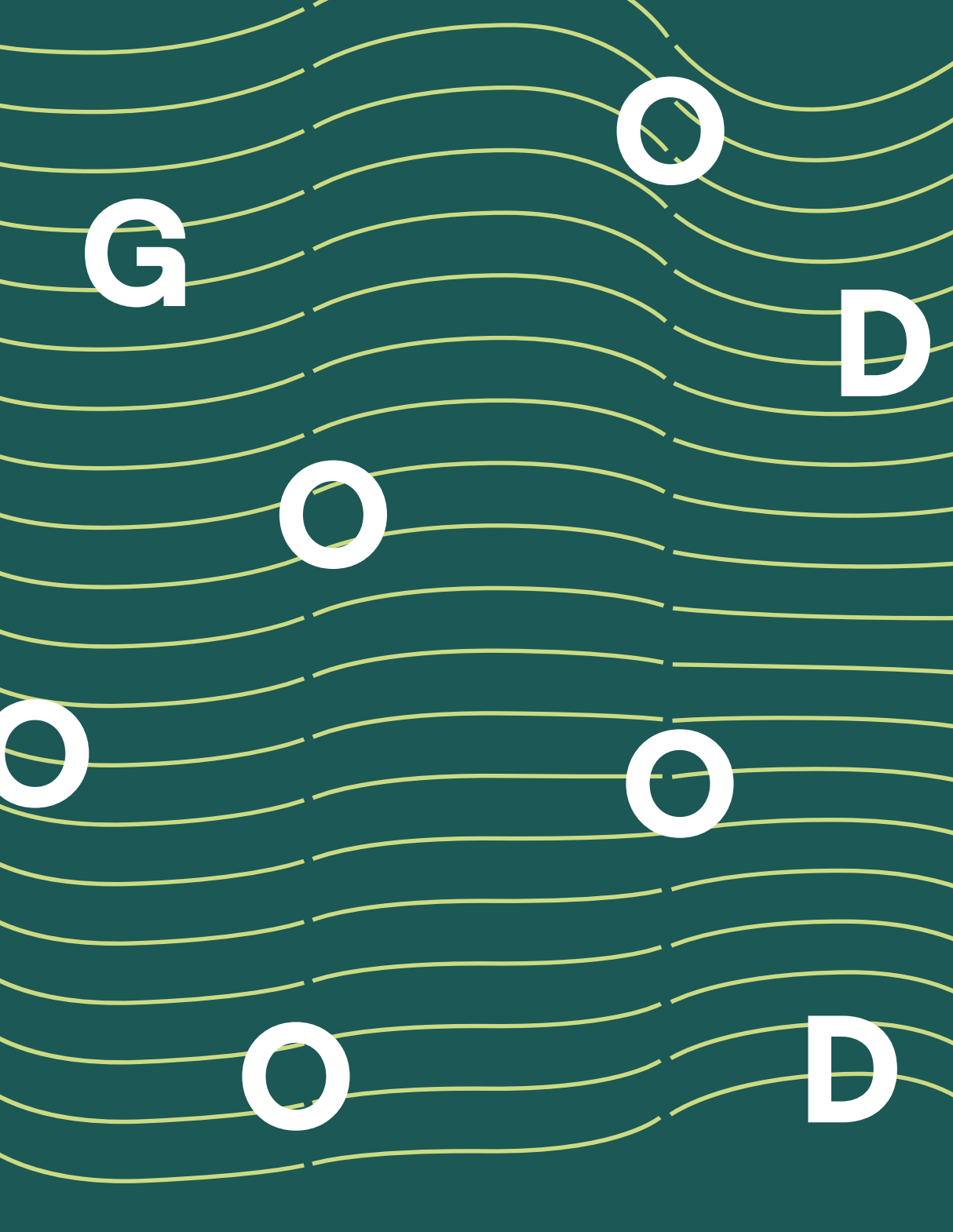
G
O
D
O
O
O
O
D

System Design Strategies

System design strategies are essential for developing effective information systems. These strategies encompass various aspects, including logical and physical design, interface design, and database design. Understanding these components is crucial for creating systems that are both functional and user-friendly.

1. Logical and Physical Design

Logical Design:

Logical design refers to the abstract representation of the data flow, inputs, and outputs of the system. This phase is independent of physical considerations and focuses on what the system will do.

