Software Requirements Specification

for

EU - Energy Consumption

and EU ECOSmart APP

Version 1.2 (approved)

Prepared by BDL-HS23 Group 2

HSLU - Business

Lucerne, 29.02.24

Revision History ii
1. Introduction 1
1.1 Purpose 1
1.2 Document Conventions 2
1.3 Intended Audience and Reading Suggestions 2
1.4 Product Scope 3
1.5 References 3
2. Overall Description 4
2.1 Product Perspective 4
2.2 Product Functions 4
2.3 User Classes and Characteristics 5
2.4 Operating Environment 5
2.5 Design and Implementation Constraints 5
3. External Interface Requirements 6
3.1 User Interfaces 6
3.2 Hardware Interfaces 7
3.3 Software Interfaces 8
3.4 Communications Interfaces 9
4. System Features 10
4.1 System Feature 1- Uploader Interface 11
4.2 System Feature 2 -Visualization Interface 11
5. Other Nonfunctional Requirements 12
5.1 Performance Requirements 12
5.2 Safety Requirements 13
5.3 Security Requirements 13
5.4 Software Quality Attributes 14
5.5 Business Rules 14

Revision History

Name	Date	Reason For Changes	Version
Tejesh Reddy Koki	26.02.2024	Additional functional information.	1.1
Maha Hazimé-Zayour	29.02.2024	Additional functional information.	1.2

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to outline the functional and non-functional requirements for the EU ECOSmart application. This document provides a comprehensive understanding of the features, functionalities, and constraints of the application. It covers the scope of the EU ECOSmart prototype application, including its capabilities related to visualizing energy consumption data for western European countries, while analyzing EU trends and correlations.
The first prototype of EU ECOSmart application aims to:

- Provide users with insights into energy consumption patterns across Western European countries, focusing on high-consumption periods such as winter heating, summer cooling, and industrial usage.
- Enable users to visualize energy consumption data through interactive dashboards created in Tableau Desktop. For instance, analyzing correlations between energy consumption in different (western europe) countries over the period from 2015 to 2020.
- Presenting the functionality of the application, including the Blob file uploader for uploading energy data files, monitoring in Data Factory, manually running the application through access to Azure executable apps,
- Present functionality of the application (Blob file uploader: Energy Data File uploader APP (message that new csv files have been uploaded in the container) , monitor in data factory manually run application (acces Azure exe app), trigger the processing of the added data into SQL further data joining cleaning and processin in Visual studio (IDE) (instead of data bricks) merging, cleaning, loading in Tableau.
- cost estimates associated with Microsoft Azure services utilized for data storage, processing, and visualization would not be made available for the users
- A monthly fee for the users would be implied.

This SRS document pertains specifically to the software requirements of the EU ECOSmart application, encompassing its data visualization, analysis, and cost estimation functionalities. It does not cover the implementation details or technical specifications of underlying systems or infrastructure.
Furthermore, this document represents a single subsystem within a larger ecosystem of energy management tools and platforms. It does not encompass the entirety of energy management systems but focuses solely on the functionalities and capabilities of the EU ECOSmart application.
The revision or release number of this document is Version 1.0.

1.2 Document Conventions

This SRS document pertains specifically to the software requirements of the EU ECOSmart application, encompassing its data visualization, analysis, and cost estimation functionalities.
Furthermore, this document represents a single subsystem within a larger ecosystem of energy management tools and platforms. This first version does not encompass the entirety of energy management systems but focuses solely on the primary functionalities and capabilities of the EU ECOSmart application prototype.

This SRS follows standard documentation conventions, including clear and concise language, consistent formatting, and organized structure. Key conventions include:

- Use of headings and subheadings to delineate different sections and subsections.
- Consistent use of bullet points for listing requirements and specifications.
- Identification of priorities for requirements, where applicable, using a standardized notation (e.g., High, Medium, Low).
- Adherence to industry best practices for documenting software requirements, ensuring clarity and comprehensibility for stakeholders.
- Any specific conventions or standards relevant to the project are noted within the document as needed.

