Scenario: A global retail chain with stores in multiple countries wants to analyze its sales data to make informed decisions about inventory management and marketing strategies. The company has diverse data sources, including point-of-sale (POS) systems in each store, online sales platforms, and inventory databases.

Transaction ID	Timestamp	Product ID	Quantity	Store Location
1	2023-05-10 09:15:00	12345	2	Store A
2	2023-05-10 10:30:00	56789	3	Store B
3	2023-05-10 12:45:00	23456	1	Store A
4	2023-05-10 14:20:00	12345	5	Store C
5	2023-05-10 16:10:00	78901	2	Store B

Clean Data in Staging (Retail_Data_Lake - Clean Layer)

Transaction ID	Timestamp	Product Name	Quantity	Store Location
1	2023-05-10 09:15:00	Widget A	2	Store A
2	2023-05-10 10:30:00	Gadget X	3	Store B
3	2023-05-10 12:45:00	Gizmo B	1	Store A
4	2023-05-10 14:20:00	Widget A	5	Store C
5	2023-05-10 16:10:00	Doodad Y	2	Store B

Staging Process:

1. **Data Extraction**: The retail chain's IT team extracts data from various sources. This includes daily sales transactions from each store's POS system, online sales records from the company's website, and inventory data from the central warehouse. Data is extracted in different formats, including CSV files, databases, and APIs.
2. **Staging Area**: The extracted data is loaded into a dedicated staging area, which is a cloud-based data lake. The data lake can store structured and semi-structured data efficiently. For this example, let's refer to it as "Retail_Data_Lake."
3. **Raw Data in Staging**: The raw data, which contains sales details such as transaction timestamps, product IDs, quantities, and store locations, is placed in the "Raw" layer within the Retail_Data_Lake.
4. **Data Transformation:** Data engineers in the IT team perform data transformations and cleaning tasks within the data lake. These tasks include:
5. **Data Normalization**: Converting currencies to a common currency for accurate analysis.
6. **Data Cleansing**: Removing duplicate transactions, correcting misspelled product names, and filling in missing customer information.
7. **Data Integration**: Merging data from various sources to create a unified dataset.

8. **Clean Data in Staging**: The cleaned and transformed data is moved to the "Clean" layer in the Retail_Data_Lake. This data is ready for analysis and reporting.
9. **Data Profiling**: Data profiling tools are used to perform quality checks and analyze the data for patterns. For example, it may identify that a specific product category experiences a spike in sales during a particular season.
10. **Data Auditing and Logging**: Detailed logs are maintained in the data lake to record every step of data transformation and cleaning. This ensures that data lineage is documented for auditing purposes and troubleshooting.

Purpose:

The staging process in this scenario serves several key purposes:

- It consolidates data from diverse sources into a central location, simplifying data management.
- Raw data in the staging area preserves the original data and its history.
- Data transformation and cleansing ensure data quality and consistency.
- Data profiling provides insights into sales patterns, customer behaviors, and other key metrics.
- Clean data is used for advanced analytics, reporting, and decision-making.

With the clean, integrated data in the Retail_Data_Lake, the retail company can generate meaningful reports, identify trends, and make informed decisions to optimize inventory, marketing, and customer experiences across its global stores and online platforms. This example showcases the practical use of the staging process in ETL for a retail organization.

Logical Data Map

Scenario: Imagine you work for a retail company that collects sales data from various **sources,** including in-store **point-of-sale** (POS) systems, an **online store**, and **mobile app transactions**. Your goal is to create a Logical Data Map to illustrate how this sales data is processed through ETL for analysis and reporting.

