Louneda Philippe
Lance McPherson

SYSTEM IMPLEMENTATION

April 5, 2024

System Architecture and System Design

Architectural Styles:

Our system is designed using a client-server architectural style. This approach allows for the separation of concerns between the client-side application and the server-side components, providing flexibility, scalability, and ease of maintenance. The client-server model enables efficient communication between the client application and the server, facilitating tasks such as user authentication, data processing, and storage.

Identifying Subsystems:

In our system, we have identified several subsystems that collectively contribute to the functionality of the overall system. These subsystems include the client application and various server-side components.

- Client Application: The client application serves as the primary interface for users to interact with the system. It facilitates tasks such as user authentication, dream customization, and dream sharing. Additionally, the client application handles user input and communicates with the server-side components to retrieve and display relevant data.
- Server-side Components: The server-side components encompass various modules responsible for handling specific functionalities of the system. These modules include user authentication, dream customization, dream sharing, and data storage. Each server-side component is designed to perform a specific set of tasks efficiently, ensuring seamless operation of the overall system.

UML Package Diagram:

Diagrams goes here

Mapping Subsystems to Hardware:
Our system is designed to run on multiple computers, with distinct subsystems allocated to different hardware resources.

- Client-side Subsystems: The client-side subsystems run on users' devices, such as computers, smartphones, or tablets. These devices serve as the platform for users to access the client application and interact with the system.
- Server-side Subsystems: The server-side subsystems run on dedicated servers hosted in a data center or cloud environment. These servers are equipped with the necessary hardware resources, including processing power, memory, and storage capacity, to support the execution of server-side components and handle concurrent user requests.

Persistent Data Storage:
Our system requires persistent data storage to store user information, dream creations, and other essential data. To meet this requirement, we have chosen a relational database management system (RDBMS) as the storage solution.

- Relational Database Management System (RDBMS): RDBMS offers several advantages, including robustness, scalability, and support for complex data queries. Additionally, RDBMS provides ACID (Atomicity, Consistency, Isolation, Durability) compliance, ensuring data integrity and reliability.

Network Protocol:
Our system utilizes the HTTP protocol for communication between the client and server components. HTTP (Hypertext Transfer Protocol) is a widely adopted protocol for transmitting data over the internet. It offers a standardized and platform-independent approach to facilitate communication between clients and servers.

Global Control Flow:

- Execution Orders: Our system follows an event-driven architecture, where user actions trigger corresponding events and actions within the system. Users have the flexibility to generate actions in different orders based on their interactions with the client application.
- Time Dependency: While our system does not have strict real-time constraints, certain tasks may involve time-dependent operations. For example, session timeouts may be implemented to manage user sessions, and scheduled tasks may be executed at predefined intervals for maintenance purposes.

- Concurrency: Our system employs multiple threads to handle concurrent user requests and background tasks efficiently. Synchronization mechanisms are implemented to ensure thread safety and prevent data corruption in multi-threaded environments.

Hardware Requirements:

- Client Devices: Users require modern web browsers with support for HTML5, CSS3, and JavaScript to access the client application. The client application should be compatible with various devices, including desktop computers, laptops, smartphones, and tablets.
- Server Hardware: The server hardware should meet the system's performance and scalability requirements. This includes sufficient processing power, memory, and storage capacity to handle concurrent user requests and data processing tasks effectively. Additionally, reliable network connectivity is essential to ensure seamless communication between client and server components.

By employing a client-server architectural style, leveraging appropriate subsystems, utilizing persistent data storage, adhering to standard network protocols, managing global control flow effectively, and addressing hardware requirements, our system architecture and design aim to provide a robust, scalable, and user-friendly platform for users to interact with DreamForge seamlessly.