1.3 Intended Audience and Reading Suggestions

This document is intended for various stakeholders involved in the development, deployment, and utilization of the EU ECOSmart application. The audience might includes:

1. **Developers:** Those responsible for implementing the software, including backend and frontend developers, database administrators, and DevOps engineers.
2. **Project Managers:** Individuals overseeing the planning, execution, and monitoring of the energy consumption project, ensuring alignment with business goals and client requirements.
3. **Energy Efficiency Institutions:** Organizations focused on promoting energy efficiency and sustainability, leveraging the insights provided by the application to inform policy decisions and initiatives.
4. **Government Agencies:** Entities responsible for regulatory oversight and policy-making related to energy consumption and environmental sustainability.
5. **Business Analysts:** Professionals tasked with analyzing market trends, identifying user needs, and translating requirements into actionable insights for development teams.
6. **Testing Teams:** Individuals responsible for validating the functionality, performance, and security of the application before deployment.

The document provides a comprehensive overview of the first version for the EU ECOSmart application, including its purpose, scope, requirements, and technical specifications. It is organized into sections that cater to different aspects of the project, allowing readers to navigate based on their specific interests and responsibilities.

1.4 Product Scope

The EU ECOSmart application is a subscription-based software solution designed to provide actionable insights into energy consumption patterns across European Union countries. It serves as a vital tool for industries, state-based institutions, educational entities, and research organizations, offering comprehensive energy data analytics and forecasting capabilities.

EU ECOSmart Purpose and Benefits:

- The application's goal is to empower users by utilizing their valuable, organized data to provide real-time visualization of energy consumption. This functionality facilitates informed decision-making and strategic planning.
- By analyzing past energy trends and correlating data across different regions, the app facilitates accurate forecasting, for instance, aiding industries in optimizing their energy usage and reducing costs.
- It aligns with corporate goals and business strategies by promoting sustainability, efficiency, and environmental responsibility.
- The app serves as a grid solution, offering intuitive visualization and analysis tools for monitoring and managing energy consumption at both macro and micro levels.
- The app also offers the capability to upload multiple coherent and uniform big data CSV files for analysis and subsequent visualization in Tableau.

Objectives and Goals:

- Provide a user-friendly web-based interface for uploading csv files accessing and interpreting energy consumption data.
- Enhance forecasting accuracy through advanced data analytics and machine learning algorithms.
- Support industry stakeholders, state-based institutions, educational entities, and research organizations in their efforts to promote energy efficiency and sustainability.
- Foster collaboration and knowledge sharing among users, facilitating collective efforts toward achieving energy conservation goals.

For a more detailed understanding of the product scope, please refer to the vision and scope document.

1.5 References

1. Energy Efficiency Directive
    - Title: Energy Efficiency Directive
    - Author: European Commission
    - Version Number: Updated in 2018 and 2023
    - Date: [Date of the latest update]
    - Source: [Link to the directive document or official European Commission website]

2. Eurostat Energy Statistics Overview
    - Title: Energy statistics - An overview
    - Author: Eurostat
    - Version Number: Not applicable
    - Date: [Date of access]
    - Source: [Link to the Eurostat webpage]

These references provide additional context and background information related to energy efficiency regulations and statistics in the European Union.

2. Overall Description

2.1 Product Perspective

The product specified in this SRS is a new, self-contained application designed to address the need for comprehensive energy consumption analysis across Western European countries. It is not a replacement for existing systems but rather a standalone solution built on Azure cloud services. This application serves as a component within a larger energy management ecosystem, facilitating data visualization, analysis, and reporting. Major components include data collection, storage, processing, visualization, and user interface modules. External interfaces include data sources, user input, and integration with Azure services.