Data Source	Transformation Process	Data Destination
In-Store POS	Data Extraction	Data Warehouse
Online Store	Data Transformation	Business Intelligence
Mobile App Transactions	Data Loading	Data Marts

1. **Data Sources:**
    - **In-Store POS:** This represents data collected from cash registers and POS systems at physical store locations.
    - **Online Store:** This is data from e-commerce transactions on the company's website.
    - **Mobile App Transactions:** Data gathered from purchases made through the company's mobile app.
2. **Transformation Process:**
    - **Data Extraction:** Data is extracted from each source. For example, from in-store POS systems, you extract sales transaction records.
    - **Data Transformation:** This process involves cleaning, enriching, and standardizing data. For example, transforming raw sales data into a consistent format.

- **Data Loading:** The transformed data is loaded into different destinations.

3. **Data Destinations:**

    - **Data Warehouse:** The final cleaned and transformed data from all sources is loaded into a data warehouse for long-term storage and historical analysis.
    - **Business Intelligence (BI):** This destination is used for creating reports, dashboards, and ad-hoc queries for business analysts to gain insights.
    - **Data Marts:** Specific subsets of data may be loaded into data marts tailored for specific departments or teams, allowing them to access relevant data without querying the entire data warehouse.

In this example, the Logical Data Map provides a simplified view of how sales data flows from multiple sources, goes through ETL processes, and is stored in different destinations for various purposes. This map helps stakeholders understand the data journey within the organization and is a valuable reference for planning, monitoring, and maintaining the ETL processes.

Business Intelligence Data Analysis and Reporting

Intro.

In this assignment, you will explore real-world business data, perform data analysis using BI tools, and create insightful reports. The goal is to gain hands-on experience in data analysis and reporting, a fundamental aspect of Business Intelligence.

Objectives:

- Utilize BI tools to extract, transform, and load (ETL) data.
- Apply data visualization techniques to create meaningful reports.
- Interpret data insights and present them effectively.

Instructions:

a. Data Selection and ETL:

- Choose a dataset relevant to a business scenario or industry (e.g., sales data, customer data, and inventory data).
- Use a BI tool of your choice (e.g., Tableau, Power BI, Excel) to extract, transform, and load the data.

b. Data Analysis:

- Explore the data to identify patterns, trends, and outliers.
- Use appropriate data analysis techniques to answer specific questions related to the dataset (e.g., sales performance analysis, customer segmentation).

c. Data Visualization:

- Create at least three different data visualizations (e.g., bar charts, line graphs, pie charts) based on the analysis.
- Ensure that visualizations are clear, well-labeled, and visually appealing.

d. Report Creation:

- Write a brief report (approximately 500-700 words) summarizing your analysis and findings.
- Include your visualizations in the report to support your insights.

e. Presentation:

Prepare a short presentation (3-5 minutes) to communicate your findings to the class. Use your report and visualizations as a guide.

Marks Distribution:

1. **Data Preparation (4 marks)**
    - Data cleaning and formatting (2 marks)
    - Data organization (2 marks)
2. **Importing Data into BI Tool (3 marks)**
    - Successfully imported data (2 marks)
    - Appropriate use of the chosen BI tool (1 mark)
3. **Data Analysis and Visualization (12 marks)**
    - Exploring and understanding the dataset (3 marks)
    - Appropriate choice of visualizations (4 marks)
    - Building a dashboard (3 marks)
    - Effective use of filters and calculations (2 marks)
4. **Data Insights and Reports (6 marks)**

- Thorough data analysis (3 marks)
- Well-structured and informative reports (2 marks)
- User-friendly dashboard (1 mark)

5. **Overall Quality (5 marks)**

- Clarity of presentation and communication (2 marks)
- Completeness and depth of analysis (2 marks)
- Proper use of the chosen BI tool's features (1 mark)

MIFS410 Advanced Database Distributed Database Scenario

An explanatory scenario designed to help understand the different issues and concepts around **Distributed Databases**, **Centralized DBMS**, and various levels of **Data and Process Distribution**. This scenario incorporates real-world examples to make the concepts clear and interactive, helping grasp the distinctions and challenges involved in distributed systems.

