Mentimeter

Instructions

39

Mentimeter

Introduction to Databases and Networking

COMP1048: Databases and Interfaces (2024-2025)

Matthew Pike and Yuan Yao

4

Mentimeter

Overview

2

This Lecture

- Introduction to databases - definition, uses, problem to solve
- Database management systems - improved data management
- DBMS architecture - ANSI/SPARC model
- Networking basics - TCP/IP, IP addresses, DNS, URLs and HTTP

Databases

What is a database?

- *"A collection of data arranged for ease and speed of search and retrieval."*
  - American Heritage Science Dictionary
- *"A structured set of data held in computer storage."*
  - Oxford English Dictionary
- *"One or more large structured sets of persistent data, usually associated with software to update and query the data."*
  - Free Online Dictionary of Computing

Databases are (virtually) everywhere!

- Library catalogues
- Medical records
- Bank accounts
- Stock market data
- Personnel systems
- Product catalogues
- Telephone directories
- Train timetables
- Airline bookings
- Credit card details
- Student records
- Customer histories
- Stock market prices
- Pretty much every website you ever use!
- This list is not exhaustive

2

The Problem

"We want to store our data in a format that allows us to easily query, update, insert and delete without affecting the integrity of the data."

- **Example**: Student Records
  - **Insert**: New Students
  - **Update**: Students details change
  - **Delete**: Student leaves
  - **Query**: How many students study CS?

Figure 1: How would you find your student record in this pile of papers?

The Initial Solution

- Applications store their data in files.
- Each file has its own format defined by the application itself.
- Program has to know format.
- Any other program using the file has to know format.

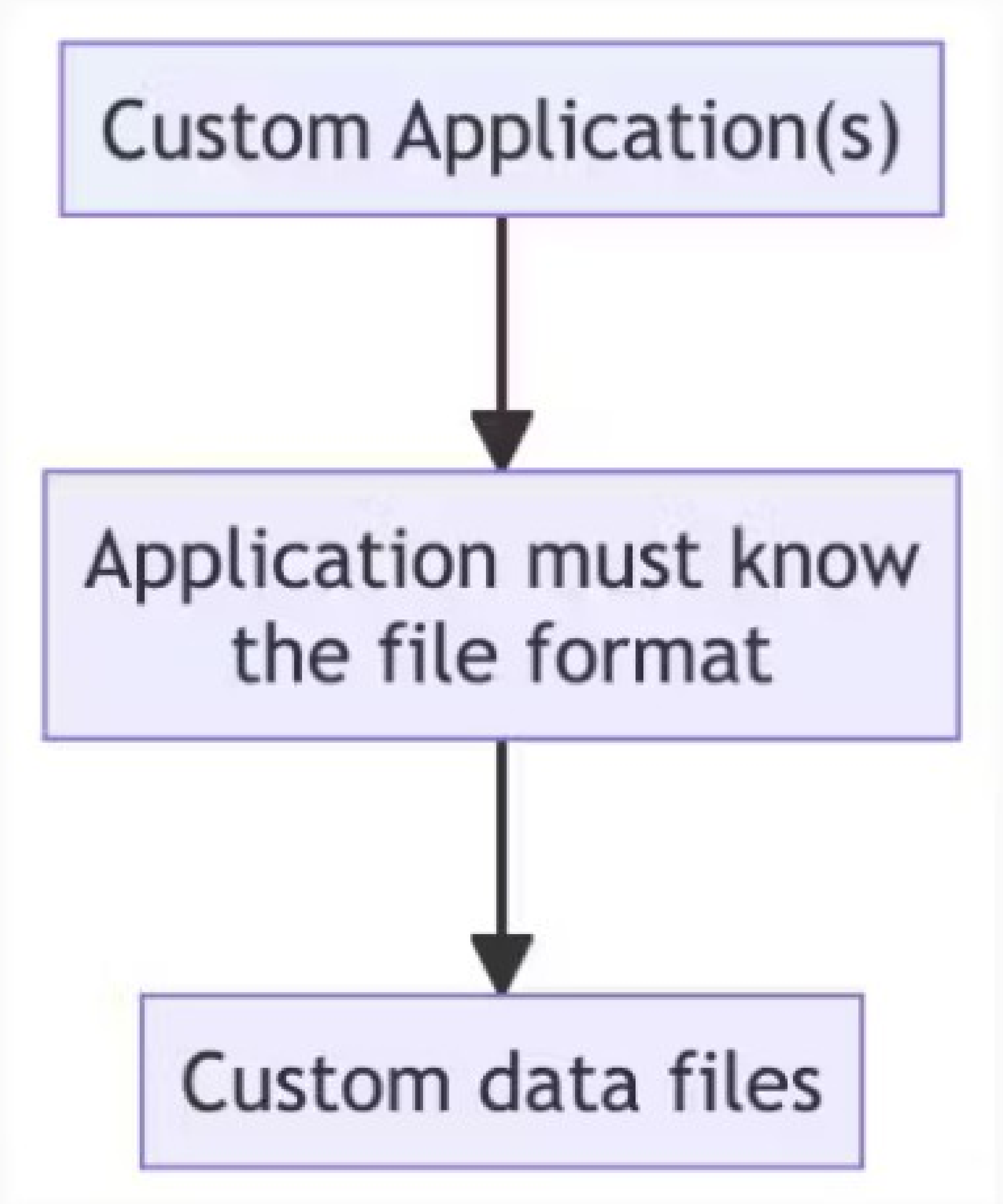


Figure 2: An example of an application using a bespoke data storage format.