2.2 Product Functions

The app must perform the following major functions:
- Enable users to upload energy consumption data files.
- Conduct data verification, cleaning, and processing.
- Store processed data securely in a SQL database.
- Provide intuitive visualization of energy consumption trends.
- Allow users to analyze correlations and insights.
- Generate real-time notifications and comprehensive reports.
- Ensure scalability, reliability, and high performance.
- Implement robust security measures for data protection.
- Monitor system health and log activities for auditing purposes.

A top-level data flow diagram illustrating the flow of data through the system would be effective in visualizing these functions.

2.3 User Classes and Characteristics

The anticipated user classes for this product include energy analysts, administrators, stakeholders, and data entry personnel. Energy analysts will frequently use the product for data analysis, administrators will manage system configurations, stakeholders will rely on the product for decision-making insights, and data entry personnel will upload energy consumption data. Each user class has distinct needs and priorities, ranging from technical expertise to simplicity *in data uploading processes.*

2.4 Operating Environment

The software will operate within the Azure cloud environment, utilizing Azure services such as Blob storage, Data Factory, SQL Database, and Tableau Desktop. It could run on virtual machines (VMs) provisioned by Azure, with the operating system being Windows Server or Linux, depending on the specific requirements of the Azure services used. The software must peacefully coexist with other Azure services and applications within the same Azure subscription, ensuring compatibility and efficient resource utilization. Additionally, it may interact with external systems or APIs for data integration purposes, requiring compatibility with those systems as well.

2.5 Design and Implementation Constraints

The following are some of the implementations constrained of the Application EU ECOSmart

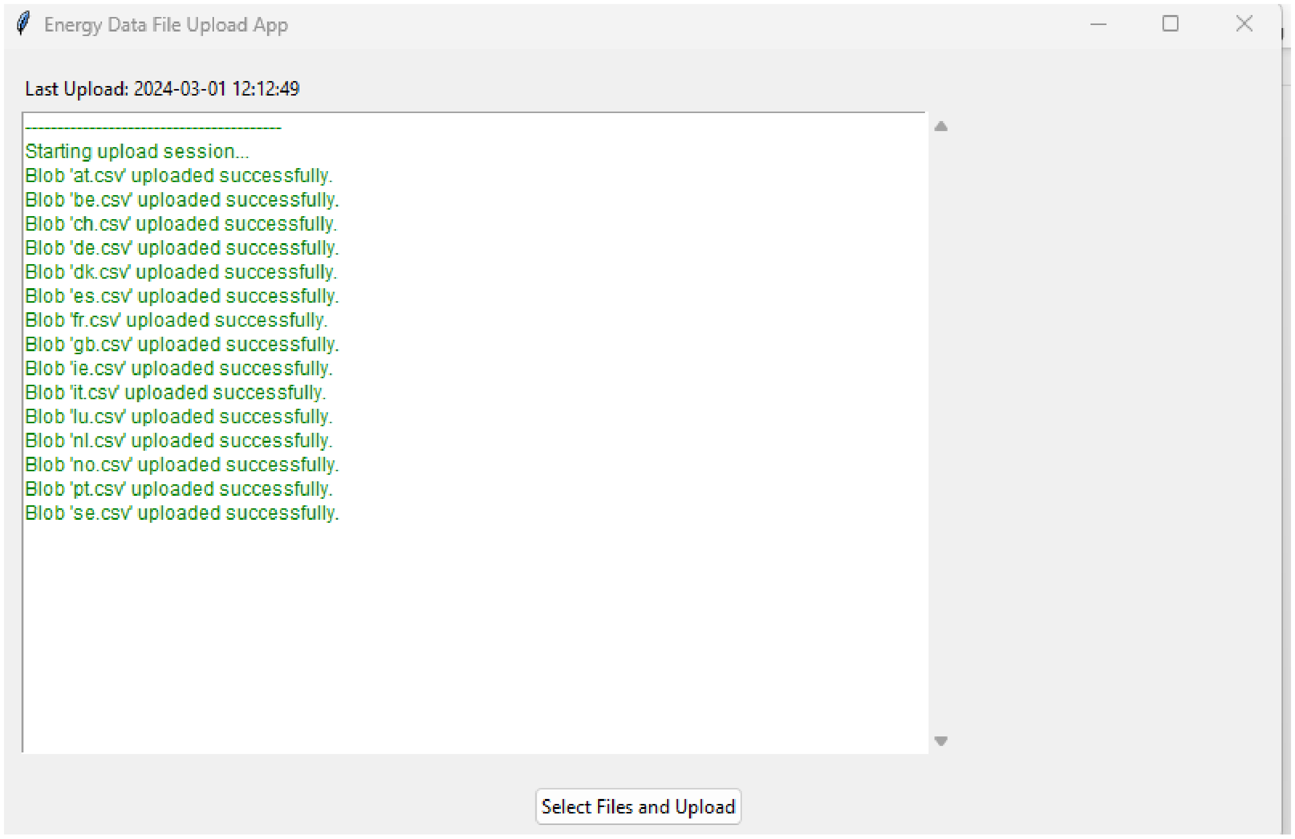
- Corporate or regulatory policies may restrict certain technology choices and implementation approaches.
- Hardware limitations, including timing requirements and memory constraints, will influence design decisions.
- Compatibility and integration requirements with other applications or systems will guide technology stack and communication protocols.
- Mandated or preferred technologies, such as Azure services and Tableau Desktop, will constrain development choices.
- Stringent security requirements will dictate encryption, authentication, and authorization mechanisms.
- Adherence to design conventions, programming standards, and maintenance practices will ensure compatibility with organizational standards.

