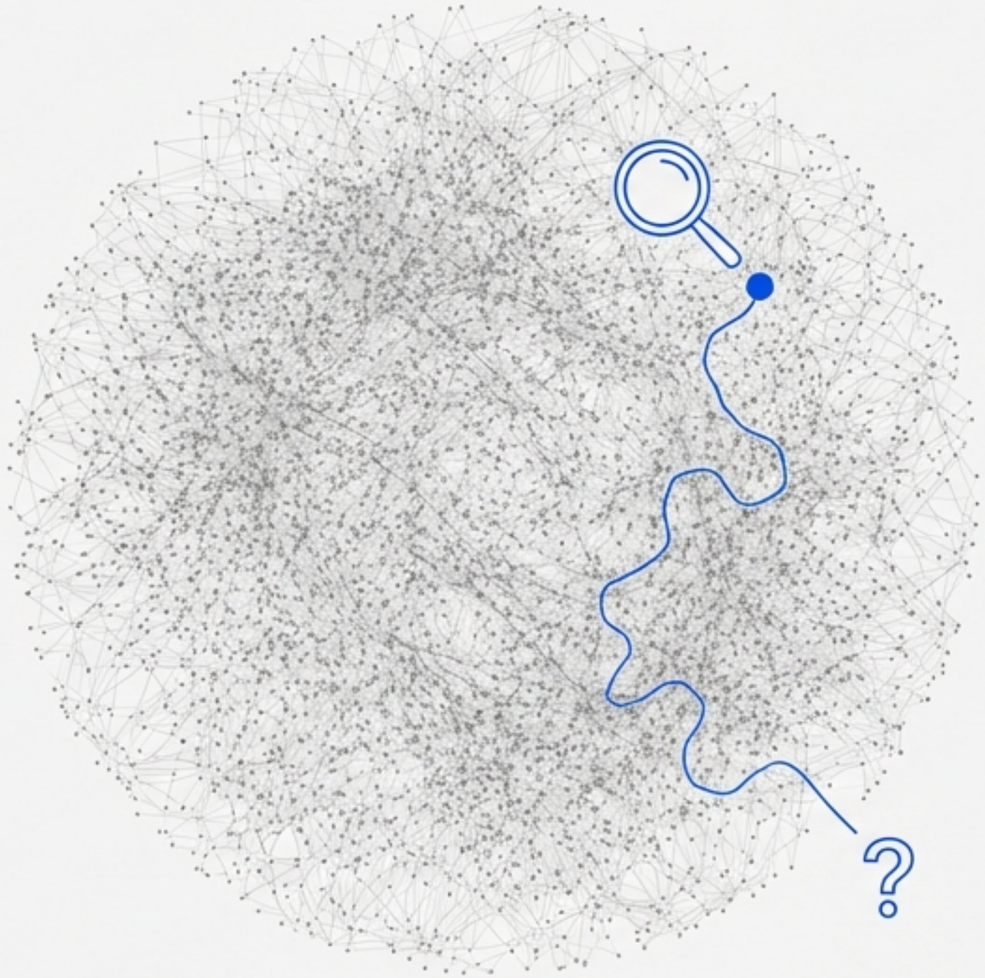
The Address of the Internet

A Deep Dive into URLs, URIs, and URNs

How do you find one specific thing in a network of billions?



The Internet is a massive, global network of computers. To retrieve any website, image, video, or file, your browser needs exact instructions. Without a universal addressing system, finding anything would be impossible.

The Solution: A Universal System for Identification

The solution is the Uniform Resource Identifier (URI). A URI is a standardized string of characters used to identify any resource on the internet. Think of it as the parent category for all web identifiers.

URI (Uniform Resource Identifier): Any string that identifies a resource.

There are two ways to identify something: by its name or by its location.

A URI can identify a resource in two distinct ways. It can give it a unique, permanent name that never changes, or it can provide the exact address of where to find it right now.

What It Is (Name)

A unique identifier, independent of location.
Example: An Aadhaar number or a book's ISBN.

Where It Is (Location)

A specific address that tells you how and where to access it.
Example: A home address.

URI is the concept, URL and URN are the implementations.

URI (Uniform Resource Identifier)

URN (Uniform Resource Name) Name)

`urn:isbn:9780134685991`

Identifies by name only. Not used in browsers.