Mentimeter

What potential issues and challenges arise from using a bespoke (custom made) data storage format?

Others may don't know how to use

The data cannot be easily used by another client/app

Incompatibility with other devices/ software making it impossible to scale it / share it

Bad compatibility

repeated data storage

Someone may eat the paper

dont know how to useI've use mongoDB

we can, but it will be intricate

7

41

Mentimeter

What potential issues and challenges arise from using a bespoke (custom made) data storage format?

Not effective	Not properly applied	Format of the messages are not unified	leak of privacy
Incompatibility / inability-to-read with other programs. Scalability becomes immensely difficult	lack of universality	not suitable for another data	data loss

7 41

Mentimeter

What potential issues and challenges arise from using a bespoke (custom made) data storage format?

It's hard for other systems to use the data

It will be personal used

too much memory

unformatted data structure causes issues

too many data format type

It will be hard for other application to reuse

It may easily attacked by hacker and have the risk of data leakage

not standard sp have to train people to use it

7

41

Mentimeter

What potential issues and challenges arise from using a bespoke (custom made) data storage format?

hard to manage

the interface may hard to use

Not legit to use

No because if they don't have a standard, they cannot be understand by other application except the programmer and make no sense in a broad context

A bespoke system may be difficult to operate for people familiar with more standard formats, alongside other issues of incompatibility

Hard to used

cross-reference

too expensive and low cost performance

7

41

Mentimeter

What potential issues and challenges arise from using a bespoke (custom made) data storage format?

N/A

Each apps' data will have its own format and it may cause chaos when others want to use it.

not reuseable

low effeciency

lack of versatility

Expandability is weak

the path may different

complex

7

41

Mentimeter

What potential issues and challenges arise from using a bespoke (custom made) data storage format?

Poor applicability

bad compatibility

Others may not to understand and use it

some security issue (like sql injection)

Tedious

hard to operate in a standard way

7

41

Problem: Multiple Applications

- No standards.
- Incompatible file formats.
- Data duplication.
- Data dependence.
- Fixed queries.
- Concurrency.
- Security.
- No theoretical foundations.