3. External Interface Requirements

3.1 User Interfaces

In terms of users, we have two user interfaces, one is an .exe application supported in Windows, and the other would be a Tableau interface where the user could interact with data and get an overview and insight of it.

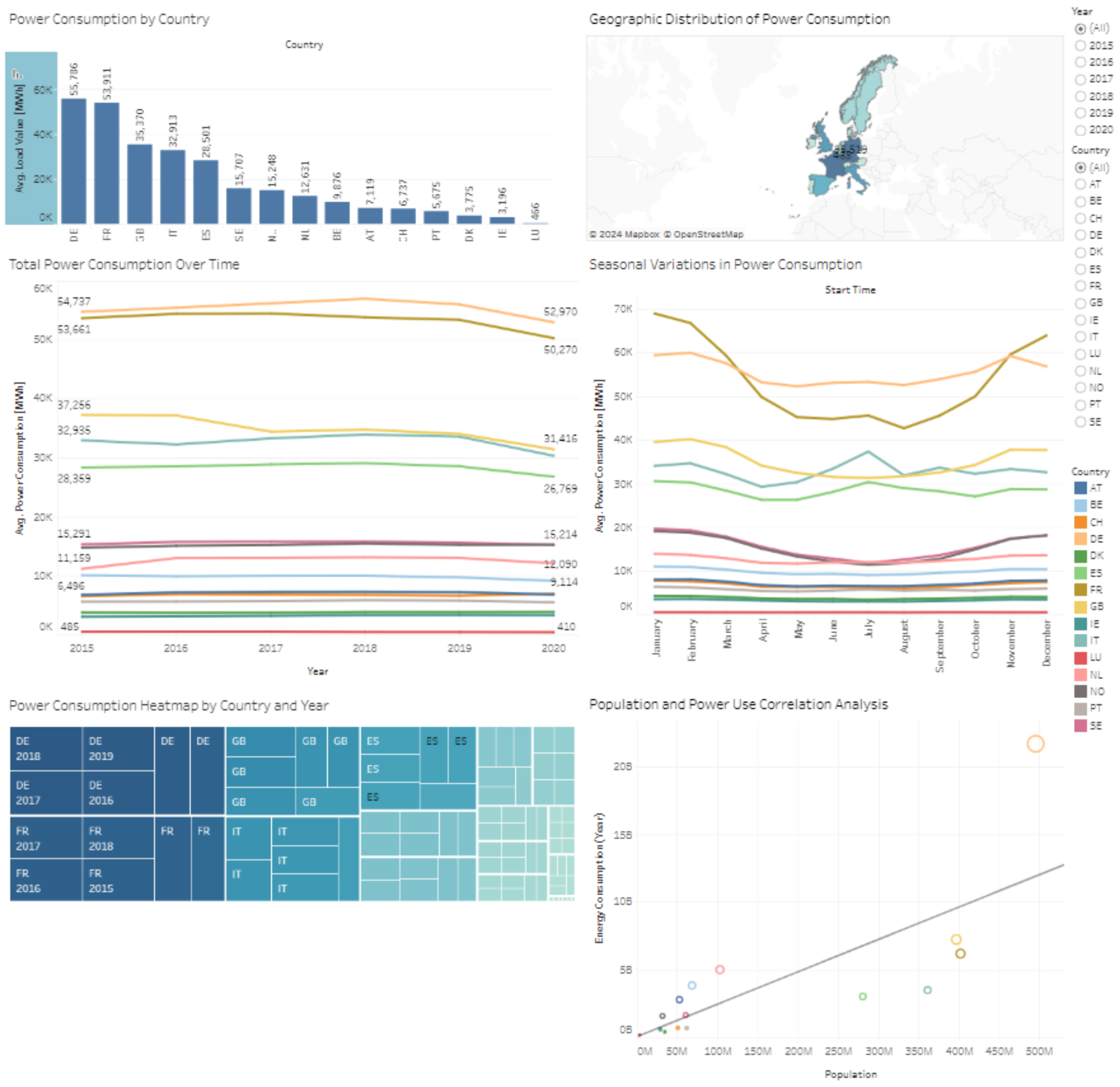
Below we can see windows-based application to upload files to the Azure Blob Storage. There is only one button available [Select Files and Upload], by clicking it takes the user to defined path in the local machine and the user should upload files in that path.