Scenario: Retail Corporation Database Management

Context: Imagine you are a database administrator for a large, multinational retail corporation called **RetailHub**. The company has stores, warehouses, and customers across multiple countries. You are tasked with deciding the best way to manage and distribute the company's database system to handle everything from inventory tracking, customer orders, and deliveries. The data needs to be accessible to different departments (Sales, Finance, Inventory Management, Customer Service) in various locations.

RetailHub is considering three different approaches: **Centralized DBMS**, **Distributed DBMS**, and **Distributed Processing** with different levels of data distribution. Each approach has its own advantages and challenges.

---

1. Centralized DBMS:

Scenario: Currently, all the company's data (e.g., customer information, sales records, inventory data) is stored in a single, central database located at the headquarters in New York. All other branches (e.g., Europe, Asia, etc.) access this data remotely via the Internet.

Issues/Challenges:

- **Single Point of Failure**: If the central database goes down, none of the branches can access any data.

- **Latency**: Remote branches experience slow response times because they are geographically far from the centralized system.
- **Scalability**: As the company grows, the centralized database struggles to handle the increasing load from thousands of users around the world.

Key Concepts:

- **Centralized DBMS**: All data is stored and processed at a single location.
- **Advantages**: Simple to manage, consistent data.
- **Disadvantages**: Poor performance for remote sites, bottlenecks, and single points of failure.

---

2. Distributed Database System:

Scenario: To solve the performance and scalability issues, RetailHub is considering a **Distributed Database System** where data is stored in databases spread across multiple locations (e.g., a database for Europe, one for Asia, one for the Americas). Each branch accesses the local database closest to it, but all databases are connected and share data.

Issues/Challenges:

- **Data Consistency**: Ensuring that all distributed databases have consistent data, especially when multiple locations are updating the same records (e.g., inventory levels).
- **Data Replication**: Deciding whether to replicate data (e.g., store copies of customer data in multiple locations) or partition it (e.g., each region handles its local data). Replication ensures availability but may lead to synchronization issues, while partitioning may limit data availability.
- **Network Latency**: Distributed databases rely on a network, so there could still be delays when synchronizing data between different locations.

Key Concepts:

- **Distributed Database System**: Databases are spread across multiple locations, but all locations work together as a single system.
- **Advantages**: Faster local access, scalability, and fault tolerance.
- **Disadvantages**: Complex management, ensuring consistency, and handling network failures.

---

3. Distributed Processing:

Scenario: RetailHub's IT department suggests a hybrid solution: **Distributed Processing**, where the processing of queries and transactions is distributed across multiple servers, but the data remains centralized at headquarters. For instance, the European branch's transactions are processed by a local server in Europe, but the data is stored in the centralized system in New York.

Issues/Challenges:

- **Network Dependency**: While processing is done locally, every transaction still needs to communicate with the central database, which can slow down operations if network issues arise.
- **Processing Power**: This approach reduces the load on the central system, but it might create additional complexity in managing distributed processes.

Key Concepts:

- **Distributed Processing**: Processing is spread across multiple locations, but the data remains centralized.
- **Advantages**: Reduces processing bottlenecks, faster transaction processing.
- **Disadvantages**: Still reliant on centralized data storage, potential delays due to network dependence.

---

4. Levels of Data and Process Distribution:

Now that the company understands the basics, they explore different **levels of data and process distribution**:

A. Single-Site Processing, Single-Site Data:

- **Scenario**: RetailHub's old system: All transactions (processing) and data storage are handled by a single system at headquarters.
- **Advantages**: Simple to manage, no need for synchronization.
- **Disadvantages**: Scalability and performance issues for remote users, single point of failure.

B. Multiple-Site Processing, Single-Site Data:

- **Scenario**: RetailHub processes customer orders at different regional offices (Europe, Asia, etc.), but all the data is stored at the headquarters.
- **Advantages**: Faster processing at regional offices, reduced load on the central system.
- **Disadvantages**: Still reliant on a single central database for data access, so any central database failure would affect the whole system.

