BAM

BAM DIGITAL TWIN

TIME SERIES DATA SUBMODEL

Shady Abdou

TABLE OF CONTENT

- Project Overview
- Architecture Diagram
- Data Mapping Logic
- Update Workflow
- Challenges & Solutions
- Live Demo
- Future Improvements

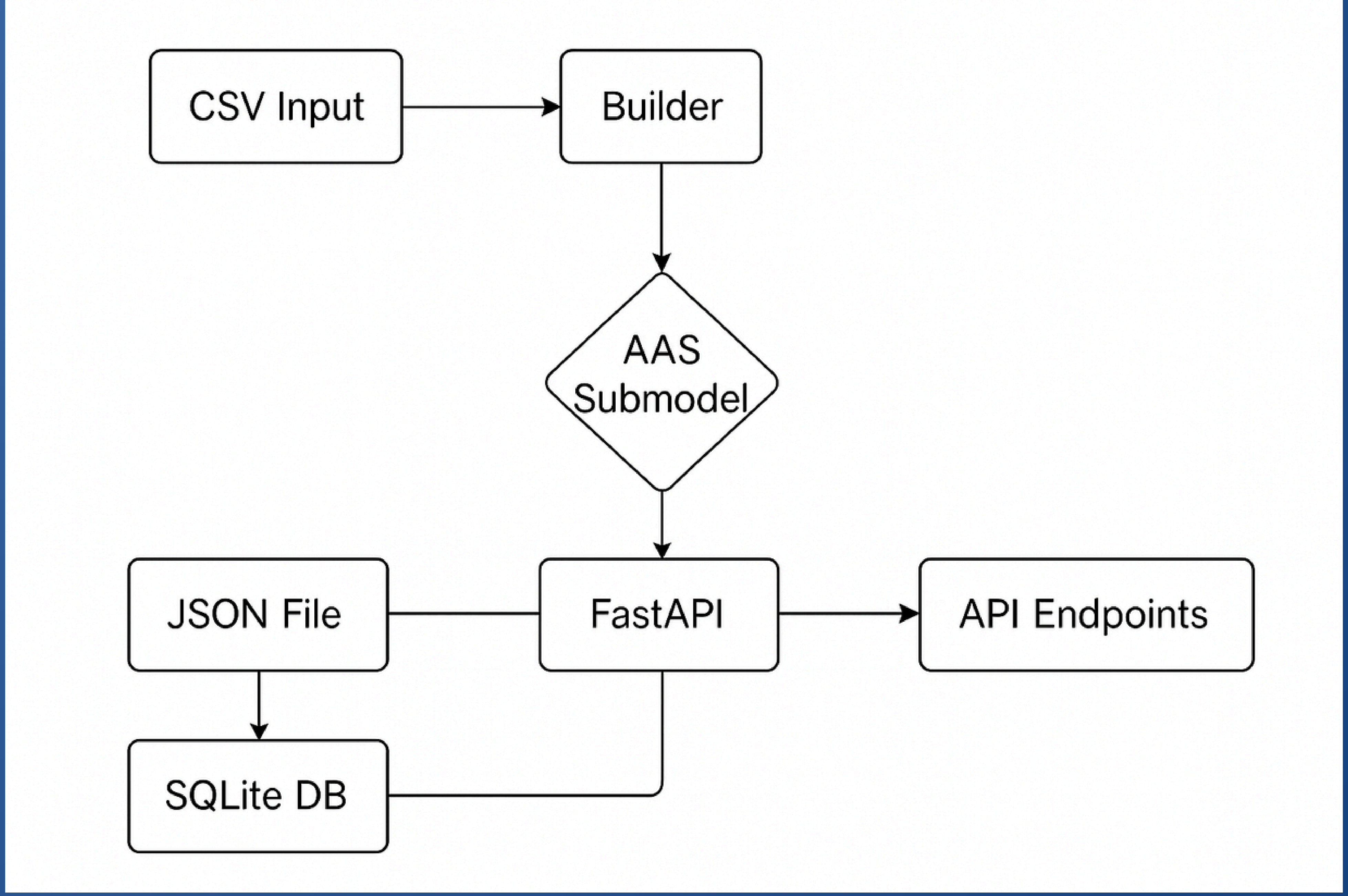
PROJECT OVERVIEW

- Purpose:

  "Convert sensor CSV data into IDTA 02008-1-1 compliant AAS submodel, and provide API"

- Key Features:
  - CSV → AAS conversion
  - REST API exposure
  - External data storage
- Tech Stack: Python, FastAPI, SQLite, aas-core3

ARCHITECTURE DIAGRAM



ARCHITECTURE DIAGRAM

Builder Component

- Functions:

- CSV → AAS Submodel conversion engine
- Calculates statistics (RecordCount, SamplingInterval)
- Enforces 500-record inline limit

- Key Logic:

- 1 InternalSegment per sensor
- Timestamp normalization (UTC)
- Schema validation via aas-core3

ARCHITECTURE DIAGRAM

SQLite Database

- Role:

- External storage for full historical data
- Enables time-range queries (/records endpoint)

ARCHITECTURE DIAGRAM

FastAPI Layer

Endpoints:

- GET /submodels/time-series → Full submodel
- GET /records → Filtered time-series data

Features:

- Auto-generated Swagger docs
- JSON-schema validation

ARCHITECTURE DIAGRAM

AAS Compliance

Standards Met::

- IDTA 02008-1-1 template

Logic Mapping

CSV → AAS Format

CSV Field	AAS Field
component_id	asset.identification.id
sensor_value	submodel.value
timestamp	submodel.timestamp

```
mapping = {
"component_id":
"asset_identifiication.id
"sensor_value":
"submodel.timestamp"
```

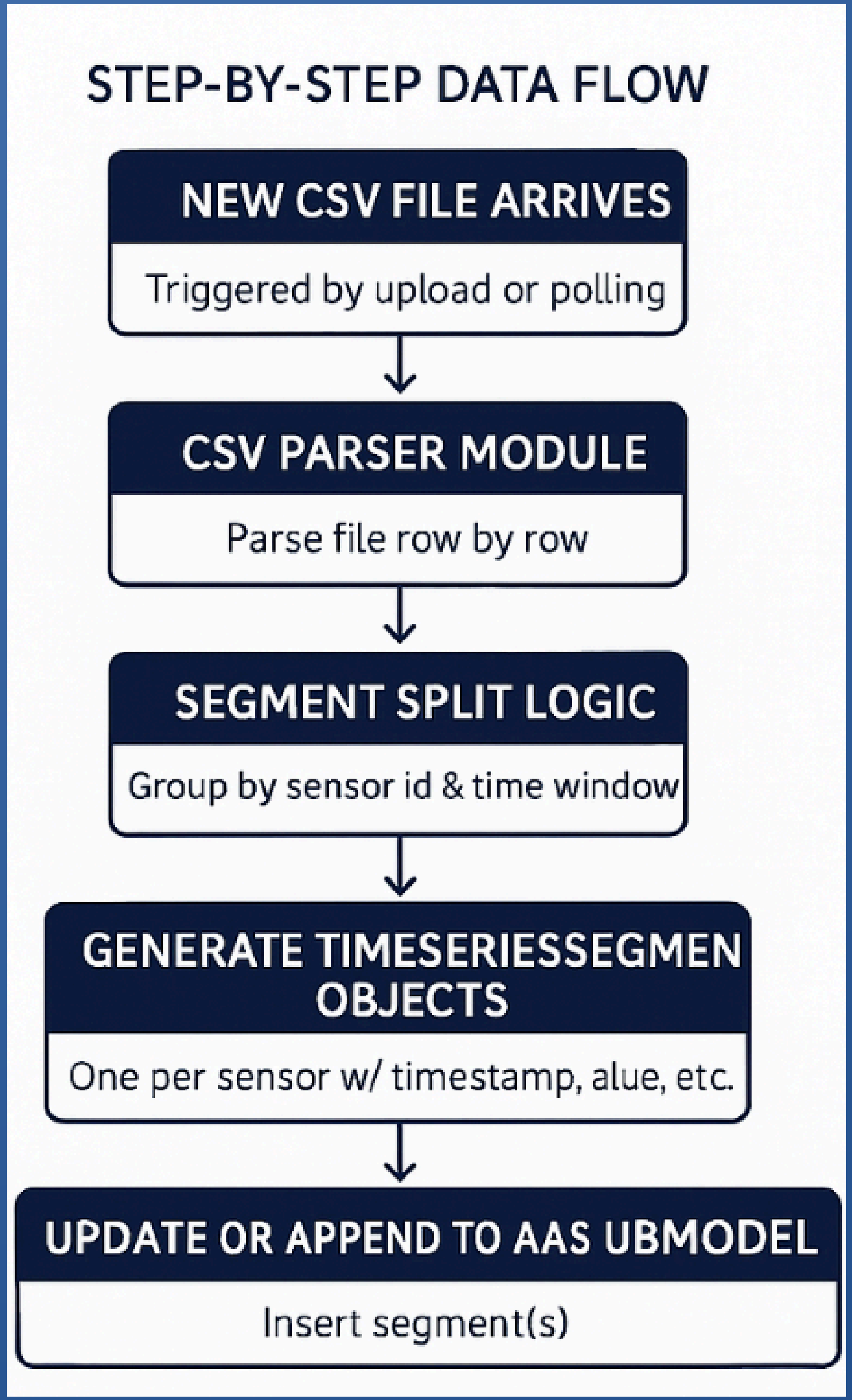
Derived Data:

