High Level Design

Prateek Shukla | Software Engineer

Airtribe

Agenda

- What is system design?
- What is High Level System Design?
- System Components
- Interview Questions and the approach.
- Functional vs Non-Functional Requirements
- Back of envelope calculation

System Design

Architecture

Choosing the overall structure of the system, such as whether it will follow a monolithic, microservices, or other architectural pattern. This decision impacts how different parts of the system communicate and scale.

Components and Modules

Breaking down the system into smaller components or modules that can be developed and tested independently. Each module should have a well-defined responsibility.

System Design

Interfaces

Defining how different components will communicate and interact with each other. This includes specifying APIs, protocols, and data formats.

Data Management

Breaking down the system into smaller components or modules that can be developed and tested independently. Each module should have a well-defined responsibility.

System Design

Fault Tolerance and Reliability

Designing the system to continue functioning even in the presence of failures. This might involve redundant components, failover mechanisms, and error handling strategies.

User Experience

Designing the user interface and interaction flows to ensure a positive and intuitive user experience.

System Design

Technology Stack

Selecting the appropriate programming languages, frameworks, libraries, and tools for building different parts of the system.

Trade-offs

Making decisions and trade-offs between different design options, considering factors like development time, complexity, and long-term maintainability.

System Components

- Client Server Architecture
- Application Server
- DNS
- Load Balancer
- Databases
- Queues
- Caches

Interview Questions & Approach

- **Clarifying Requirements**
  - Will users of our service be able to post tweets and follow other people?
  - Should we also design to create and display the user's timeline?
  - Will tweets contain photos and videos?
  - Are we focusing on the backend only or are we developing the front-end too?
  - Will users be able to search tweets?
  - Do we need to display hot trending topics?
  - Will there be any push notification for new (or important) tweets?

Interview Questions & Approach

- **Back-of-the-envelope estimation**
  - What scale is expected from the system (e.g., number of new tweets, number of tweet views, number of timeline generations per sec., etc.)?
  - How much storage will we need? We will have different storage requirements if users can have photos and videos in their tweets.
  - What network bandwidth usage are we expecting? This will be crucial in deciding how we will manage traffic and balance load between servers.

Interview Questions & Approach

- **System interface definition**
  - **postTweet**(user_id, tweet_data, tweet_location, user_location, timestamp, ...)
  - **generateTimeline**(user_id, current_time, user_location, ...)
  - **markTweetFavorite**(user_id, tweet_id, timestamp, ...)
  - ....

Interview Questions & Approach

- **Defining data model**
  - **User**: UserID, Name, Email, DoB, CreationData, LastLogin, etc.
  - **Tweet**: TweetID, Content, TweetLocation, NumberOfLikes, TimeStamp, etc.
  - **UserFollow**: UserdID1, UserID2
  - **FavoriteTweets**: UserID, TweetID, TimeStamp

Interview Questions & Approach

- **High-level design**
  - Pictorial representation of overall system
- **Detailed design**
  - Scalability
  - Security
  - Handle Traffic
  - Cache
- **Identifying and resolving bottlenecks**
  - Failure tolerant
  - Debugging/Monitoring

Functional Requirements

- **Definition:** These specify what a system should do. They describe the functions, features, and interactions that the system must have. Functional requirements are typically the "what" of a system. They are often specific and quantifiable.

Airtribe

Functional Requirements

- **Example:**
  - Will users of our service be able to post tweets and follow other people?
  - Should we also design to create and display the user's timeline?
  - Will tweets contain photos and videos?
  - Users receive notifications for likes, retweets, and mentions.
  - Users can search for tweets based on keywords, hashtags, or usernames.
  - Users can include mentions (@username) and hashtags (#hashtag) in their tweets.
  - Users can view their followers and users they are following.

Non-Functional Requirements

- **Definition:** These specify how a system should perform its functions. They describe the qualities and constraints that the system must adhere to. Non-functional requirements are often the "how" of a system and are not always quantifiable in the same way as functional requirements.

Airtribe

Non-Functional Requirements

- **Example:**
  - The system should provide high availability, with 99.9% uptime.
  - The system should handle a large number of users and tweets efficiently.
  - The system should be highly available and reliable.
  - The system should respond quickly to user actions, such as posting tweets or loading timelines.
  - User data should be stored securely.
  - Tweets and user data should not be lost or corrupted.
  - The system should have monitoring and logging in place to detect and diagnose issues.
  - User privacy should be respected.

Back of the envelope calculations

Thank You!

See you again on next date here

Airtribe