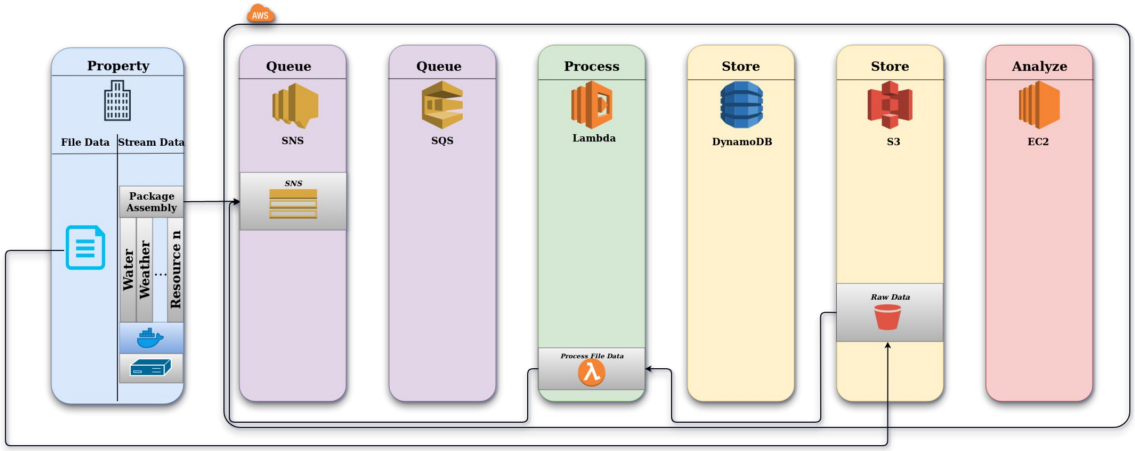
- Representation as JSON
- Thrift IDL
  - DataPaket
  - Data
  - Point
- Extensible
- Portable

```
struct Data{

    1: required i32 version,
    2: required string property,
    3: required i64 reading_ts,
    4: optional i64 sensor_ts,
    5: required string device_id,
    6: optional Resource resource,
    7: optional list<Point> point

}
```

Thrift Data struct implementation.

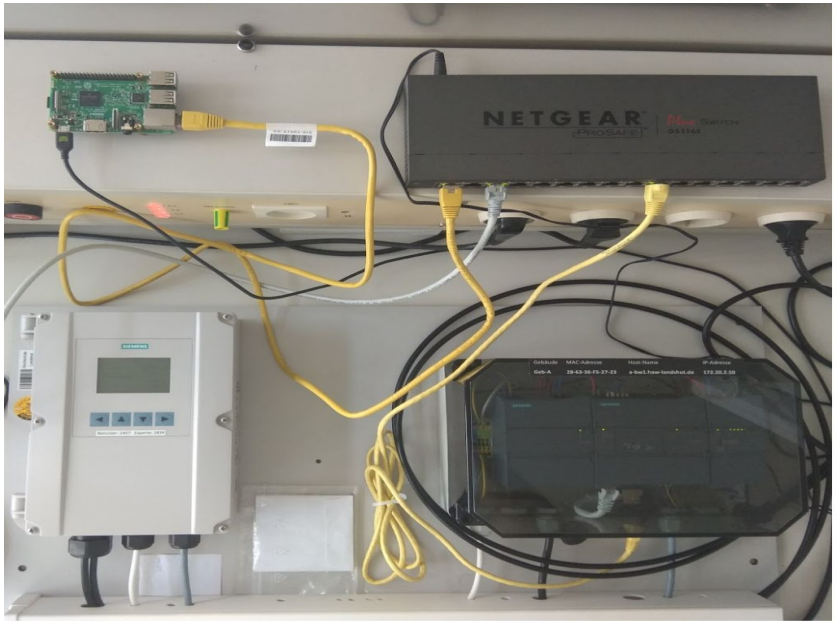
Ingress

- Stream Data
  - Concentrator
  - Raspberry Pi
- File Data
  - CSV
  - Excel



Selena architecture overview. Data produced at properties enter the system either via S3 bucket or through Selena's SNS endpoint.