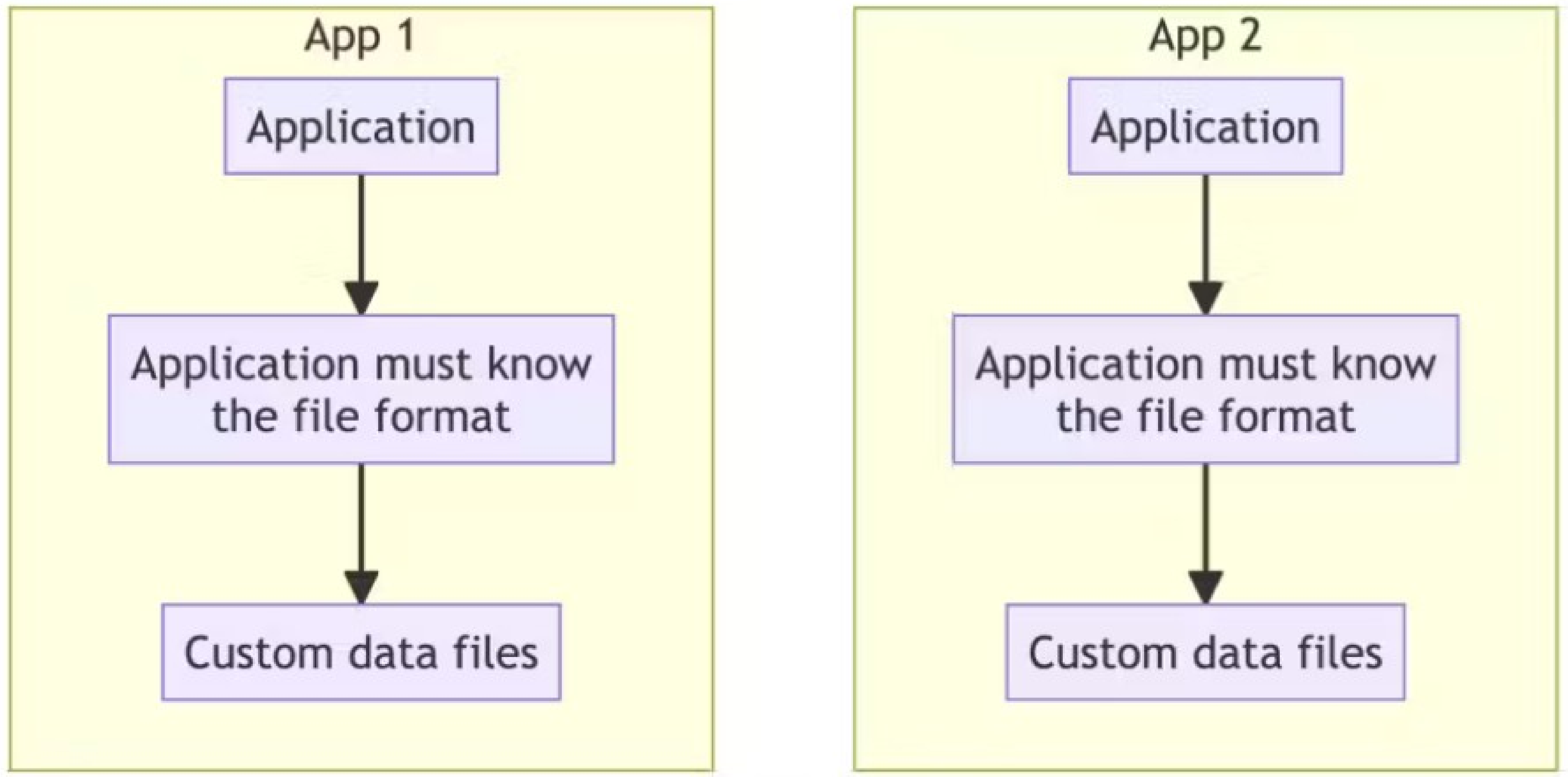

App 1
Application
Application must know the file format
Custom data files
App 2
Application
Application must know the file format
Custom data files

Figure 3: In this less-than-ideal approach, each application needs to maintain its own data storage format. This significantly adds to the Software Engineering load borne by the developers.

A Better Approach: Using a Database Management System (DBMS)

- A database management system (DBMS) is system software for creating and managing databases.
- It provides:
  - Data Definition Language (DDL) - defines data structures
  - Data Manipulation Language (DML) - inserts, updates, deletes data
  - Data Control Language (DCL) - controls access to data
  - Data Query Language (DQL) - queries and reports on data
- Structured Query Language (SQL) is used for data definition, manipulation and queries.
- A DBMS provides centralised data management and abstracts storage details.
- This improves data consistency, integrity, security, and access.

2

A Better Approach: Using a DBMS (cont)

Mentimeter

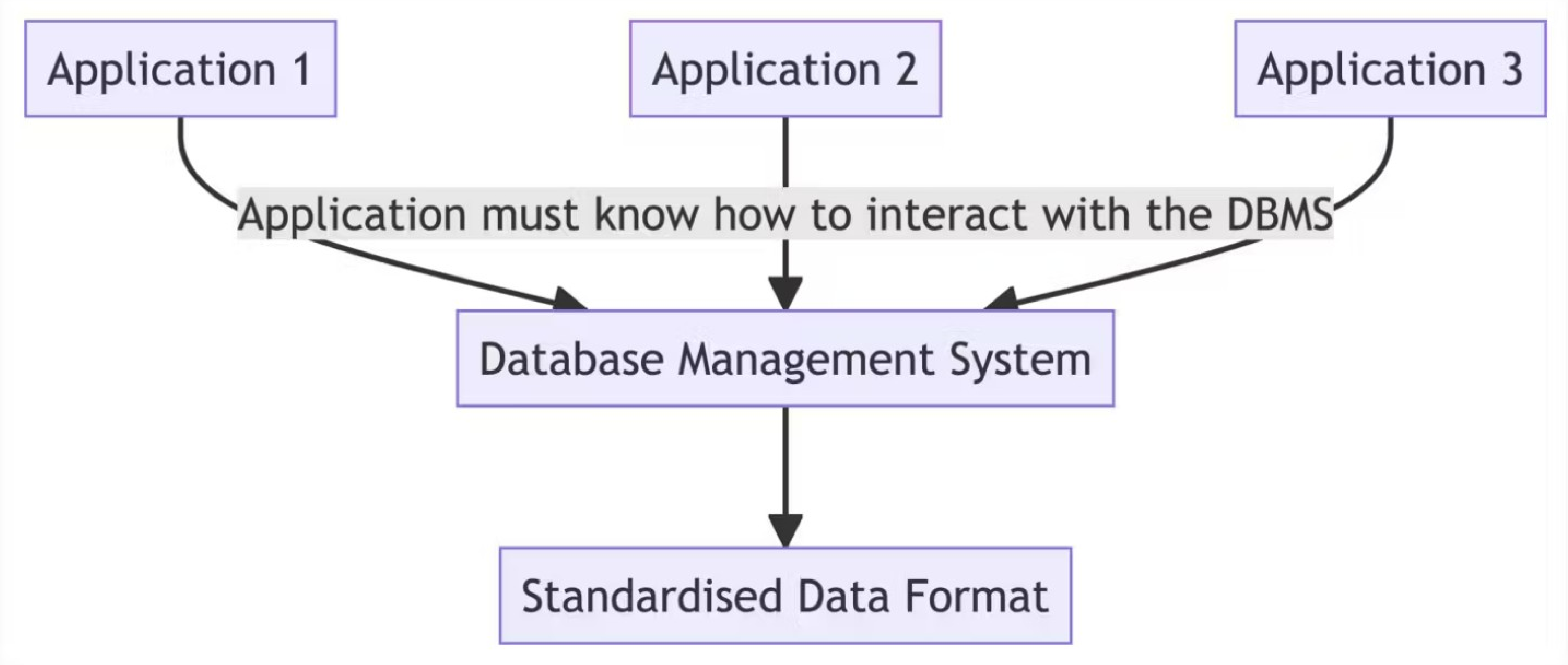

Application 1
Application 2
Application 3
Application must know how to interact with the DBMS
Database Management System
Standardised Data Format

Figure 4: Instead of creating bespoke file formats, applications can now utilise a standard interface, delivered by the DBMS, for the Creation, Reading, Updating and Deletion of application data.