Last Upload: 2024-03-01 12:12:49

```
------------------------------------------
Starting upload session...
Blob 'at.csv' uploaded successfully.
Blob 'be.csv' uploaded successfully.
Blob 'ch.csv' uploaded successfully.
Blob 'de.csv' uploaded successfully.
Blob 'dk.csv' uploaded successfully.
Blob 'es.csv' uploaded successfully.
Blob 'fr.csv' uploaded successfully.
Blob 'gb.csv' uploaded successfully.
Blob 'ie.csv' uploaded successfully.
Blob 'it.csv' uploaded successfully.
Blob 'lu.csv' uploaded successfully.
Blob 'nl.csv' uploaded successfully.
Blob 'no.csv' uploaded successfully.
Blob 'pt.csv' uploaded successfully.
Blob 'se.csv' uploaded successfully.
```

Here we can see the other interface that users could use to analyze the data and get insights.



3.2 Hardware Interfaces

Visualization Interfaces

1. **Supported Devices:** The visualization component of the software is designed to be compatible with a wide range of devices including desktop computers, laptops, tablets, and smartphones. This ensures accessibility and usability across different user preferences and scenarios.

2. **Data and Control Interactions:** Visualization interfaces are designed to fetch and display energy consumption data from the software's database. Users interact with the software through devices' input methods (keyboard, mouse, touch) to navigate, filter, and manipulate the visualized data.

3. **Communication Protocols:** The visualization interfaces use HTTP/HTTPS for secure web communication, facilitating the retrieval of data from the server and the submission of user commands. The responsive design adapts to different screen sizes and resolutions, ensuring a seamless user experience across devices.

Data File Uploader Interface

1. **Supported Platforms:** Currently, the data file uploader interface is supported on Windows-based systems and Virtual Machines (VMs). This facilitates the secure and efficient uploading of energy consumption data files into the system for processing and analysis.

