



**Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»**

Інформаційні системи

Викладач: к.т.н., доц. Саяпіна Інна Олександрівна

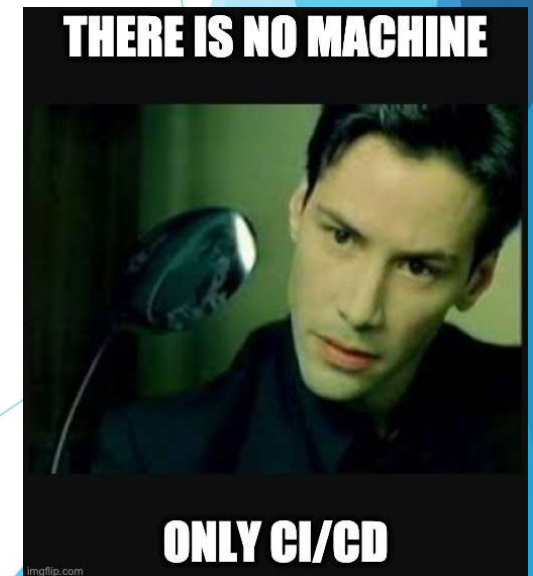
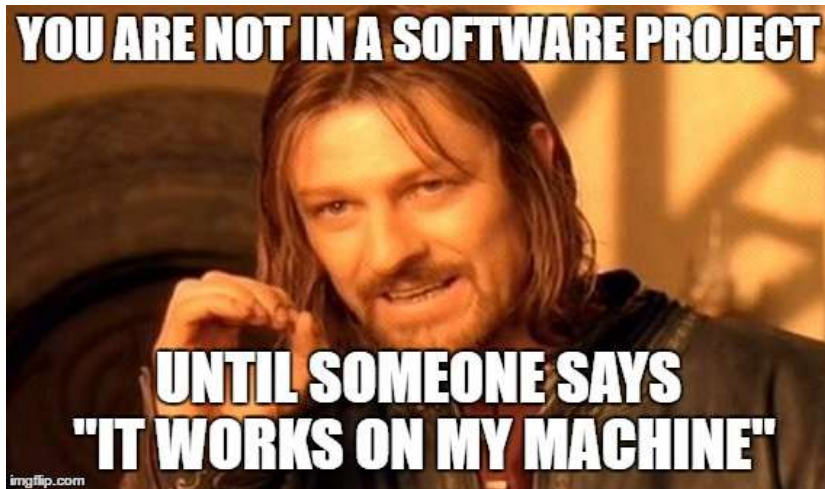
План заняття:

- ▶ Розгортання мікросервісів
 - ▶ Використання контейнерів
 - ▶ Docker
 - ▶ Kubernetes
- ▶ Види додатків