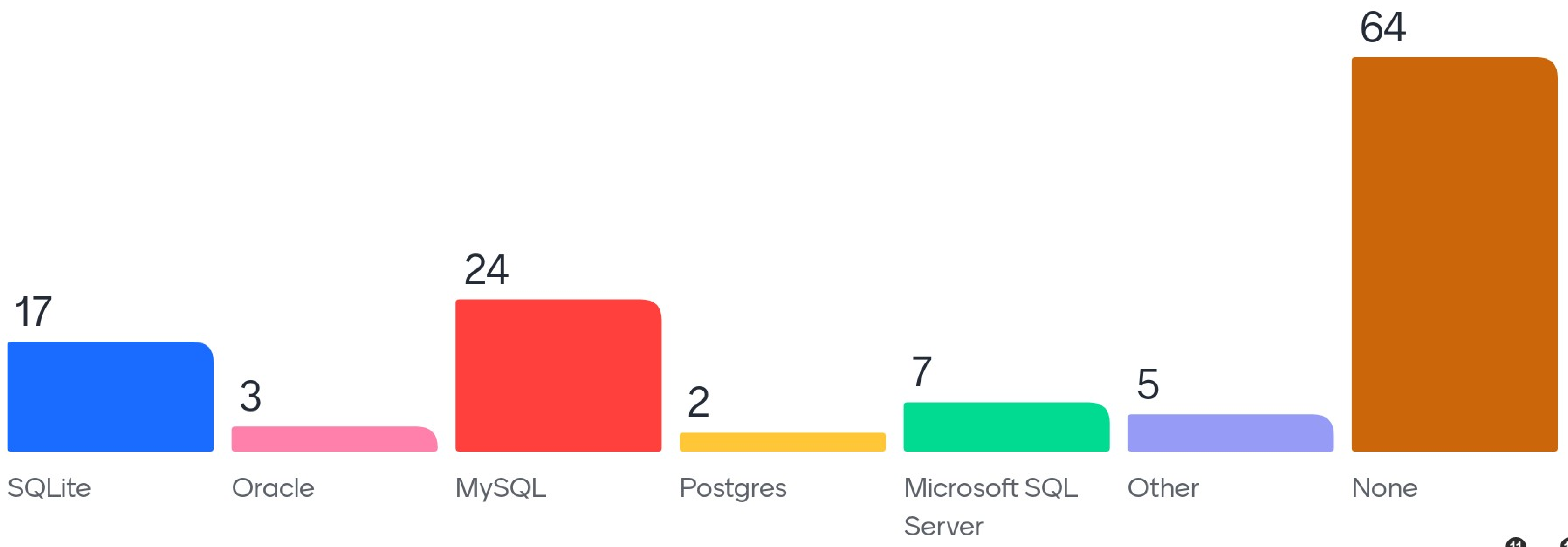
DBMS Functionalities

- Allow users to store, retrieve and update data.
- Ensure either that all the updates corresponding to a given action are made or that none of them is made.
- Ensure that DB is updated correctly when multiple users are updating it concurrently.
- Recover the DB in the event it is damaged in any way.
- Ensure that only authorised users can access the DB.
- Be capable of integrating with other software.

- Examples of Modern DBMS
  - SQLite - https://www.sqlite.org/index.html
  - Oracle - https://www.oracle.com/database/
  - MySQL - https://www.mysql.com/
  - PostgreSQL - https://www.postgresql.org/
  - Microsoft SQL Server - https://www.microsoft.com/en-us/sql-server/

Mentimeter

Have you used any of the following DBMSs already?



11 100

ANSI/SPARC Architecture

- Three tier/level architecture:
  - **External level**: This is the level that database users interact with. It presents data in a way end users can understand, abstracting unnecessary complexity.
  - **Conceptual level**: This part is primarily for database designers. It provides a holistic view of the entire database independent of the physical implementation, hiding details like storage locations and data pathways.
  - **Internal level**: This is meant for the system designers and deals with the physical storage of data. Working behind the scenes, it focuses on how data is stored and accessed, optimising for speed and efficiency.
- The goal is to separate the user-facing external views from the physical database implementation details.
  - This provides abstraction and independence between the levels, allowing changes to be made at one level without affecting the others.

5

ANSI/SPARC Architecture (cont)