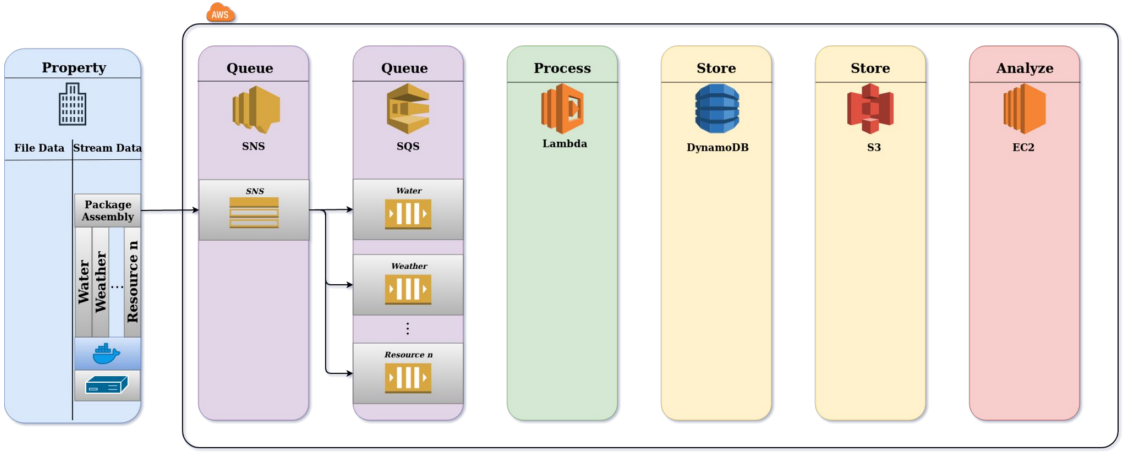
UAS Landshut



Hardware setup for the acquisition of water data at the UAS Landshut.

Queue

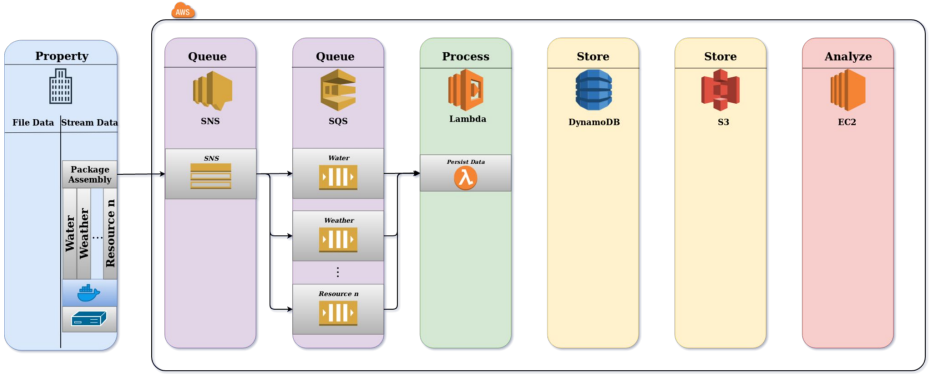
- **Simple Notification Service (SNS)**
- **Simple Queue Service (SQS)**
- **Single entrance point**
- **Resource as Topic**



SQS queues that are specific to the energy type, consume the data from the SNS queue.

Processing

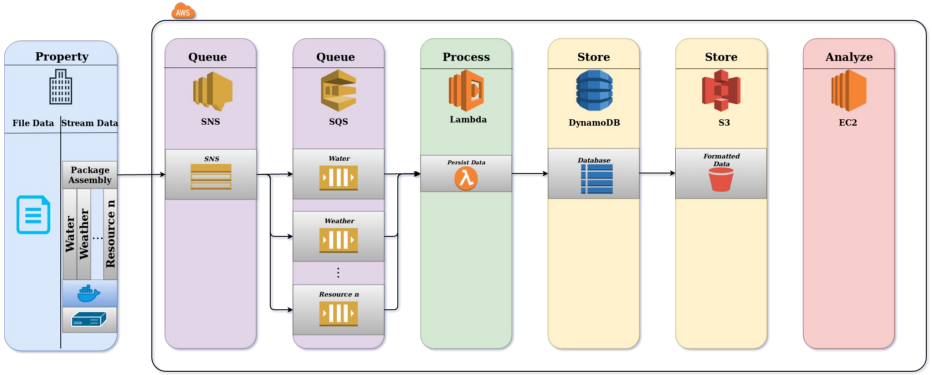
- Core component
- Single lambda function
- *“Persist Data”* function
- Code is only 50 lines



SQS queues which trigger Lambda functions that process and store the data.

