GROUP 7

NAME	REG NO	SIGNATURE
Emmanuel Masolo	SC150/0112/2022	
Janet Mutindi	SC150/0800/2022	
Lee Waithaka	SC150/4112/2022	
Faith Muthee	SC150/0793/2022	
Timothy Maina	SC150/0355/2022	

SYSTEM DESIGN

System design is the process of defining the architecture, components, and interfaces for a system so that it meets the end-user requirements. It is a multi-disciplinary field that involves trade-off analysis, balancing conflicting requirements, and making decisions about design choices that will impact the overall system.

System design is used in a wide variety of industries, including software development, hardware development, and manufacturing. It is also used in the design of complex systems such as transportation systems, communication systems, and power grids.

For a system design to take place, there are steps and principles to be followed, patterns and tools to be used.

The system design process typically involves the following steps:

1. **Requirements gathering:** The first step is to gather the requirements for the system. This includes understanding the needs of the users and stakeholders, as well as the functional and non-functional requirements of the system.
2. **System analysis:** Once the requirements have been gathered, the next step is to analyse the system. This involves identifying the different components of the system, their relationships to each other, and how they will work together to meet the requirements.
3. **System design:** Once the system has been analysed, the next step is to design the system. This includes defining the overall architecture of the system, as well as the specific components and technologies that will be used to implement it.
4. **System implementation:** Once the system has been designed, the next step is to implement it. This involves developing the software and hardware components of the system, and integrating them together.
5. **System testing:** Once the system has been implemented, the next step is to test it. This involves verifying that the system meets the requirements and that it works as expected.
6. **System deployment:** Once the system has been tested, the next step is to deploy it. This involves making the system available to the users.
7. **System maintenance:** Once the system has been deployed, it is important to maintain it. This includes fixing bugs, adding new features, and making other changes to the system as needed.

There are a number of principles that should be followed when designing systems include:

1. **Modularity/ Molecularity:** The system should be designed as a collection of modules, each of which performs a specific function. The modules should be independent of each other as much as possible, so that changes to one module do not affect other modules.
2. **Abstraction:** The system should be designed using abstraction layers. Each layer should hide the details of the lower layers from the higher layers. This makes the system easier to understand and maintain.
3. **Data hiding:** The system should be designed to hide data from unauthorized access. This can be done by encapsulating data within objects and modules.
4. **Separation of concerns:** The system should be designed so that each component has a single responsibility. This makes the components easier to understand and maintain.
5. **Scalability:** The system should be designed to be scalable, so that it can handle increased load without performance degradation.
6. **Reliability:** The system should be designed to be reliable, so that it can continue to operate even if some components fail.

System design patterns

There are number of system design patterns that are used. They include:

1. **Layered pattern:** The layered pattern organizes the system into a series of layers, each of which performs a specific function. The layers are stacked on top of each other, and the higher layers rely on the lower layers for support.
2. **Client-server pattern:** The client-server pattern separates the system into two parts: a client and a server. The client is responsible for interacting with the user and sending requests to the server. The server is responsible for processing the requests and sending back the results to the client.
3. **Event-driven pattern:** The event-driven pattern uses events to communicate between different components of the system. When an event occurs, the component that generated the event sends a notification to all other components that are interested in the event.

4. **Micro-services pattern:** The micro-services pattern breaks the system down into a collection of small, independent services. Each service performs a specific task and can be developed, deployed, and scaled independently.

✧ LOGICAL SYSTEM DESIGN

Logical system design is the process of defining the abstract of defining representation of a system, including its inputs, outputs, data flow, and processing logic. It does not specify the specific technologies or components that will be used to implement the system, but rather focuses on the high-level architecture and functionality.

It is typically performed using a variety of modeling techniques, such as Unified Modeling Language(UML) diagrams and flowcharts. These diagrams can help to visualize the different components of the system and how they interact with each other.

It is used to develop a physical system design, which specifies the specific technologies and components that will be used to implement the system.

EXAMPLES OF LOGICAL SYSTEM DESIGNS

- **Designing a database:** A database is a collection of data that is organized in a structured way. Logical system design can be used to design the database schema, which defines the structure of the database and the relationships between the different tables in the database.
- **Designing a software application:** A software application is a program that performs a specific task. Logical system design can be used to design the architecture of the software application, as well as the functionality of each component of the application.
- **Designing a network:** A network is a group of computers that are connected together so that they can communicate with each other. Logical system design can be used to design the topology of the network, as well as the protocols that will be used for communication.

BENEFITS OF LOGICAL SYSYTEM DESIGN

- **Improved communication**: It can help to improve communication between system designers, developers, and stakeholders. This is because it provides a high-level overview of the system and how it will work.

- **Reduced risk:** It can help to reduce the risk of errors and omissions in the system design. This is because it allows system designers to identify and address potential problems early on in the development process.
- **Increased flexibility:** It can help to increase the flexibility of the system. This is because it is not tied to any specific technologies or components. This makes it easier to make changes to the system design as needed.