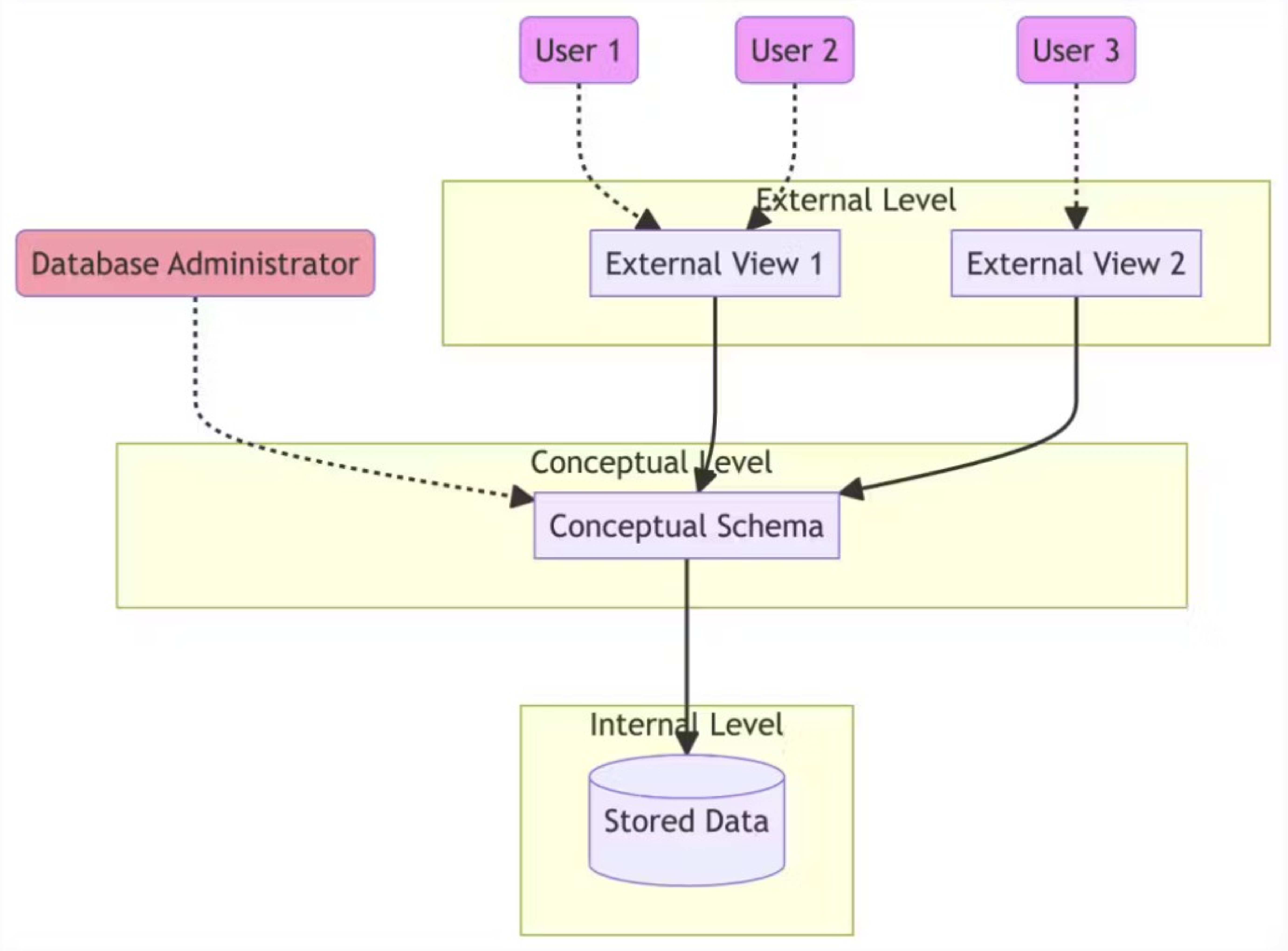


Figure 5: The ANSI/SPARC Architecture.