Store

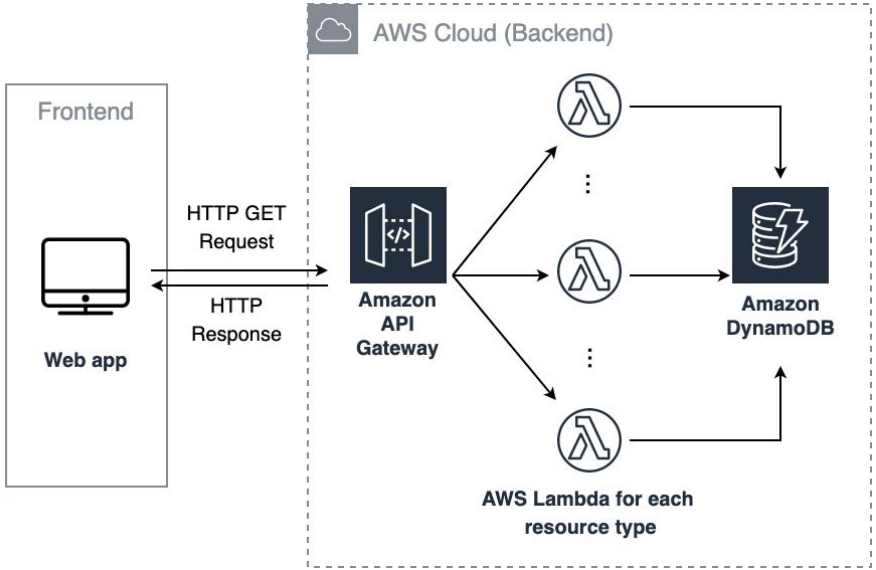
- DynamoDB as main storage
- Fast access
- Stream data
- Efficient storing in S3



AWS DynamoDB as hot data storage and s3 for the cold data.

Visualization

- **AWS API Gateway**
- **Microservice approach**
- **Graphical UI runs on EC2**

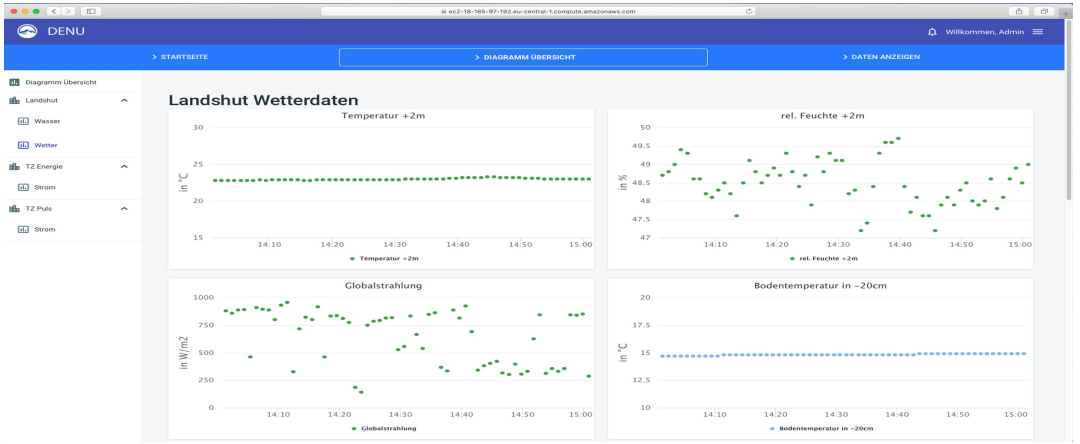


Selena front-to-backend communication

Frontend

- **Implementation based on:**
  - **React**
  - **Highcharts**

- **Different user roles**



Selena frontend dashboard. Visualization of a specific resource type and site.

Estimated Costs

- For the Ingress:
  - 60 data point/minute
  - Running 24/7

- 200 Device less than $25 per month

Conclusion

- Severless architecture
- Scales naturally
- Operational costs are low

Future work

- Front-end based on Zappa
- Implement data analysis
- Real-time alerting system

Acknowledgments

My special thanks go to Mr. Nikolai Koerber and Prof., PhD Markus Mock from the University of Applied Sciences Landshut.

Furthermore, we thank the Federal Ministry of Economics and Technology of Germany for supporting the research.

Thank You