C. Multiple-Site Processing, Multiple-Site Data:

- **Scenario**: RetailHub's ideal solution: Each region (Europe, Asia, etc.) has its own processing servers and local databases. Each regional office processes its own orders, updates inventory, and stores customer data locally.
- **Advantages**: Fast local processing, no dependency on a single site for data or processing, highly scalable.
- **Disadvantages**: Managing data consistency between regions, potential for data replication or synchronization challenges.

Discussion Points for :

1. **What are the advantages and disadvantages of each approach (Centralized DBMS, Distributed DBMS, Distributed Processing)?**
    - Encourage to discuss the trade-offs between simplicity, performance, and complexity in these systems.
2. **What could go wrong in a distributed database system when multiple sites are updating the same data (e.g., inventory levels)?**
    - Explore the potential issues of **data consistency**, **conflicts**, and **synchronization** in distributed databases.
3. **Which level of data and process distribution would you recommend for RetailHub?**
    - You should critically evaluate the different levels (Single-Site vs. Multiple-Site Processing/Data) and argue for the best solution based on RetailHub's needs for scalability, performance, and reliability.

Conclusion:

By using the **RetailHub** scenario, you can better understand the complexities and trade-offs involved in **Distributed Databases**, **Centralized DBMS**, **Distributed Processing**, and the **Levels of Data and Process Distribution**. These real-world examples will help them grasp how different systems handle data, processing, and communication across multiple locations, and they can see how the choice of architecture impacts scalability, reliability, and performance.

Scenario: Distributed Database in SQL

Imagine a retail company that operates in three cities: **New York**, **Los Angeles**, and **Chicago**. Each city has its own branch office, and you want to create a **distributed database** where each office has a local database (a branch database) that stores customer and order information. The main **headquarters** in New York will manage the central master database, but data from other branches will be synchronized regularly.

Step-by-Step Example in SQL:

Step 1: Create the Central Database

1. **Create the Master Database (at Headquarters - New York):**

At the headquarters, you need a central master database to store data from all branches. Here's how you set it up:

```
CREATE DATABASE RetailHQ;

USE RetailHQ;

-- Create central Customers table

CREATE TABLE Customers (

   CustomerID INT PRIMARY KEY AUTO_INCREMENT,

   FirstName VARCHAR(50),

   LastName VARCHAR(50),

   Email VARCHAR(100),

   PhoneNumber VARCHAR(15),

   City VARCHAR(50)

);

-- Create central Products table

CREATE TABLE Products (

   ProductID INT PRIMARY KEY AUTO_INCREMENT,

   ProductName VARCHAR(100),

   Price DECIMAL(10, 2)

);

-- Create central Orders table
```

```
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY AUTO_INCREMENT,
    CustomerID INT,
    OrderDate DATE,
    BranchID INT,
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
);
-- Create central OrderDetails table
CREATE TABLE OrderDetails (
    OrderDetailID INT PRIMARY KEY AUTO_INCREMENT,
    OrderID INT,
    ProductID INT,
    Quantity INT,
    UnitPrice DECIMAL(10, 2),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
-- Add a Branch table to identify which branch each order comes from
CREATE TABLE Branches (
    BranchID INT PRIMARY KEY AUTO_INCREMENT,
    BranchName VARCHAR(50),
    BranchCity VARCHAR(50)
);
-- Insert branch data
INSERT INTO Branches (BranchName, BranchCity) VALUES ('New York HQ', 'New York');
INSERT INTO Branches (BranchName, BranchCity) VALUES ('Los Angeles Branch', 'Los Angeles');
INSERT INTO Branches (BranchName, BranchCity) VALUES ('Chicago Branch', 'Chicago');
```

At this point, the **headquarters database** (RetailHQ) is ready. It contains central tables for storing **Customers**, **Products**, **Orders**, and **OrderDetails**. There's also a **Branches** table to track which branch is associated with each order.