URL (Uniform Resource Locator)

`https://example.com/profile`

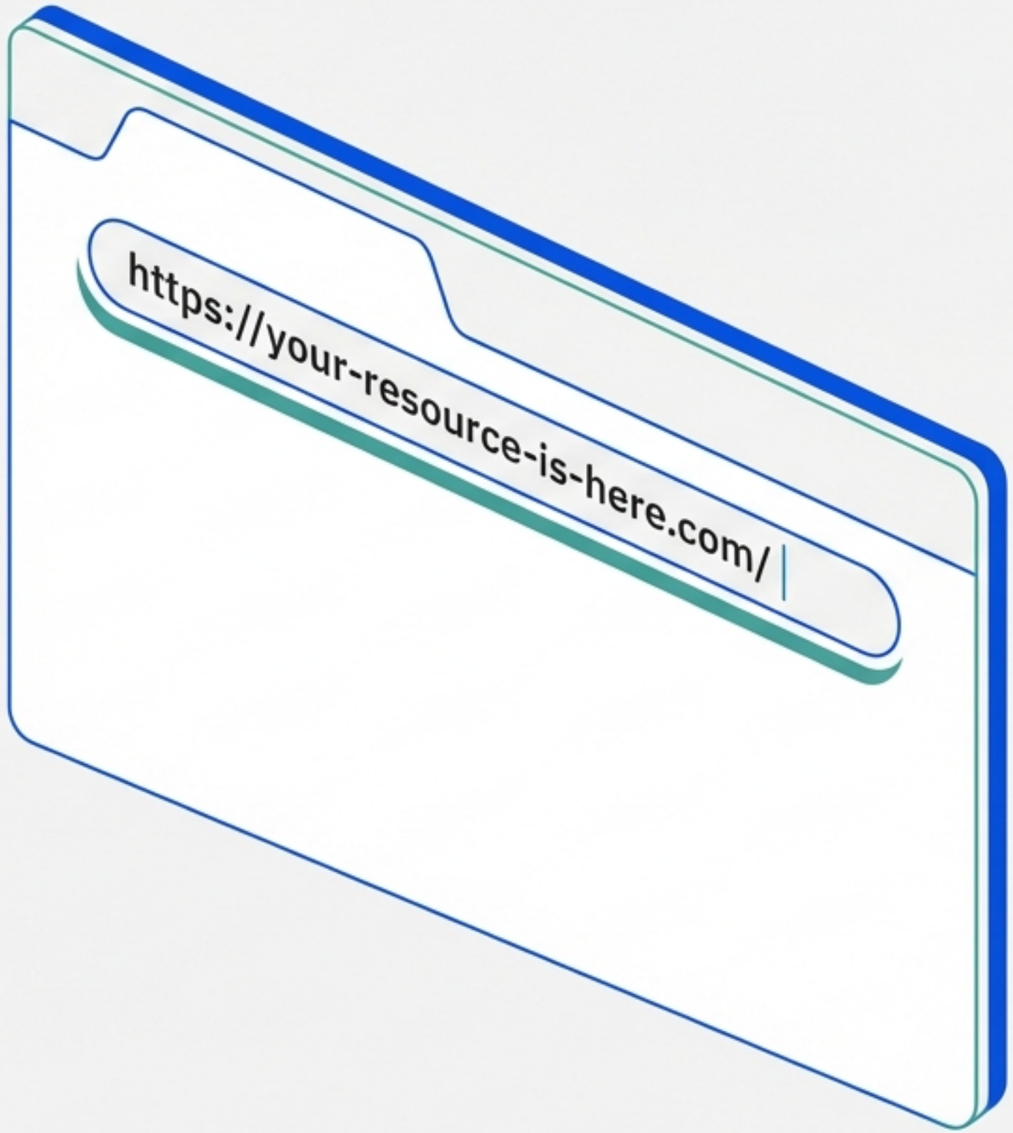
Identifies by location and provides access method. Used by all browsers.

All URLs are URIs, but not all URIs are URLs.