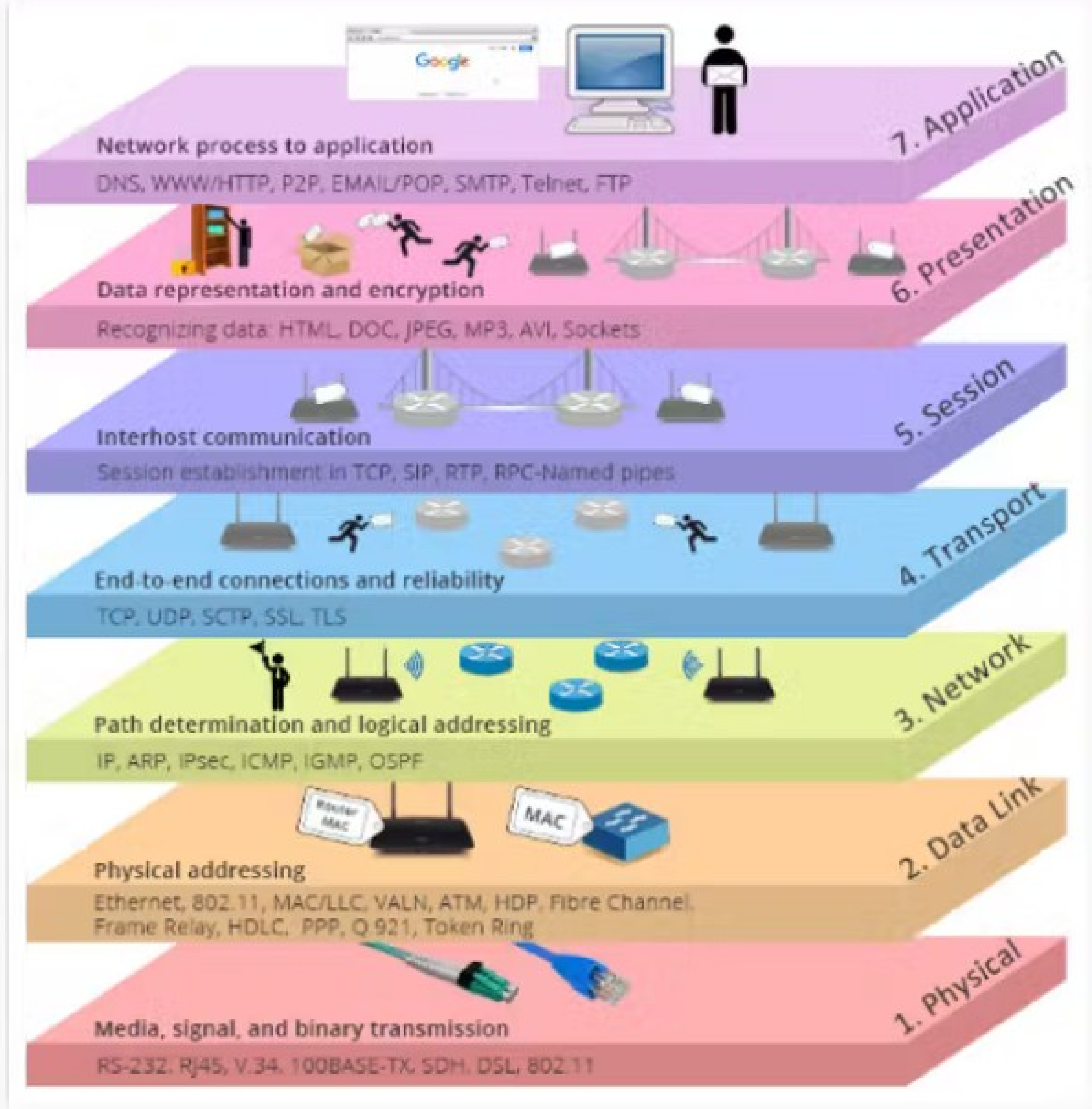
Networking

TCP/IP

Mentimeter

We will not be covering the OSI model in this module. It will be covered in the SYS module.

- Transmission Control Protocol/Internet Protocol (TCP/IP) allows devices on the Internet to communicate with each other.
- Internet Protocol (IP) addresses uniquely identify devices connected to the Internet.
  - IP addresses are four-part numbers (e.g. 192.168.1.1)
- This module will primarily operate at the Application Layer of the OSI model.



Google
7. Application
Network process to application
DNS, WWW/HTTP, P2P, EMAIL/POP, SMTP, Telnet, FTP
6. Presentation
Data representation and encryption
Recognizing data: HTML, DOC, JPEG, MP3, AVI, Sockets
5. Session
Interhost communication
Session establishment in TCP, SIP, RTP, RPC-Named pipes
4. Transport
End-to-end connections and reliability
TCP, UDP, SCTP, SSL, TLS
3. Network
Path determination and logical addressing
IP, ARP, IPsec, ICMP, IGMP, OSPF
2. Data Link
Router MAC
MAC
Physical addressing
Ethernet, 802.11, MAC/LLC, VALN, ATM, HDP, Fibre Channel, Frame Relay, HDLC, PPP, Q.921, Token Ring
1. Physical
Media, signal, and binary transmission
RS-232, RJ45, V.34, 100BASE-TX, SDH, DSL, 802.11

Figure 6: Hierarchy of the OSI model.

3

Domain Name

- Domain names are meaningful to humans, but not to machines.
- Domain names are translated into IP addresses by the Domain Name System (DNS), which are meaningful to machines.
  - DNS is a distributed database that maps domain names to IP addresses.
- Begins with the name of the host machine, followed by progressively larger collections of machines i.e., subdomains.