2. **Data Interactions:** The primary function involves users selecting and uploading energy consumption data files. The system processes these files, validating format and size before transferring them to the server for storage and subsequent analysis.

3. **User Input:** Input is received via traditional input devices such as keyboards and mice, enabling users to navigate file directories, select files for upload, and initiate the upload process.

3.3 Software Interfaces

1. **Azure Blob Storage:** Used for storing data files uploaded from local storage. Interface facilitates the secure transfer of CSV files to the cloud environment.
2. **Azure Data Factory:** Triggers data pipelines upon blob creation in Azure Blob Storage. Manages the data flow from Azure Blob Storage to SQL databases.
3. **SQL Database (Azure SQL Database/SQL Server):** Receives and stores data from Azure Blob Storage for processing and analysis.
4. **Visual Studio Code:** Provides an integrated development environment (IDE) for writing scripts to clean and manipulate data in the SQL database. Extensions for SQL server management and Azure resources are utilized. *Version*: 1.87.0
5. **Tableau:** Connects to the SQL Database for data visualization and analysis. *Version*: Specify the version used. Version:2023.3

Data Items and Messages:

1. **Incoming Data Files:** CSV files uploaded by users to the local folder, transferred to Azure Blob Storage. *Purpose*: For data analysis and visualization.
2. **Data Pipeline Trigger:** A message/event triggered when a new blob is created in Azure Blob Storage, activating the data pipeline in Azure Data Factory.
3. **Processed Data:** Cleaned and structured data in SQL Database ready for analysis. Purpose: To be used for generating insights and visualizations in Tableau.

Services and Communications:

1. Secure file transfer services between local storage and Azure Blob Storage.
2. Event-driven data pipeline activation in Azure Data Factory.
3. Data querying and manipulation services via Visual Studio Code to SQL Database.
4. Data visualization services from SQL Database to Tableau.

Data Sharing and Implementation Constraints:

1. **Data Sharing:** Data shared across Azure Blob Storage, Azure Data Factory, and SQL Database must adhere to secure transfer protocols (e.g., HTTPS, SSL/TLS) to ensure data integrity and confidentiality.
2. **Implementation Constraints:** Global data areas in a multitasking operating system must be used for managing concurrent data processing tasks, ensuring thread safety and data consistency.

3.4 Communications Interfaces

Internet and Network Connectivity:

1. **Protocol Requirements:** The application requires HTTP/HTTPS protocols for secure web communication between the client-side interface and the server. This is essential for data transfer to and from Azure Blob Storage and for accessing the SQL database.
2. **Bandwidth Requirements:** Adequate bandwidth is necessary to support the uploading of data files to Azure Blob Storage and the subsequent data transfers to the SQL database. The application should support bandwidths sufficient to handle expected data loads without significant latency or downtime.

Azure Blob Storage Communication:

Azure Storage REST API: Communication with Azure Blob Storage is facilitated through the Azure Storage REST API, utilizing HTTPS for secure data transfers. This API is used for uploading blobs in cloud storage.

Azure Data Factory Communication:

Azure Management REST API: Azure Data Factory uses the Azure Management REST API for creating, scheduling, and managing data pipelines. This includes triggering pipelines based on events, such as the creation of a new blob in Azure Blob Storage.

SQL Database Communication:

SQL Server Connection: The application connects to the SQL database using the TCP/IP protocol, typically over port 1433 for SQL Server. This connection is used for executing SQL queries, updating, and retrieving data.

Visual Studio Code Integration:

Extensions and Debugging Protocol: Visual Studio Code communicates with SQL databases and Azure services through specific extensions that utilize Microsoft's Language Server Protocol (LSP) for code editing support and the Debug Adapter Protocol (DAP) for debugging.

Tableau Data Visualization:

ODBC/JDBC Drivers: Tableau connects to the SQL database using ODBC or JDBC drivers, depending on the specific SQL database being used. This connection allows for the direct querying and visualization of data stored in the database.