✧ Physical Design

Physical design, in the context of system analysis and design, refers to the stage in the software development process where you create a detailed plan for how the proposed ‘system will be constructed and implemented. It involves making technical decisions and specifying the actual components, hardware, software, databases, and network infrastructure required to build and deploy the system. Here are some key aspects and considerations related to physical design

Advantages of Physical Design

a) **Concrete Implementation Plan** - Physical design provides a clear and concrete implementation plan for the system. It defines the actual components and technologies that will be used.
b) **Resource Allocation**- It allows for the allocation of specific resources, such as servers, storage, and network equipment, leading to accurate cost estimation and resource provisioning.
c) **Performance Optimization-** Physical design takes into account the performance requirements of the system. It ensures that hardware and software components are selected and configured to meet these requirements.
d) **Scalability** Design decisions are made to support the system's growth and scalability, ensuring that it can handle increased loads and data volumes as needed.
e) **Reliability and Redundancy**- It addresses redundancy and fail over mechanisms to enhance system reliability. Redundant hardware, backup systems, and disaster recovery plans are considered.

f) **Security**- Security measures are incorporated into the physical design, ensuring that the system is protected from unauthorized access, data breaches, and other security threats.

g) **Interoperability-** It addresses how the new system will interact with existing systems, ensuring compatibility and integration with other applications and services.

Disadvantages of Physical Design

1) **Rigidity-** Once the physical design is implemented, it can be challenging to make major changes without significant time and cost implications. This rigidity can be a disadvantage when requirements change.

2) **Resource Intensive**- Creating a detailed physical design can be resource-intensive, requiring a significant investment in terms of time, expertise, and potentially hardware and software purchases

3) **Complexity-** Managing the configuration and integration of various hardware and software components can be complex, especially in large and distributed systems.

4) **Technology Obsolescence-** The technology chosen during the physical design phase may become outdated by the time the system is implemented, requiring updates or replacements.

5) **Limited Flexibility-** The design may not easily accommodate new technologies or changing business needs, which can lead to a lack of flexibility in adapting to evolving requirements.

✧ Architecture design

It is a crucial phase in the system analysis and design process. It focuses on creating a high-level structural blueprint for a system that serves as a foundation for system development and implementation.it is also known as system architecture design.

Advantages of Architecture Design

a) **Structural Clarity-** Architecture design provides a clear and structured overview of the entire system, including its key components, modules, and their interactions.

b) **Scalability**-Well-designed architectures are scalable, allowing the system to grow and handle increased loads or data volumes without major structural changes.

c) **Modularity**- It encourages the use of modular design, making it easier to develop, maintain, and enhance the system. Individual components can be developed and tested independently.

d) **Interoperability-** The architecture addresses how the system interacts with external systems, services, and API's, ensuring compatibility and smooth integration.

e) **Performance Optimization-** Architects consider performance requirements and design decisions to meet those requirements, such as optimizing data retrieval, processing, and communication.

f) **Security**- Security considerations are integrated into the architectural design, ensuring that the system has appropriate security measures to protect data and resources.

g) **Maintainability-** Well-designed architectures make maintenance and updates more straightforward, reducing the risk of unintended side effects when changes are made.

Disadvantages of Architecture Design

a) **Complexity-** Architectural design can be complex, especially for large and intricate systems. Managing all the architectural components and their interactions can be challenging.

b) **Time-Consuming-** Creating a comprehensive architecture takes time, which may delay the development process.

c) **Expertise Required-** Designing a robust architecture requires specialized knowledge and expertise. Architects need a deep understanding of system design principles and best practices.

d) **Rigidity-** If the architecture is overly rigid or inflexible, it can hinder adaptability to changing business requirements or emerging technologies.

e) **Cost-** Developing and implementing a well-defined architectural design may involve additional costs, including the use of specific technologies or tools.

✧ Detailed Design

It is a process of creating a detailed plan or blueprint of a system or product. It is typically done after the high-level design has been completed and all of the requirements have been gathered. The detailed design should specify all of the components of the system or product, how they will interact with each other, and how they will be implemented.

Advantages of detailed design

a) **Reduced risk of errors and omissions-** A detailed design can help to identify and correct errors and omissions before they are implemented. This can save time and money in the long run.

b) **Improved communication and collaboration between team members-** A detailed design can provide a common reference point for team members to work from. This can improve communication and collaboration, and help to ensure that everyone is on the same page.

c) **Facilitates change management-** A detailed design can make it easier to manage changes to the system or product. This is because the design provides a clear understanding of how all of the components interact with each other.

d) **Provides a road map for implementation-** A detailed design can provide a road map for implementing the system or product. This can help to ensure that the implementation is carried out in a systematic and efficient manner.

Disadvantages of detailed design

a) **Can be time-consuming and expensive-** Creating a detailed design can be a time-consuming and expensive process. This is especially true for complex systems or products.

b) **Can be difficult to keep up with changes in the requirements-** The requirements for a system or product can change frequently. This can make it difficult to keep a detailed design up to date.

c) **Can lead to over-engineering-** It is important to avoid over-engineering when creating a detailed design. Over-engineering can lead to a design that is more complex and expensive than is necessary.