Контейнери

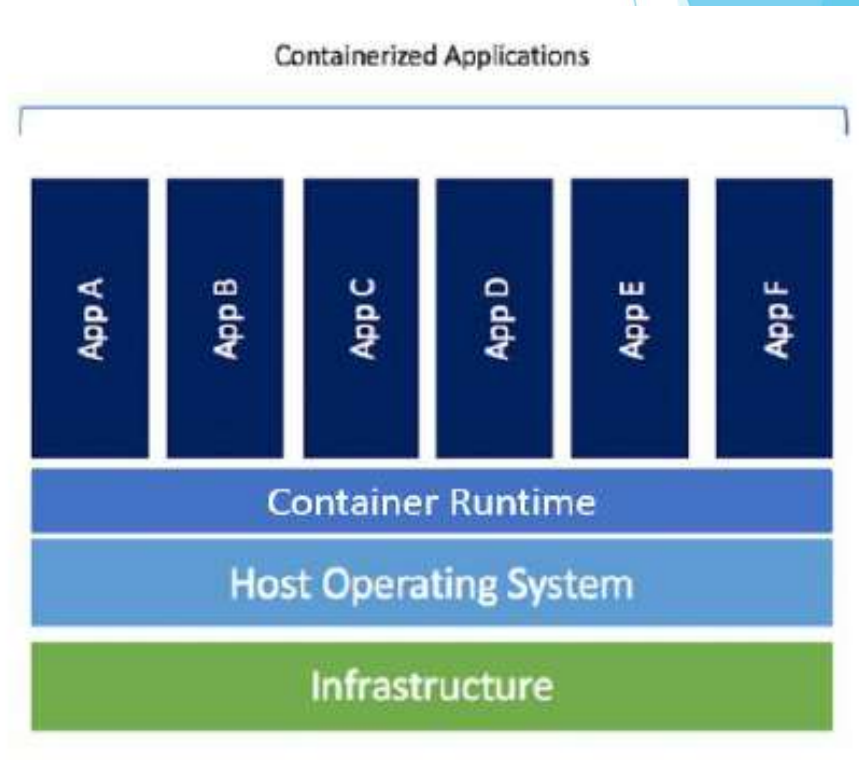
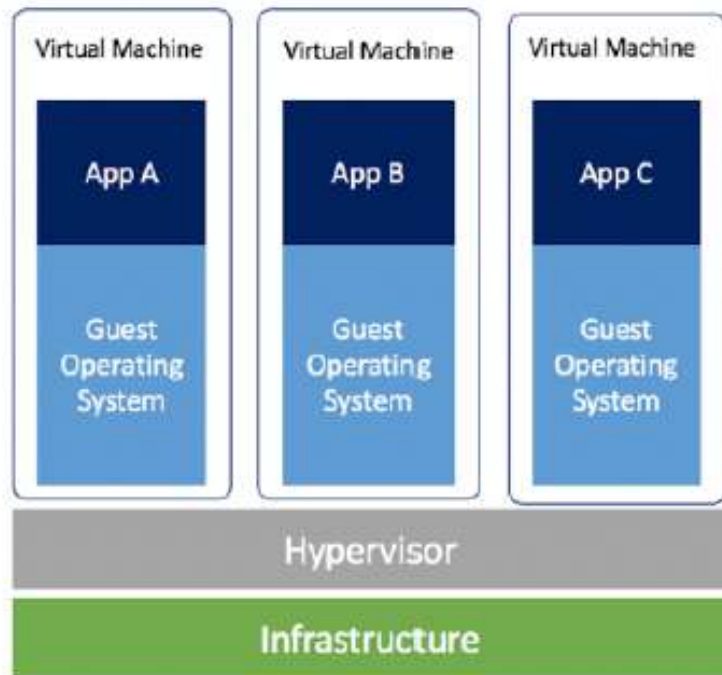
- ▶ Традиційне розгортання:
 - ▶ Код було скопійовано та зібрано на робочому сервері
 - ▶ На серверах виявлено проблеми, яких не було у машині розробників



Контейнери

- ▶ Легкий, автономний і виконуваний програмний пакет, який містить усі необхідні компоненти для запуску програми
- ▶ Пакує програмне забезпечення, його залежності та файли конфігурації
- ▶ Можна копіювати між машинами
- ▶ Використовує базову операційну систему
- ▶ Контейнери ізолювані від базової хост-системи, що дозволяє їм узгоджено працювати в різних середовищах.

Container vs VM



Контейнери vs віртуальні машини

- ▶ Контейнери і віртуальні машини - це обидві форми віртуалізації, які дозволяють працювати декільком ізольованим середовищам на одній фізичній машині.
- ▶ Контейнери є більш легкими та гнучкими, ніж віртуальні машини, оскільки вони використовують те саме ядро операційної системи та лише упаковують код програми та залежності.
- ▶ Віртуальні машини є більш безпечними та сумісними, ніж контейнери, оскільки вони мають власну операційну систему та апаратний рівень абстракції.
- ▶ • Контейнерізація та віртуалізація можуть підвищити ефективність, масштабованість, портативність і надійність програм, але також створюють певні проблеми, такі як складність, накладні витрати на продуктивність і суперечка за ресурси.