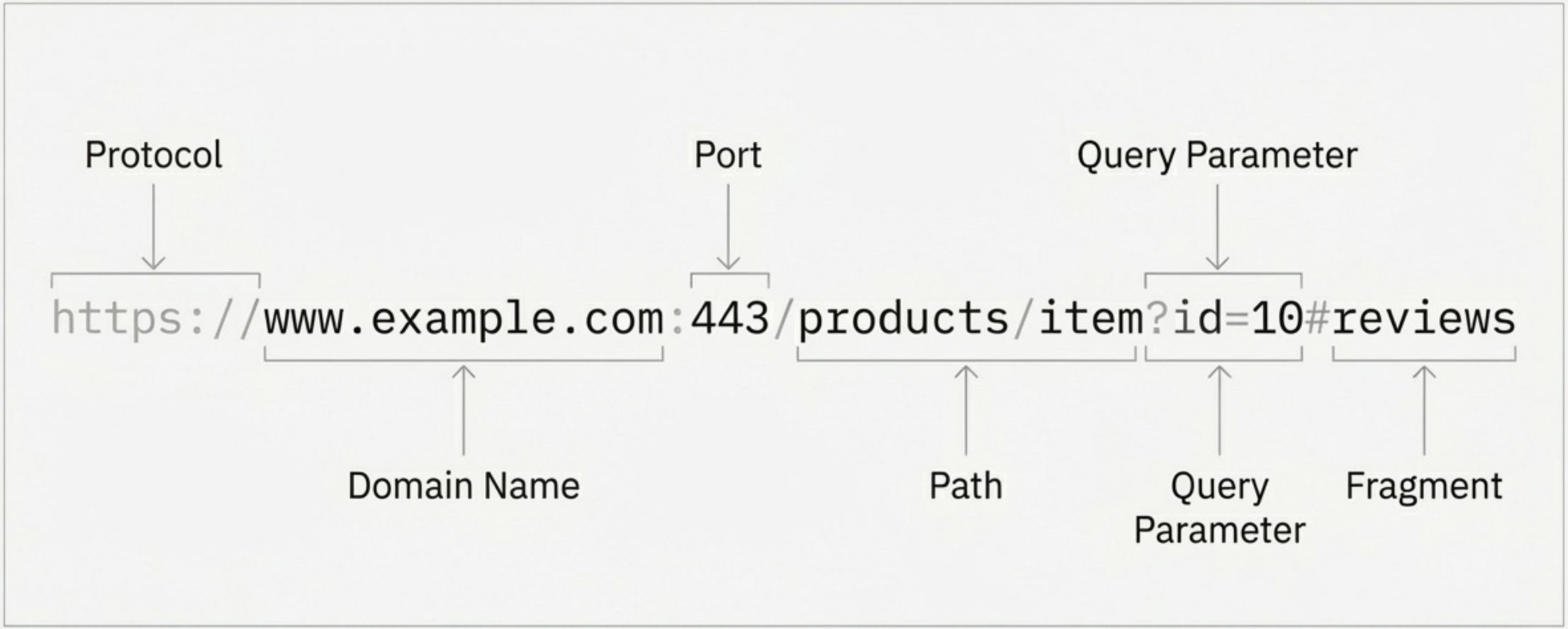
The URL is the address we use every day.

While URNs are a useful concept, the **Uniform Resource Locator (URL)** is what powers the web. It is the complete address of a resource, providing both *where* it is located and *how* to access it.

> Think of sending a letter. To ensure delivery, you need the full address: Country, City, Street, and House Number. A URL works the same way for the Internet.



The Anatomy of a URL: Every piece has a purpose.



Each segment has one responsibility. There is no overlap.

Part 1: The Protocol and Domain

Protocol (`https://)

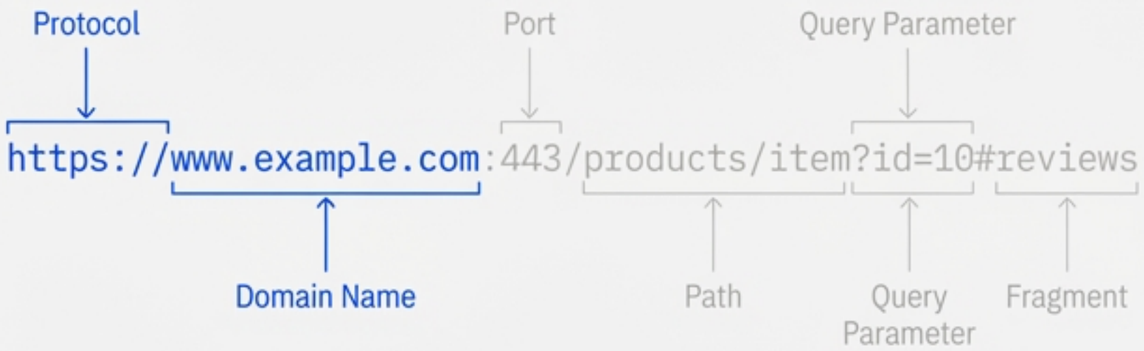
Tells the browser **how to communicate** with the server. It defines the rules of communication.