✧ DATABASE DESIGN

It is a process of creating a database that is efficient, effective, and scalable. It involves identifying the data that needs to be stored, designing the database schema, and implementing the database. The database schema defines the structure of the database, including the tables, columns, and relationships between them.

Advantages of database design

a) **Improves data quality-** A well-designed database can help to improve the quality of the data that is stored. This is because the database design can enforce constraints on the data, such as data type and range.

b) **Reduces data redundancy-** A well-designed database can help to reduce data redundancy. Data redundancy is the storage of the same data in multiple places. This can waste storage space and lead to data inconsistencies.

c) **Improves data security-** A well-designed database can help to improve data security. This is because the database design can implement security measures, such as access control and encryption.

d) **Enhances data performance-** A well-designed database can help to enhance data performance. This is because the database design can normalize the data and create indexes.

e) **Facilitates data access and retrieval-** A well-designed database can help to facilitate data access and retrieval. This is because the database design can provide a variety of ways to query the data.

Disadvantages of database design

a) **Can be complex and time-consuming-** Designing a database can be a complex and time-consuming process. This is especially true for large and complex databases.

b) **Requires specialized skills and knowledge-** Designing a database requires specialized skills and knowledge. This is because database designers need to have a good understanding of database theory and practice.

c) **Can be difficult to maintain and update-** Maintaining and updating a database can be difficult. This is because database designs can be complex and it can be difficult to keep up with changes in the data requirements.

DATA MODELING TECHNIQUES

It is a critical aspect of system analysis and design, as it involves creating abstract representations of data structures and their relationships within a system. Data models help in understanding and defining how data will be organized and accessed within the system. There are various data modeling techniques used in system design, each with its own strengths and suitability for different scenarios.

Data modeling techniques are also used to create a blueprint of a database or data warehouse. They help to define the entities, attributes, and relationships between the data and are important for creating a database that is efficient, effective, and scalable.

Here are some commonly used data modeling techniques:

1) **Entity-relationship (ER) modeling**

ER modeling is a conceptual data modeling technique that identifies the entities and relationships in a database. ER models are typically represented using a visual diagram, which makes them easy to understand and communicate.

Entities are the real-world objects that are being represented in the database, such as customers, products, and orders. Attributes are the properties of the entities, such as customer name, product price, and order date. Relationships are the connections between the entities, such as a customer placing an order or an order containing multiple products.

ER models can be used to create a variety of different database schemas, including relational, hierarchical, and network databases. ER modeling is a good starting point for any data modeling project, as it provides a high-level overview of the data that needs to be stored in the database.

2) **Relational modeling**

Relational modeling is a logical data modeling technique that defines the database schema. Relational databases are made up of tables, which are related to each other through foreign keys.

Tables are a simple but powerful way to store data. Each table has a row for each record and a column for each attribute. Foreign keys are used to link tables together. For example, an order table might have a foreign key to the customer table, which would allow you to easily find the customer who placed the order.

Relational modeling is the most common data modeling technique used today. It is used to model most types of databases, including relational databases, data warehouses, and data lakes.

3) **Dimensional modeling**

Dimensional modeling is a physical data modeling technique that is used to optimize databases for data warehousing and analytic. Dimensional models are made up of fact tables and dimension tables. Fact tables contain the quantitative data, while dimension tables contain the qualitative data.

Fact tables contain the quantitative data, such as sales figures and website traffic. Dimension tables contain the qualitative data, such as customer demographics and product categories.

Dimensional models are designed to be easy to query and analyze. This makes them ideal for data warehousing and analytics applications.

4) **Hierarchical modeling**

Hierarchical modeling is a data modeling technique that organizes data in a tree-like structure. Each node in the tree can have one or more child nodes, but only one parent node. The top node of the tree is called the root node.

Hierarchical models are often used for modeling data in file systems and object-oriented databases. For example, a file system might be modeled using a hierarchical model, with the root node representing the root directory and the child nodes representing the sub-directories and files.

Another example of a hierarchical model is a table of contents in a book. The root node of the table of contents is the chapter title. The child nodes of the chapter title are the section headings. The section headings can have their own child nodes, which are the subsection headings.

5) **Network modeling**

Network modeling is a data modeling technique that allows for complex relationships between entities. In a network model, any entity can be connected to any other entity. This makes network models well-suited for modeling complex data relationships, such as social networks and transportation networks.

Network models are often used for modeling data in graph databases. Graph databases are a type of database that is specifically designed to store and query graph data.

An example of a network model is a social network. In a social network model, each user is represented as an entity. The entities are connected to each other by relationships, such as friendship and following.

6) **Object-oriented modeling**

Object-oriented modeling is a data modeling technique that is based on the principles of object-oriented programming. Object-oriented models represent data as objects, which are self-contained entities that have their own properties and behaviors.

Object-oriented models are often used for modeling data in object-oriented databases. Object-oriented databases are a type of database that is specifically designed to store and query object-oriented data.

An example of an object-oriented model is a customer object in a CRM system. A customer object would have properties such as name, address, and phone number. It would also have behaviors such as placing an order or returning an item.