Контейнери vs віртуальні машини

- ▶ **Продуктивність:** Контейнери, як правило, швидші та ефективніші, ніж віртуальні машини, оскільки вони мають менше накладних витрат і використовують те саме ядро операційної системи. Віртуальні машини, з іншого боку, мають працювати з повною операційною системою та гіпервізором, що може споживати більше ресурсів і спричиняти погіршення продуктивності. Однак віртуальні машини також можуть запропонувати кращу ізоляцію та безпеку, оскільки вони менш схильні до втручання інших процесів або зловмисних атак.
- ▶ **Сумісність:** Віртуальні машини більш сумісні та портативні, ніж контейнери, оскільки вони можуть запускати будь-яку операційну систему та програму, незалежно від базового апаратного чи програмного забезпечення. Контейнери, з іншого боку, можуть зіткнутися з деякими проблемами сумісності, оскільки вони залежать від операційної системи хоста та середовища виконання контейнера. Наприклад, ви не можете запустити контейнер Windows на хості Linux або контейнер Docker на кластері Kubernetes без додаткової конфігурації чи адаптації.
- ▶ **Масштабованість:** Контейнери більш масштабовані та гнучкі, ніж віртуальні машини, оскільки їх можна легко створювати, знищувати, тиражувати та оркеструвати. Контейнери також більше підходять для мікросервісної архітектури, яка передбачає розбиття програми на менші та незалежні служби, які взаємодіють між собою. Віртуальні машини, з іншого боку, більш жорсткі та складні в управлінні, оскільки вони вимагають більше конфігурації та обслуговування. Віртуальні машини також більше підходять для монолітної архітектури, яка передбачає побудову програми як єдиного інтегрованого блоку.