Step 2: Create the Branch Databases

For each branch, you will have a local database that stores data relevant to that specific branch. You will create branch databases for **Los Angeles** and **Chicago**. Each branch will synchronize its data with the central **RetailHQ** database at headquarters.

Create Los Angeles Branch Database:

```
CREATE DATABASE RetailLA;

USE RetailLA;

-- Create local Customers table

CREATE TABLE Customers (

   CustomerID INT PRIMARY KEY AUTO_INCREMENT,

   FirstName VARCHAR(50),

   LastName VARCHAR(50),

   Email VARCHAR(100),

   PhoneNumber VARCHAR(15),

   City VARCHAR(50)

);

-- Create local Orders table

CREATE TABLE Orders (

   OrderID INT PRIMARY KEY AUTO_INCREMENT,

   CustomerID INT,

   OrderDate DATE,

   BranchID INT DEFAULT 2, -- LA Branch

   FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);
```

```
-- Create local OrderDetails table
CREATE TABLE OrderDetails (
    OrderDetailID INT PRIMARY KEY AUTO_INCREMENT,
    OrderID INT,
    ProductID INT,
    Quantity INT,
    UnitPrice DECIMAL(10, 2),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID)
);
-- Insert some test data
INSERT INTO Customers (FirstName, LastName, Email, PhoneNumber, City)
VALUES ('John', 'Doe', 'john@example.com', '123-456-7890', 'Los Angeles');
INSERT INTO Orders (CustomerID, OrderDate) VALUES (1, '2024-10-01');
INSERT INTO OrderDetails (OrderID, ProductID, Quantity, UnitPrice)
VALUES (1, 101, 2, 10.99);
```

Create Chicago Branch Database:

```
CREATE DATABASE RetailChicago;
USE RetailChicago;

-- Create local Customers table
CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY AUTO_INCREMENT,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Email VARCHAR(100),
    PhoneNumber VARCHAR(15),
```

```
    City VARCHAR(50)
);

-- Create local Orders table
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY AUTO_INCREMENT,
    CustomerID INT,
    OrderDate DATE,
    BranchID INT DEFAULT 3, -- Chicago Branch
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
);

-- Create local OrderDetails table
CREATE TABLE OrderDetails (
    OrderDetailID INT PRIMARY KEY AUTO_INCREMENT,
    OrderID INT,
    ProductID INT,
    Quantity INT,
    UnitPrice DECIMAL(10, 2),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID)
);

-- Insert some test data
INSERT INTO Customers (FirstName, LastName, Email, PhoneNumber, City)
VALUES ('Jane', 'Smith', 'jane@example.com', '987-654-3210', 'Chicago');
```

```
INSERT INTO Orders (CustomerID, OrderDate) VALUES (1, '2024-10-02');


INSERT INTO OrderDetails (OrderID, ProductID, Quantity, UnitPrice)

VALUES (1, 102, 3, 15.49);
```

Step 3: Set Up Synchronization Between Databases

1. **Replication (Master-Slave Setup)**: In a distributed system, you can use **replication** to synchronize data between branch databases and the central database. You can configure **master-slave replication** where each branch acts as a slave database and the central HQ acts as the master database.

For example:

   - Changes made at the HQ are replicated to the branches.
   - New orders placed at branch offices (Los Angeles and Chicago) will be sent to the central HQ.

2. **Replication Configuration Example**: Configuring replication will depend on the SQL server you are using (e.g., MySQL, PostgreSQL). Typically, you'd need to configure the my.cnf or postgresql.conf files to enable replication settings, such as:

   - **Master database (HQ) configuration** to log replication events:

```
log_bin = /var/log/mysql/mysql-bin.log

server-id = 1

binlog-do-db = RetailHQ


server-id = 2 # for LA

replicate-do-db = RetailLA
```

3. **Triggers for Data Synchronization**: In addition to replication, you can also set up **triggers** to automatically insert or update data into the central HQ database whenever a change is made to the branch database.