Common Protocols: `https`, `http`, `ftp`, `mailto`.

Domain Name (www.example.com)

The **human-friendly name** of the server to contact.

Domain names are converted into IP addresses (e.g., `142.250.190.14`) by the **DNS (Domain Name System)**.



Part 2: The Path and Port

Port (`:443`)

A server runs many programs. The port number specifies **which program** to talk to.

Common Defaults: `80` for HTTP, `443` for HTTPS. These are usually hidden because browsers assume them by default.

Path (`/products/item`)

Specifies the **exact resource** you want from the server.

If the domain is the house, the path is the specific room inside the house.

Examples: `/login` (login page), `/images/logo.png` (an image file).

Part 3: Query Parameters for Extra Data

Purpose

To send **extra data** to the server. They always start with a `?`.

Format

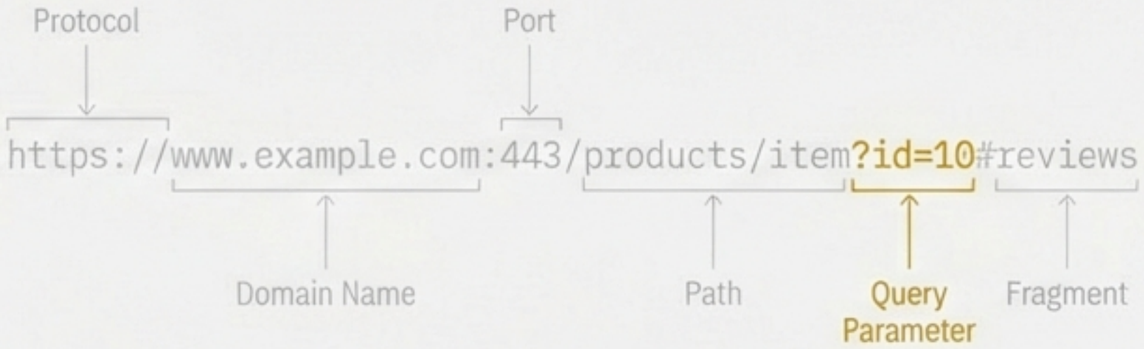
Consists of `key=value` pairs.

Multiple Parameters

Separated by an ampersand (`&`), for example: `?user=123&role=admin`.

Common Uses

Search terms, applying filters, passing user IDs.

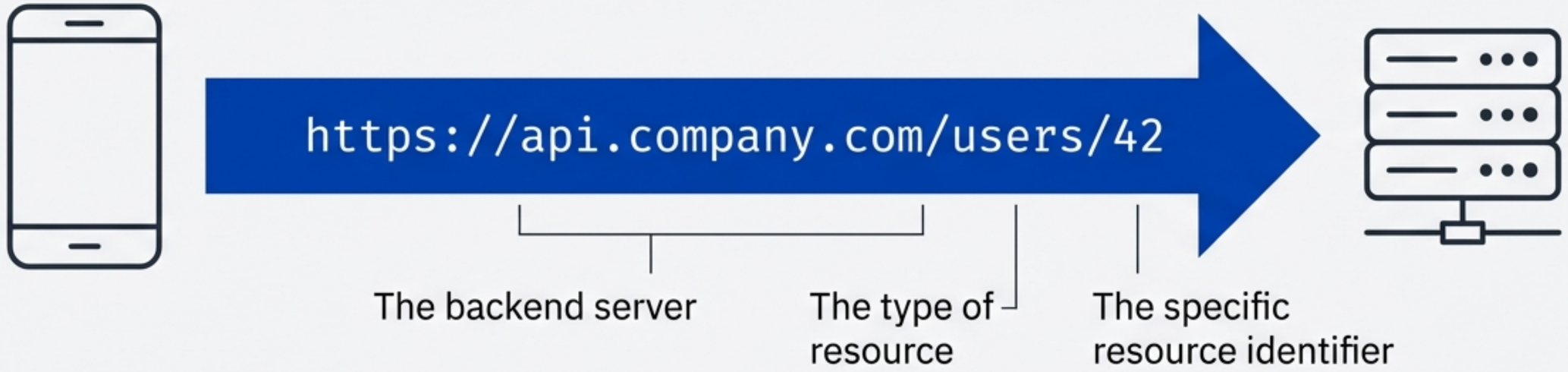


Part 4: The Fragment for Browser Navigation

- **Purpose:** Tells the browser **where to jump to on the page.** It always starts with a #'.
- **Key Detail:** The fragment is **processed only by the browser.** It is never sent to the server.
- **Use Case:** Navigating to a specific section or heading within a long document.

