Rakshit Somkuwar

  21BCS122

software engineering assgniment 2

Theme : Evolution of digitalisation in the energy sector:

The energy sector is now in a profound transition towards a very important energy

transformation, and digitalisation is one of the key facilitators to ensure that it is fulfilled. In

the recent past, companies started by switching the use of analogue meters to digital meters,

smart meters etc., in order to improve energy efficiency.

Digitalisation acts as a lever in the sector to combat climate change and optimise power

generation processes to reduce emissions and meet the objective of decarbonisation of the

energy model.

Main problems of the renewable energy sector : Impediments faced by companies in the

sector are:

• Geographically dispersed energy data ,

• Lack of integrated platform ,

• Inability to track assets,

• Lack of clear and traceable objectives

Benefits of digital transformation in the renewable energy sector:

Digitalisation, if carried out guided by an integrated operations platform, facilitates the

integration of renewable energies, energy policies and transparency in the management of

these. In addition, it allows to have the user much more connected, offering the following

benefits:

• Digitalisation tools and platforms help build renewable energy plants with automated

processes, for informed decision making. In addition, the interconnections they propose

are the basis for a more decentralised generation, thus avoiding isolated ‘energy

islands’.

• These platforms reduce downtime by offering alerts based on predictive maintenance,

anticipating asset maintenance. The modernisation of production plants is necessary to

make them more competitive and efficient.

• They allow a more accurate forecast of the weather and market conditions, which

helps to maximise renewable production, by offering a deep analysis of all information

received in real time, to be able to make decisions and offer stability in demand.

• The use of artificial intelligence and machine learning to optimise the engineering and

construction of new renewable sources and plants reduces time to market,

anticipating the benefits of free C02 generation and increasing production.

Objective: To develop Digital-based future energies

New power plants are born digital by their design, guaranteeing the efficiency and high

availability of their services. In addition, they are backed by digital twins that help with

modelling, forecasting, and testing for optimal performance, from power generation to its link

with the customers.

But for most existing plants, the basic need is in installing sensors and counters throughout the

system to create Smart Grids. All these new systems must be connected to existing ones in

order to achieve digitalisation in the sector.

Digitalisation : To achieve this, energy companies must rely on management software

capable of interconnecting all assets and centralising their management in order to transition

to renewable energy generation and reduce the carbon footprint in their operations

Target audiences :

• Private and Public Organisations, Homes, etc

Assignment scope :

1. List various requirements(scope) for the above program initiative that can be used for

developing a suitable technology oriented digital solution.

2. Identify various technologies, tools and systems available in the market to support these

needs.

3. Generate one API and suitable data analysis Code base to access the energy related data

set and perform data analysis

Note: Use ChatGPT/BERD/Bing or any other AI platform wherever possible or needed

Deliverables :