Security Considerations:

Encryption: All communications between the application components and external systems must be encrypted using TLS 1.2 or higher to ensure data privacy and integrity.
Authentication: Secure authentication mechanisms, such as OAuth 2.0, should be used for accessing Azure services and the SQL database, ensuring that only authorized users and services can access sensitive data.

4. System Features

The functional requirements of the product are organized around key system features. Firstly, the Data Upload feature enables users to upload energy consumption data files, validating their format and size before processing and securely storing the data. Notifications are provided upon successful upload, while error handling ensures appropriate feedback for any invalid inputs. Secondly, the Visualization Interface offers users access to interactive data visualizations, ensuring prompt loading and rendering of visual elements and responsive interaction. Finally, the Data Analysis feature empowers users to analyze energy consumption trends and correlations, leveraging statistical tools for interpretation and facilitating sharing of insights with stakeholders.

4.1 System Feature 1- Uploader Interface

This feature facilitates the uploading of energy consumption data files by users. It is considered a high-priority component of the system due to its foundational role in data collection and subsequent analysis processes.

Stimulus/Response Sequences:
User Action: User selects the "Upload Data" option on the application interface.
System Response: The system prompts the user to select the data file they wish to upload.
User Action: The user selects the desired file from their local system.
System Processing: The system validates the file format and size, processes the file for storage, and logs the upload activity.
System Notification: The user is notified of a successful upload or informed of any errors related to file format or size.

Functional Requirements:

1. **Provide Option to Upload Data Files:** The system must offer users a clear and accessible option to upload energy consumption data files from their local system.
2. **Validate File Format and Size:** The system must validate the format of the uploaded file to ensure it matches the supported types (e.g., CSV, Excel) and does not exceed the maximum allowed size.
3. **Process Uploaded Data for Storage:** Upon successful validation, the system must process the uploaded data files for storage in a designated data repository, ensuring data integrity and availability for future processing.
4. **Notify User Upon Successful Data Upload:** The system must provide a confirmation message to the user once their data file has been successfully uploaded and processed, confirming the action's success.
5. **Display Error Messages for Invalid File Formats or Sizes:** If a file does not meet the specified format or size requirements, the system must inform the user of the specific issue, guiding them to correct the error and try again.

4.2 System Feature 2 -Visualization Interface

This feature enables users to create, customize, and interact with data visualizations using Tableau, a leading tool for data analytics and visualization. The interface provides seamless access to data stored in the SQL database, allowing for real-time data analysis and reporting.

Functional Requirements:

1. **Data Connection and Syncing:** The application must establish a live data connection between the SQL database and Tableau, ensuring that data visualizations are updated in real-time as the underlying data changes.
2. **Visualization Customization:**Users must be able to create and customize data visualizations in Tableau using the graphical user interface (GUI) without needing to write SQL queries manually.
3. **Dashboard Creation and Sharing:**The interface should allow users to compile visualizations into dashboards and share these dashboards with other users or stakeholders.

5. Other Nonfunctional Requirements

In this chapter, we will some nonfunctional requirements.

5.1 Performance Requirements

1. Data Transfer Speeds: Specify the minimum and maximum time allowed for data files to be transferred from local storage to Azure Blob Storage and from there into the SQL database. For example, "Data files less than 100MB must be transferred to Azure Blob Storage within 2 minutes of file creation."
2. Data Processing Latency: Define the acceptable latency for data processing steps, including time to trigger data pipelines upon blob creation and the time taken to execute SQL queries for data cleaning. For instance, "Data pipelines must be triggered within 60 seconds of new blob creation, and SQL queries for data cleaning must execute within 5 minutes for datasets up to 500MB."
3. Concurrency and Load Handling: Establish benchmarks for the number of concurrent data transfers and processing operations the system can handle without significant performance degradation. Example: "The system must support up to 50 concurrent data file uploads and 20 concurrent data processing operations without exceeding a 10% increase in response time."
4. Availability: Set an availability target for the application, such as "The application should be available 99.9% of the time, excluding scheduled maintenance windows."
5. Response Times for User Interactions: Define maximum response times for user interactions within the application, including data visualization rendering in Tableau. For example, "All user-initiated operations within the application should have a response time of less than 3 seconds."
6. Scalability: Outline requirements for scaling operations, such as "The system must automatically scale to accommodate up to a 100% increase in data load with no more than a 5% decrease in performance."
7. Backup and Recovery Times: Specify targets for backup operations and recovery times in case of system failure, e.g., "Full system backups must be completed within 1 hour, and system recovery must be achievable within 4 hours of identifying a failure."