Step 4: Querying the Distributed Database

From Headquarters (Central Database - RetailHQ):

Example: Find all orders placed across all branches.

```
SELECT O.OrderID, C.FirstName, C.LastName, O.OrderDate, B.BranchCity
FROM Orders O
JOIN Customers C ON O.CustomerID = C.CustomerID
JOIN Branches B ON O.BranchID = B.BranchID;
```

From a Branch (e.g., Los Angeles - RetailLA):

Example: Find all orders placed in the Los Angeles branch.

```
SELECT O.OrderID, C.FirstName, C.LastName, O.OrderDate
FROM Orders O
JOIN Customers C ON O.CustomerID = C.CustomerID
WHERE O.BranchID = 2;
```

Step 5: Managing the Distributed Database

1. **Backup**: Ensure regular backups of both central and branch databases. Since the databases are distributed, each branch should maintain its own backup strategy.
2. **Data Consistency**: Ensure **data consistency** using the replication mechanism and periodic consistency checks between branches and the headquarters database.

Summer 2024

Course Code/Name: MIFS315 Data communication and Computer Networks

Assignment Title: Understanding Networking Fundamentals

Assignment Weightage: 30%

Objective: The objective of this assignment is to deepen your understanding of advanced networking fundamentals through both theoretical knowledge and hands-on practice using Cisco Packet Tracer. You will explore key concepts, design a network that includes common network devices, and troubleshoot common networking issues while incorporating advanced networking topics.

Part 1: Theoretical Understanding 15 Marks

1. **Introduction to Networking 3 Marks**
    - Provide an overview of the importance of networking in the modern world.
    - Define key networking terms, such as protocols, IP addresses, routers, switches, and the Internet.
    - Explain the role of networking in connecting devices, enabling communication, and supporting critical applications such as real-time video conferencing and cloud services.

2. **Types of Networks 3 Marks**

    - Discuss different types of networks, including LANs (Local Area Networks), WANs (Wide Area Networks), MANs (Metropolitan Area Networks), and the Internet.
    - Explain the distinctions between these types of networks and provide examples of each.
    - Explore emerging network types like SD-WAN (Software-Defined Wide Area Network) and their impact on modern network architectures.

3. **Networking Protocols 3 Marks**

    - Explore advanced networking protocols such as BGP (Border Gateway Protocol) and OSPF (Open Shortest Path First) used in large-scale routing on the Internet.
    - Discuss security protocols like IPsec (Internet Protocol Security) and SSL/TLS (Secure Sockets Layer/Transport Layer Security) for secure data transmission.
    - Explain the role of IPv6 in addressing the limitations of IPv4 and supporting the future growth of the Internet.

4. **IP Addressing and Subnetting 3 Marks**

    - Deepen your understanding of IP addressing and subnetting:
        - Discuss CIDR (Classless Inter-Domain Routing) notation and its importance in efficient IP address allocation.
        - Explain the concept of VLSM (Variable Length Subnet Masking) and its role in optimizing IP address usage.
        - Provide real-world examples of IP address planning for complex networks, considering factors like scalability, redundancy, and security.

5. **Networking Devices and Architectures 3 Marks**

    - Explore advanced networking devices and architectures:
        - Discuss the role of load balancers in distributing network traffic for high availability and performance.
        - Explain the concept of SDN (Software-Defined Networking) and its potential to revolutionize network management and automation.
        - Describe the principles of network virtualization and the use of technologies like VLANs (Virtual LANs) and VRFs (Virtual Routing and Forwarding) for network segmentation and isolation.

Part 2: Practical Application with Cisco Packet Tracer 12 Marks

---

6. **Network Design in Cisco Packet Tracer with IoT Integration 10 Marks**

    - Use Cisco Packet Tracer to design and configure a small to medium-sized office network. Components include:
        - 2 router (Router1)
        - 2 switch (Switch1)
        - 1 Pc with each switch
    - Assign appropriate IP addresses and subnet masks to each component based on the IP addressing knowledge gained in Part 1.
    - Create a network diagram using Cisco Packet Tracer to visualize the connections and components.