1. List of requirements

2. List of tools, technologies and systems to support such needs.

3. Working API code

The solution of the given problem is:-

1. Requirements (scope) for the digital solution in the energy sector could include:
2. Sure, here are some requirements or scope for developing a technology-oriented digital solution related to the evolution of digitalization in the energy sector:
3. Data management: The solution should be able to manage and analyze large amounts of data related to energy generation, consumption, and distribution, including data from smart grids, IoT devices, and other sensors.
4. Real-time monitoring and control: The solution should provide real-time monitoring and control of energy systems to optimize energy efficiency, reduce waste, and minimize downtime.
5. Energy storage and management: The solution should be able to manage and optimize the storage and distribution of energy from various sources, including renewables like solar and wind.
6. Predictive maintenance: The solution should be able to predict potential failures and maintenance needs in energy systems based on data analytics, reducing downtime and maintenance costs.
7. Cybersecurity: The solution should have robust security features to protect against cyber threats and ensure data privacy and confidentiality.
8. Integration with existing systems: The solution should be able to integrate with existing energy systems, including legacy systems, to ensure seamless operations and compatibility with existing infrastructure.
9. User interface and experience: The solution should have a user-friendly interface that is easy to use and navigate for energy operators, technicians, and other stakeholders.
10. Here is an example API and suitable data analysis code base to access an energy-related data set using Python flask framework:
11. Data Management: Apache Hadoop, Apache Spark, Cassandra, MongoDB, Oracle Database, Microsoft SQL Server, PostgreSQL, and MySQL.
12. Real-time Monitoring and Control: SCADA systems, Distributed Control Systems (DCS), Programmable Logic Controllers (PLC), IoT platforms, and edge computing.
13. Energy Storage and Management: Lithium-ion batteries, Flow batteries, Pumped Hydro Storage (PHS), Compressed Air Energy Storage (CAES), Thermal Energy Storage (TES), and Flywheels.
14. Predictive Maintenance: Predictive maintenance software such as IBM Maximo, SAP Predictive Maintenance and Service, GE Predix, and Microsoft Azure Machine Learning.
15. Cybersecurity: Cybersecurity solutions such as firewalls, intrusion detection systems, and antivirus software. Additionally, blockchain technology can provide a secure and transparent way to store and manage energy data.
16. Integration with Existing Systems: Application Programming Interfaces (APIs), Service-Oriented Architecture (SOA), and Enterprise Service Bus (ESB) are used for integrating different energy systems.
17. User Interface and Experience: Human-machine interfaces (HMI), dashboard software such as Tableau and Power BI, and augmented and virtual reality technologies.
18. Sustainability: Renewable energy technologies such as solar panels, wind turbines, and hydropower generators, and energy-efficient devices such as smart thermostats, LED lighting, and high-efficiency HVAC systems.
19. Scalability: Cloud computing platforms such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform.
20. Regulatory Compliance: Compliance management systems such as RSA Archer, MetricStream, and SAP GRC

from flask import Flask, jsonify, request

from flask\_sqlalchemy import SQLAlchemy from flask\_marshmallow import Marshmallow

# Initialize Flask application

app = Flask(\_\_name\_\_)

# Set up database connection

app.config['SQLALCHEMY\_DATABASE\_URI'] = 'sqlite:///energy\_data.db'

app.config['SQLALCHEMY\_TRACK\_MODIFICATIONS'] = False

db = SQLAlchemy(app)

ma = Marshmallow(app)

# Define Energy Data model

class EnergyData(db.Model):

id = db.Column(db.Integer, primary\_key=True)

timestamp = db.Column(db.DateTime, nullable=False)

energy = db.Column(db.Float, nullable=False)

def \_\_init\_\_(self, timestamp, energy):

self.timestamp = timestamp

self.energy = energy

# Define Energy Data schema

class EnergyDataSchema(ma.Schema):

class Meta:

fields = ('id', 'timestamp', 'energy')

energy\_data\_schema = EnergyDataSchema()

energy\_datas\_schema = EnergyDataSchema(many=True)

# Define API endpoints

@app.route('/energy-data', methods=['GET'])

def get\_energy\_data():

all\_energy\_data = EnergyData.query.all()

result = energy\_datas\_schema.dump(all\_energy\_data)

return jsonify(result)

@app.route('/energy-data', methods=['POST'])

def add\_energy\_data():

timestamp = request.json['timestamp']

energy = request.json['energy']

new\_energy\_data = EnergyData(timestamp, energy)

db.session.add(new\_energy\_data)

db.session.commit()

return energy\_data\_schema.jsonify(new\_energy\_data)

@app.route('/energy-data/<id>', methods=['PUT'])

def update\_energy\_data(id):

energy\_data = EnergyData.query.get(id)

timestamp = request.json['timestamp']

energy = request.json['energy']

energy\_data.timestamp = timestamp

energy\_data.energy = energy

db.session.commit()

return energy\_data\_schema.jsonify(energy\_data)

@app.route('/energy-data/<id>', methods=['DELETE'])

def delete\_energy\_data(id):

energy\_data = EnergyData.query.get(id)

db.session.delete(energy\_data)

db.session.commit()

return energy\_data\_schema.jsonify(energy\_data)

# Run the Flask application

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)