URLs are the language of modern APIs.

In modern applications, URLs are not just for websites. They are fundamental to how software communicates. Backend systems use structured URLs, called **endpoints**, to expose data and functionality.



Connection to Code: This URL typically maps to a specific function in the backend code, such as `getUserById(42)`.

The URL is the Noun, the HTTP Method is the Verb.

This separation is an intentional and foundational principle of robust API design. The URL precisely identifies the resource you want to interact with, while the accompanying HTTP method defines the action you want to perform on that resource.

URL Identifies the Resource

`/users/42`

The user with ID 42.

HTTP Method Defines the Action

Fetch, Create, Update, or Delete.

Putting It All Together: A Full Spectrum of Operations

HTTP Method	URL	Meaning
GET	/users	Fetch a list of all users
GET	/users/42	Fetch the user with ID 42
POST	/users	Create a new user
PUT	/users/42	Update the user with ID 42
DELETE	/users/42	Delete the user with ID 42

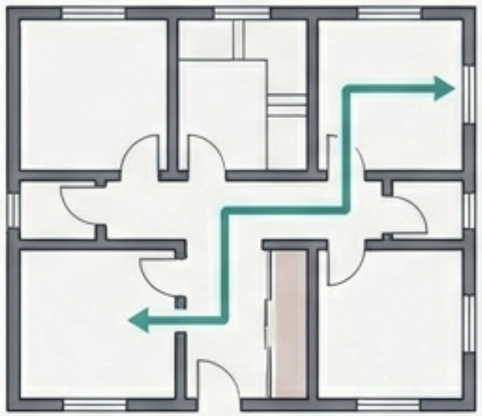
Location, Location, Location: Absolute vs. Relative URLs

Absolute URL

```
https://example.com/images/logo.png
```

A full address that works from anywhere on the internet.

Relative URL



```
/images/logo.png
```

A partial address that depends on the current domain. Shorter and commonly used for links within the same website.

The Hidden Detail: URL Encoding

URLs can only safely contain a specific set of characters. Special characters, like spaces, must be converted into a safe format to avoid breaking the URL structure. This encoding process happens automatically in your browser.

Before

`Search for Hello World`

Browser Encodes

After

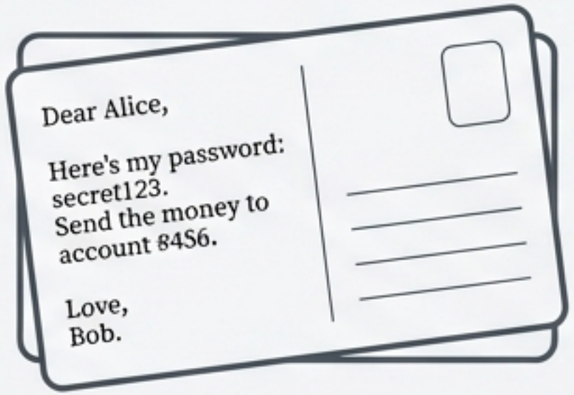
`Search+for+Hello%20World`

Character	Encoded As
Space	%20
@	%40
/	%2F

The Padlock: Why HTTPS is Non-Negotiable

HTTPS is not just ‘http with an s’—it stands for HTTP + Secure. It uses encryption (TLS) to protect the data transferred between your browser and the server. Modern browsers will actively block or warn users about sites that do not use HTTPS.

HTTP



HTTP

- Data is sent in plain text.
- Passwords can be intercepted.
- Content can be modified.

HTTPS

HTTPS

- Data is encrypted.
- Server identity is verified.
- Protects against attackers.

Your Mental Model for the Web's Address System

[URL] + **[Protocol]** + **[Server]** + **[Resource]** + **[Instructions]**

- A **URL (Uniform Resource Locator)** is the address of a resource on the Internet, specifying how to access it and where it is located.
- **URI** is the general concept of an identifier; **URL** and **URN** are the specific types.
- The URL identifies the **noun** (resource); the **HTTP Method** defines the **verb** (action).
- The **Path** specifies the resource location on the server.
- **Query parameters** send additional data *to the server*.
- The **Fragment** is an instruction *for the browser*, not the server.
- **HTTPS** is essential for security and trust.