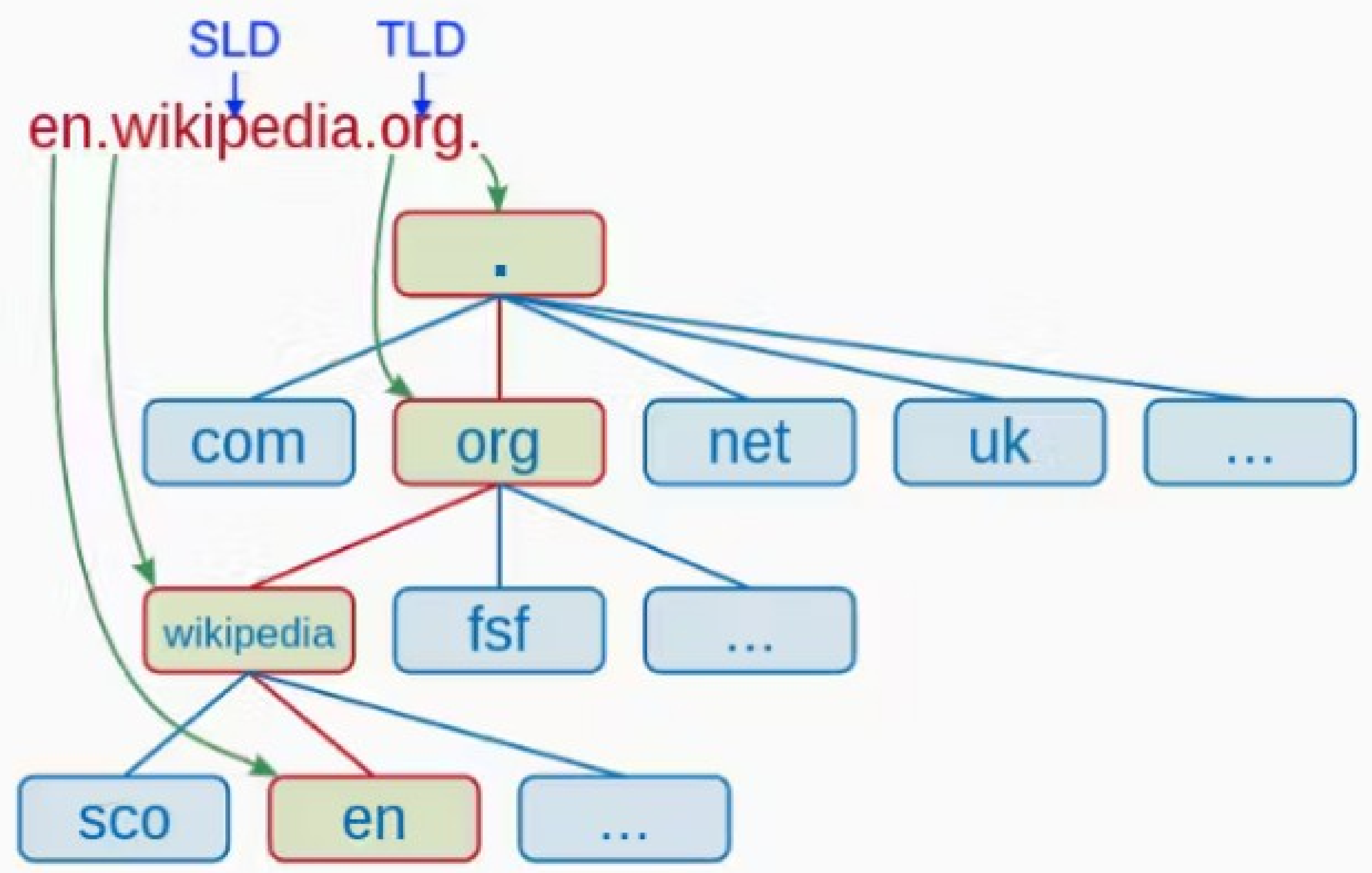


Figure 7: Components of a Domain Name.

Uniform Resource Locator (URL)

- A URL is a web address that specifies the location of a resource on the internet. It points to the unique location of a file, page, or other web resource.
- The structure of a URL typically begins with the protocol (https://), then the domain name, path to the resource, and optional parameters.
    - The protocol is often the HTTP protocol, but can also be FTP, HTTPS, etc.

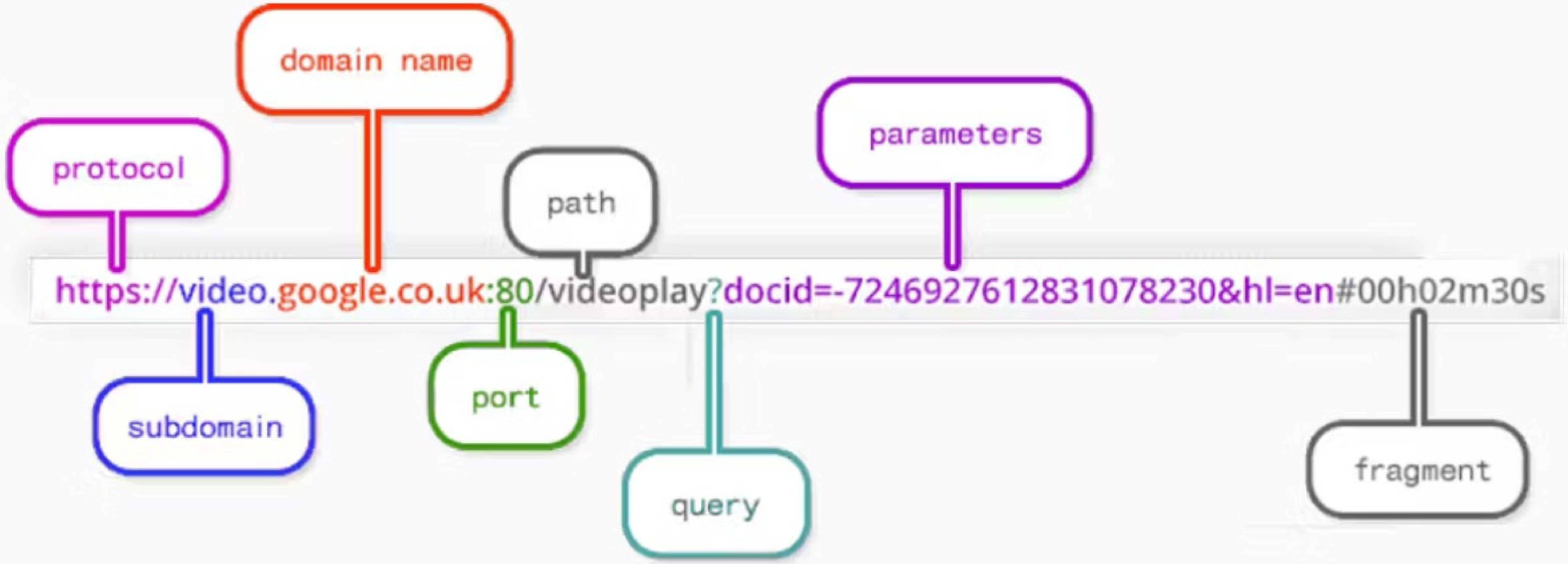

domain name
protocol
path
parameters
https://video.google.co.uk:80/videoplay?docid=-7246927612831078230&hl=en#00h02m30s
subdomain
port
query
fragment

Figure 8: An example of a valid URL with annotated components.