Statistics Calculation

- Load incoming batch
- Parse values
- Group by component
- Calculate: Mean, Min, Max StdDev
- Add to AAS metadata

Metric	Calculation Example
Mean	avg(sensor_value)
Standar-Dev	std(sensor_value)

UPDATE WORKFLOW



STEP-BY-STEP DATA FLOW

NEW CSV FILE ARRIVES

Triggered by upload or polling

↓

CSV PARSER MODULE

Parse file row by row

↓

SEGMENT SPLIT LOGIC

Group by sensor id & time window

↓

GENERATE TIMESERIESSEGMEN OBJECTS

One per sensor w/ timestamp, alue, etc.

↓

UPDATE OR APPEND TO AAS UBMODEL

Insert segment(s)

Rules For Segment Handeling

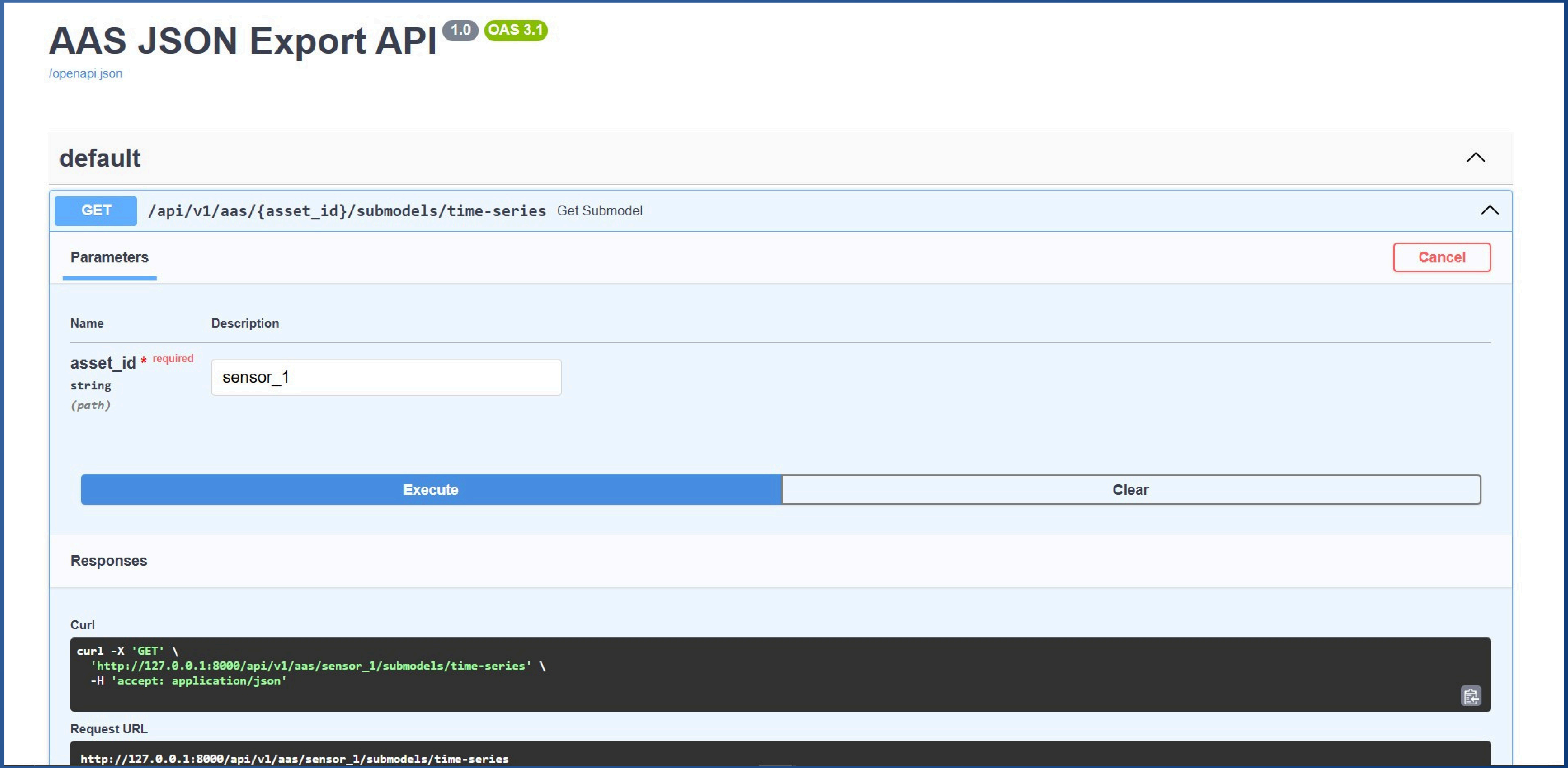
- One TimeSeries Segment per sensor
- split segments by sampling rate “every one min”
- Fields to fill: observed Time ,value , unit,segmentId, description.
- Avoid Overlap
- Update logic segmentId exist append new data else , create new segment

All tested and validated using AASX Package Explorer

CHALLENGES

- Challenge 1: Handling Dynamic Data in Tests
- Challenge 2: Validating Segment Structure Against IDTA 02008-1-1