7. **Troubleshooting in Cisco Packet Tracer 2 Marks**

- Simulate common networking issues within your Cisco Packet Tracer-designed network. These issues may include Routing problems affecting IoT device data.

- Identify the issues and apply your troubleshooting skills in Cisco Packet Tracer to resolve them.
- Explain the troubleshooting steps you took, including any commands used in Cisco Packet Tracer, to diagnose and resolve the issues.

Part 3: Assignment Writing Guidelines

---

8. **Assignment Writing Guidelines 3 Marks**

    - Format: Your assignment should be well-structured, with clear headings and subheadings for each section.
    - Clarity: Use clear and concise language
    - Citations and References: Properly cite and reference any external sources used in your assignment. Follow a recognized citation style (IEEEE or Harvard).
    - Plagiarism: Plagiarism is strictly prohibited. Ensure that all content is your own or properly attributed to the source.
    - Submission: 21.7.2024

Conclusion

9. **Conclusion**

    - Summarize the key takeaways from the practical application (Part 2) regarding networking fundamentals, including the integration of IoT devices and advanced networking topics.

- Reflect on the significance of hands-on experience in designing and troubleshooting networks with IoT components and advanced networking concepts.
- Discuss how advanced topics covered in Part 1 contribute to a deeper understanding of networking and its applications in complex and evolving network environments.

Individual Project in System Analysis and Design

Objective:

To apply system analysis and design concepts to a real-world problem of the student's choice, covering all stages from project management to system implementation.

Project Structure and details:

Students must select a system they are interested in developing. This could range from an e-commerce platform, a booking system, an educational tool, to a health management system.

1. Create a project plan detailing objectives, scope, constraints, and a timeline. **(4 Marks)**
2. Conduct a thorough feasibility analysis. **(2 Marks)**
3. Justify the system's selection based on specific criteria relevant to the problem being addressed. **(1 Mark)**
4. Identify functional and non-functional requirements through methods such as observation, interviews, or literature review. **(2 Marks)**
5. Use the following diagrams: **(12 Marks)**
    a. Data Flow Diagrams (DFDs) to represent how data moves through the system, identifying key processes and data stores. (Use draw.io)
    b. Use Case Diagrams: to Illustrate system functionalities and interactions between the system and its users (actors). (Use: UMLet)
    c. Sequence Diagrams to detail the interactions between actors and the system in a specific sequence of events. (use: WebSequenceDiagrams)
    d. Class Diagrams to describe the static structure of the system showing classes, attributes, operations, and relationships. (Use: Class Visualizer)
6. Compile a comprehensive report detailing each step of the project, from initial planning to design and implementation strategy.
7. Present the project, highlighting key aspects of the system analysis and design process, and discuss the application of course concepts. **(5 Marks)**

Remaining Marks distributed as follows:

8. Innovation: The project that introduces a novel approach or solution to a problem, showcasing creativity in design or analysis. **(1 Mark)**
9. Integration: The project that exhibits exemplary integration of different system components and diagrams, presenting a cohesive and fully realized design. **(1 Mark)**
10. Depth of Analysis: The project that goes beyond the surface level, offering deep insights into system requirements, design considerations, and potential impacts. **(1 Mark)**
11. Quality of Presentation: The project is presented in a clear, professional manner, with attention to detail that enhances understanding and engagement. **(1 Mark)**

Notes for Students:

- Choose a project that is both interesting and manageable within the given timeframe.
- Regular consultations with the instructor are encouraged to ensure the project is on track.
- Utilize a variety of tools and resources for diagramming, mockup creation, and project management.

Submission Guidelines:

- The project must be submitted via Moodle according to the project timeline.
- The final report should be professional, well-organized, and include all relevant documentation (diagrams, mockups, analyses, etc.).
- Prepare a presentation summarizing the project work, to be delivered in the specified time.
- ASU Similarity policy will be applied.