- **Data Flow Diagrams (DFDs):** These diagrams represent the flow of information within the system. They help in understanding the processes and how data moves between them.
- **Entity-Relationship Diagrams (ERDs):** These diagrams depict the relationships between data entities in the system, helping to visualize how data is structured and related.
- **Modeling Processes:** Logical design involves detailed descriptions of system processes, often using tools like flowcharts or pseudocode.

Physical Design:

Physical design translates the logical design into actual physical components, considering hardware and software specifics.

- **Hardware Specifications:** Determining the necessary hardware components, such as servers, workstations, and networking equipment.
- **Software Specifications:** Choosing appropriate software platforms, operating systems, and applications needed to implement the system.
- **Database Design:** Creating detailed database schemas that map logical data models to specific database management systems (DBMS).
- **Interface Design:** Developing the user interface layout and specifications to ensure it

aligns with user requirements and system functionality.

2. Interface Design

Interface design is the process of defining how users interact with the system. It involves creating intuitive and user-friendly interfaces that facilitate effective user interaction.

- **User Interface (UI) Design:** Focuses on the look and feel of the application. This includes designing screens, dialogs, and visual elements that users interact with.
- **User Experience (UX) Design:** Ensures that the system is easy to use and meets the users' needs. This includes usability testing, user feedback, and iterative design improvements.
- **Prototyping:** Creating mock-ups or prototypes of the interface to test design concepts and gather user feedback before full-scale development.
- **Accessibility:** Ensuring the system is accessible to users with disabilities by following guidelines such as the Web Content Accessibility Guidelines (WCAG).

3. Database Design

Database design involves creating a detailed data model that defines the structure of the data within the system. It ensures data integrity, security, and efficiency.

- **Conceptual Design:** Creating a high-level data model, often represented by an ER diagram, to define the entities and relationships.
- **Logical Design:** Translating the conceptual design into a logical model that specifies the data types, relationships, and constraints without considering physical aspects.
- **Physical Design:** Implementing the logical model in a specific DBMS. This involves defining tables, indexes, views, and other database objects.
- **Normalization:** Organizing the database to reduce redundancy and improve data integrity. This involves dividing large tables into smaller, related tables.
- **Optimization:** Enhancing the performance of the database through indexing, query optimization, and other techniques.

A. Data Flow Diagram (DFD)

A Data Flow Diagram (DFD) is a graphical representation of the flow of data through an information system. It illustrates how data is processed by a system in terms of inputs and outputs. DFDs are used to visualize the data processing steps and can be helpful in understanding the system's functionality and identifying potential areas for improvement.

Components of a DFD:

1. **External Entities:**
    1.
    2.
2. **Processes:**
    1.
    2.
3. **Data Stores:**
    1.
    2.
4. **Data Flows:**
    1.
    2.

Types of DFD:

1. **Context Diagram:**
    1.
    1.
1. **Level 0 DFD:**
    1.
    1.
1. **Level 1 DFD:**
    1.
    1.

Benefits of Using DFDs:

- **Clarity:** Helps in understanding the flow of data and the interactions within the system.
- **Communication:** Facilitates communication among stakeholders by providing a clear visual representation of the system.
- **Analysis:** Identifies inefficiencies and potential improvements in data processing.
- **Documentation:** Acts as a documentation tool for system analysis and design.

Example of a DFD:

Scenario: Online Ordering System

1. **External Entities:**

1.

2.

1. **Processes:**
    1.

    2.

    3.

2. **Data Stores:**
    1.

    2.

3. **Data Flows:**
    1.

    2.

    3.

    4.

Click the link below for more information on this topic

==> https://www.visual-paradigm.com/guide/data-flow-diagram/what-is-data-flow-diagram/

B. Entity-Relationship Diagram (ERD)

An Entity-Relationship Diagram (ERD) is a graphical representation of an information system that illustrates the relationships between entities within that system. It is widely used during the design phase of a database system to define the data structure and the relationships among data elements.

Components of an ERD:

1. **Entities:**
    1. ○ **Definition:** An entity represents a real-world object or concept with a distinct existence in the domain being modeled. Entities can be tangible (e.g., a person, a product) or intangible (e.g., a transaction).
    2.
    3.
2. **Attributes:**
    1.
    2.
    3.
3. **Relationships:**
    1.
    2.
    3.

1. **Primary Key:**
    1.
    2.
    3.
2. **Foreign Key:**
    1.
    2.
    3.
3. **Cardinality:**
    1.
    2.
    3.

Types of Relationships

1. **One-to-One (1:1):**
    1.
    1.