8. Monitoring and Alerts: Require real-time monitoring of performance metrics with thresholds set for triggering alerts. For example, "Performance metrics such as data transfer speed, processing latency, and system load must be monitored in real-time, with alerts triggered if metrics deviate by more than 10% from established thresholds."

5.2 Safety Requirements

1. Data Integrity and Backup: Ensure mechanisms are in place to prevent data loss during transfer from local folders to Azure storage and from there to SQL tables. Implement regular backup schedules and data integrity checks.
2. Error Handling: Develop robust error handling procedures to manage exceptions during data transfer and processing, minimizing the risk of data corruption or loss.
3. Regulatory Compliance: Adhere to relevant data protection and privacy regulations applicable to the storage and processing of data, such as GDPR for users in the European Union. This includes implementing necessary safeguards for personal data.
4. Safety Certifications: Obtain and maintain certifications as required by industry standards for data safety and integrity, such as ISO/IEC 27001 for information security management.

5.3 Security Requirements

1. Authentication and Authorization: Implement secure user authentication mechanisms to access the application. Define roles and permissions to ensure that only authorized personnel can perform specific operations on the data. For example, only the analysts and IT team head from Energy Consumption have some specific admin rights to do the changes on pipelines and alert systems.
2. Data Encryption: Encrypt sensitive data at rest in Azure storage and in transit between the application components to protect against unauthorized access and data breaches. For example, the data we store right now is publicly available as assumed in case if we are storing any other data based on requirements, we make sure the data is encrypted.
3. Compliance and Standards: Ensure the application complies with relevant security standards and regulations, such as the Payment Card Industry Data Security Standard (PCI DSS) in processing payment information like paying for Azure services.
4. Security Certifications: Aim to satisfy security certifications pertinent to the application industry, such as SOC 2 for service organizations, to demonstrate adherence to high security standards.

5.4 Software Quality Attributes

1. Reliability: Ensure the application can perform its required functions under stated conditions for specified periods without failure. Aim for an uptime of 99.9%.
2. Usability: Design the application interface to be intuitive and user-friendly, prioritizing ease of use over ease of learning. Provide comprehensive documentation and support materials.
3. Performance: Ensure the data transfer and processing operations meet predefined performance benchmarks, such as processing times for data loads and queries.
4. Maintainability: Structure the codebase and documentation to facilitate easy updates, bug fixes, and feature additions. Adhere to coding standards and practices that support code readability and reusability.
5. Security: Implement comprehensive security measures, as outlined in section 5.4, to protect against unauthorized access, data breaches, and other cybersecurity threats

5.5 Business Rules

1. Access Control: Define which roles (e.g., administrators, data analysts) can perform specific operations within the application, such as initiating data transfers, cleaning data, or executing visualizations.
2. Data Handling: Specify rules for data handling, including data quality checks before loading into the SQL table, approved methods for data cleaning, and conditions under which data can be shared or exported.
3. Operational Procedures: Establish standard operating procedures for regular maintenance activities, data backups, and handling of data incidents.
4. Change Management: Implement a formal process for managing changes to the application, including feature updates, security patches, and modifications to data processing workflows. This process should involve review and approval by designated authority figures to ensure continuity and integrity of operations.