API DEMO



AAS JSON Export API 1.0 OAS 3.1

/openapi.json

default

GET `/api/v1/aas/{asset_id}/submodels/time-series` Get Submodel

Parameters

Name	Description
asset_id * required string *(path)*	sensor_1

Responses

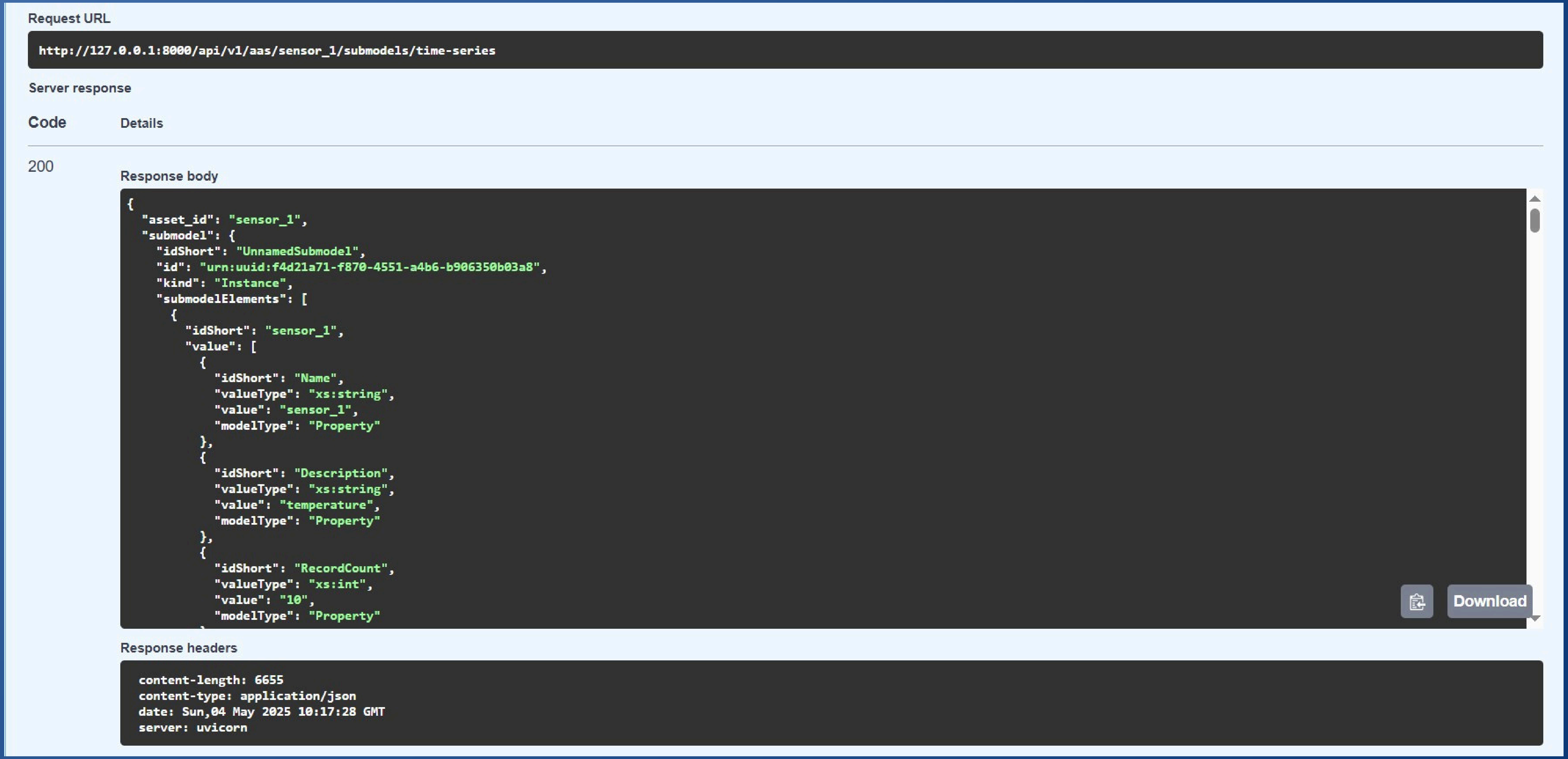
Curl

```
curl -X 'GET' \
  'http://127.0.0.1:8000/api/v1/aas/sensor_1/submodels/time-series' \
  -H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:8000/api/v1/aas/sensor_1/submodels/time-series
```

API DEMO



Request URL

```
http://127.0.0.1:8000/api/v1/aas/sensor_1/submodels/time-series
```

Server response

Code Details

200

Response body

```
{
  "asset_id": "sensor_1",
  "submodel": {
    "idShort": "UnnamedSubmodel",
    "id": "urn:uuid:f4d21a71-f870-4551-a4b6-b906350b03a8",
    "kind": "Instance",
    "submodelElements": [
      {
        "idShort": "sensor_1",
        "value": [
          {
            "idShort": "Name",
            "valueType": "xs:string",
            "value": "sensor_1",
            "modelType": "Property"
          },
          {
            "idShort": "Description",
            "valueType": "xs:string",
            "value": "temperature",
            "modelType": "Property"
          },
          {
            "idShort": "RecordCount",
            "valueType": "xs:int",
            "value": "10",
            "modelType": "Property"
```

Response headers

```
content-length: 6655
content-type: application/json
date: Sun,04 May 2025 10:17:28 GMT
server: uvicorn
```

BAM

THANK YOU

Questions?