Hypertext Transfer Protocol (HTTP)

- HTTP is the underlying protocol used by the World Wide Web to define how messages are formatted and transmitted between web browsers and servers. HTTP works on a request-response model (Shown in Figure 9).
- There are various HTTP request methods that specify the type of action to be performed on a resource.
  - Examples include: GET, POST, PUT, DELETE.
  - We'll look at GET and POST in more detail later in the module.
- HTTP messages consist of a header and a body. The header contains metadata like request method, URLs, status codes. The body contains the resource content.
- HTTP is stateless, meaning the server does not retain any state about previous requests.

HTTP (cont)

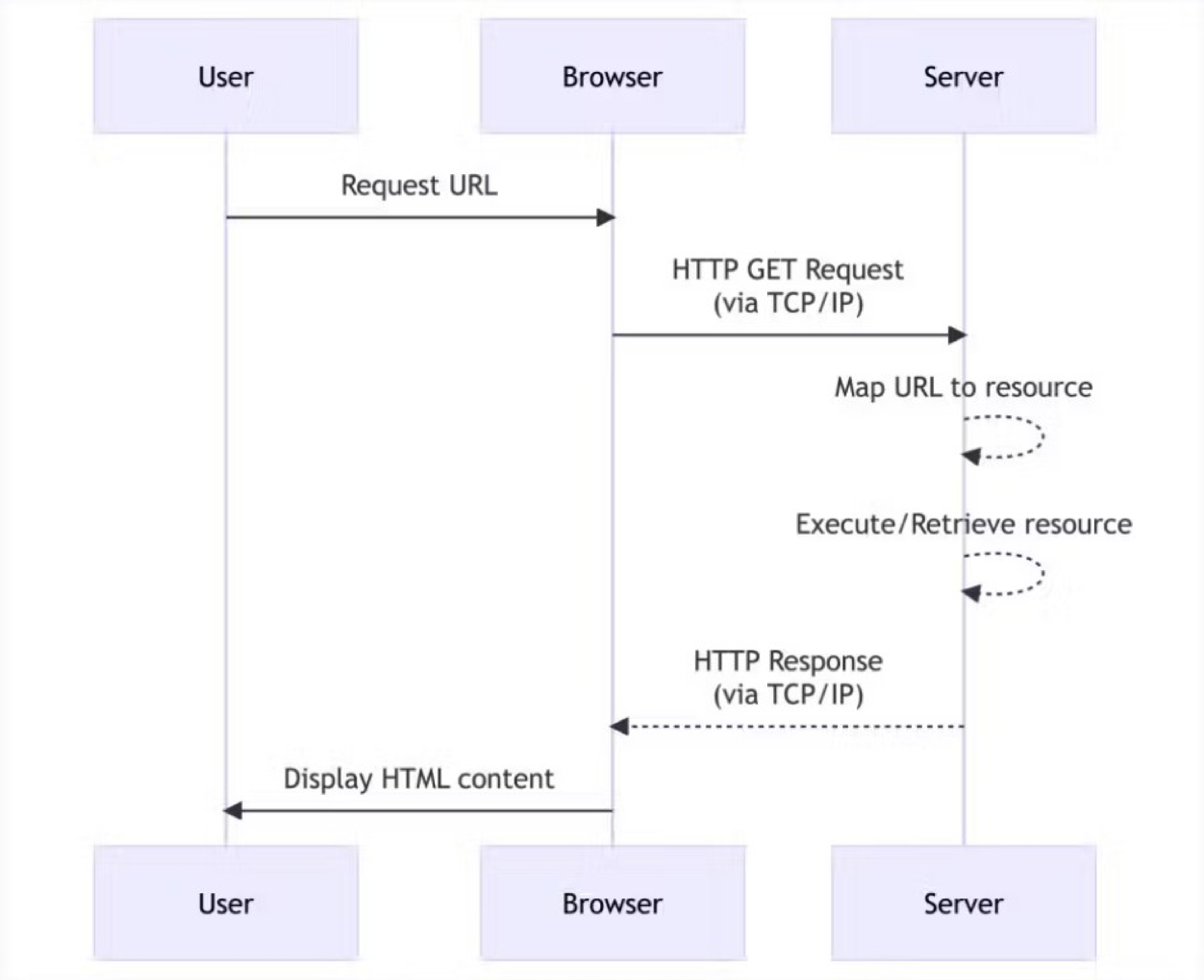


Figure 9: Example demonstrating the stages of a HTTP request/response cycle.

Networking Summary

Mentimeter

- Devices connected to the Internet rely on the TCP/IP protocol suite to communicate.
- IP addresses are four-part numbers (e.g. 192.168.1.1) that uniquely identify devices on a network.
- DNS (Domain Name System) translates domain names that are meaningful to humans into the numerical IP addresses.
- URLs (Uniform Resource Locators) specify the location of resources on the web (e.g. a web page, image, video file).
- HTTP (Hypertext Transfer Protocol) is the fundamental protocol used for communication over the web.
- The most common HTTP request methods are GET for retrieving data, and POST for submitting data.

Mentimeter

Any Questions?

11 questions
21 upvotes

1