Переваги використання контейнерів

Predictability

The same package is deployed from the dev machine to the test to production

Performance

Container goes up in seconds vs minutes in VM

Density

One server can run thousands of containers vs dozens of VMs

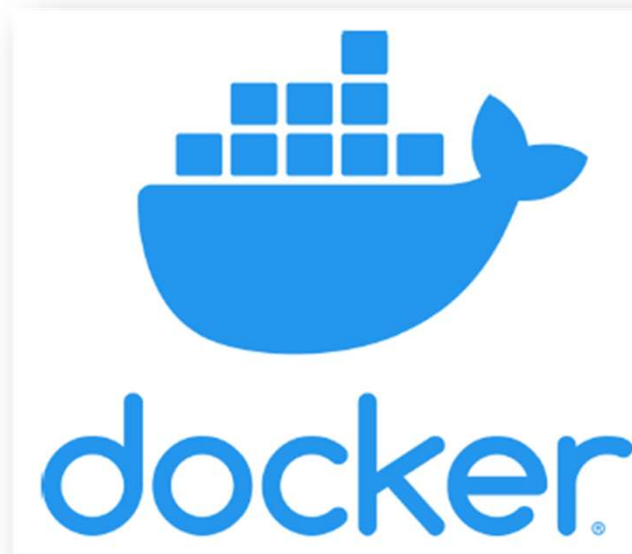
Недоліки використання контейнерів

Isolation

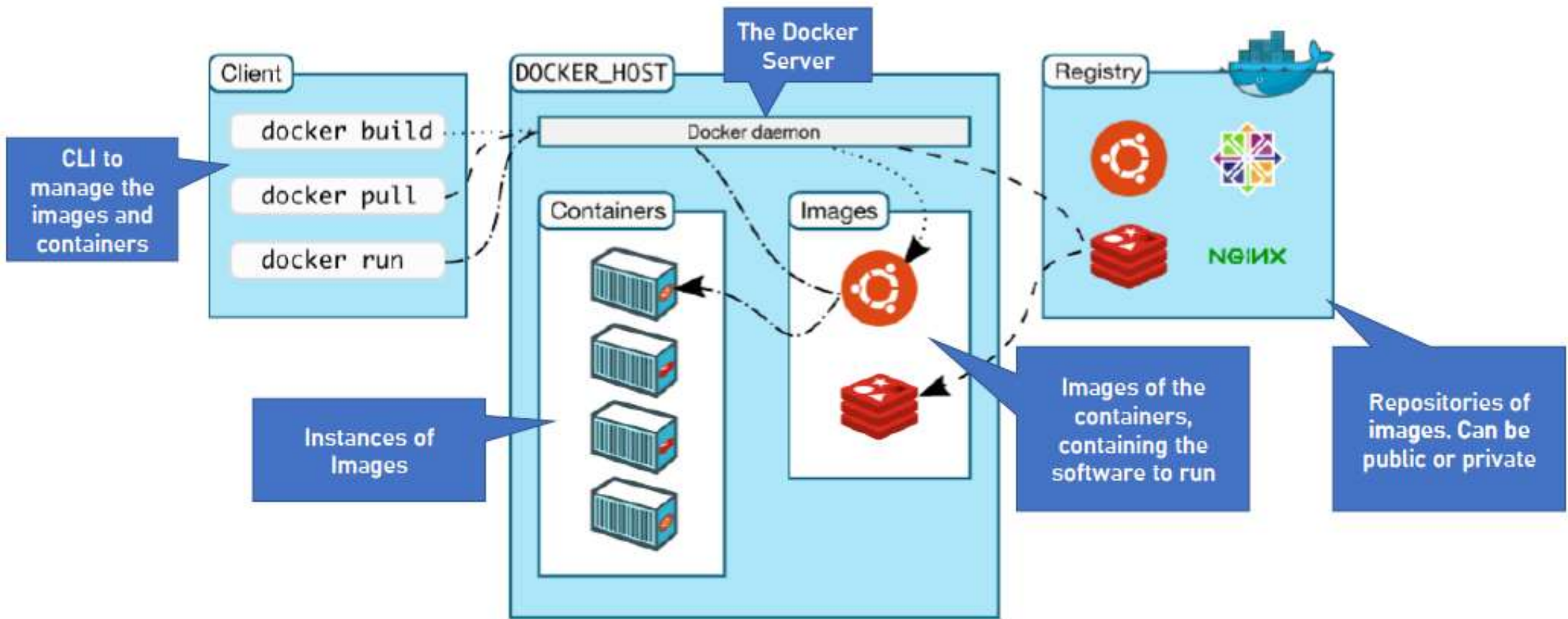
Containers share the same OS, so isolation is lighter than VM

Docker

- ▶ Найпопулярніше контейнерне середовище
- ▶ Фактичний стандарт для контейнерів
- ▶ Випущений в 2013 році



Docker



<https://docs.docker.com/get-started/overview/>

dockerfile

- ▶ Містить інструкції зі створення власних образів

```
1 WORKDIR /opt/node_app
2 COPY package.json package-lock.json* ./
3 RUN npm install --no-optional && npm cache clean --force
4 ENV PATH /opt/node_app/node_modules/.bin:$PATH
5 WORKDIR /opt/node_app/app
6 COPY . .
```

<https://www.docker.com/blog/keep nodejs rockin in docker/>

Підтримка докера

- ▶ Підтримується всіма основними операційними системами (Windows, Linux, OSX)
- ▶ Підтримується основними хмарними провайдерами



amazon
ECR



Azure
ACR



Google Container
Registry

Управління контейнерами

- ▶ Контейнери — чудовий механізм розгортання
- ▶ Здобув популярність
- ▶ Що станеться, коли їх забагато?



Управління контейнерами



Frontend



Backend



Database

- Розгортання
- Масштабованість
- Моніторинг
- Маршрутизація
- Доступність



Batch
Processes

Kubernetes

- ▶ Найпопулярніша платформа управління контейнерами
- ▶ Де-факто стандарт управління контейнерами
- ▶ Випущено Google у 2014 році



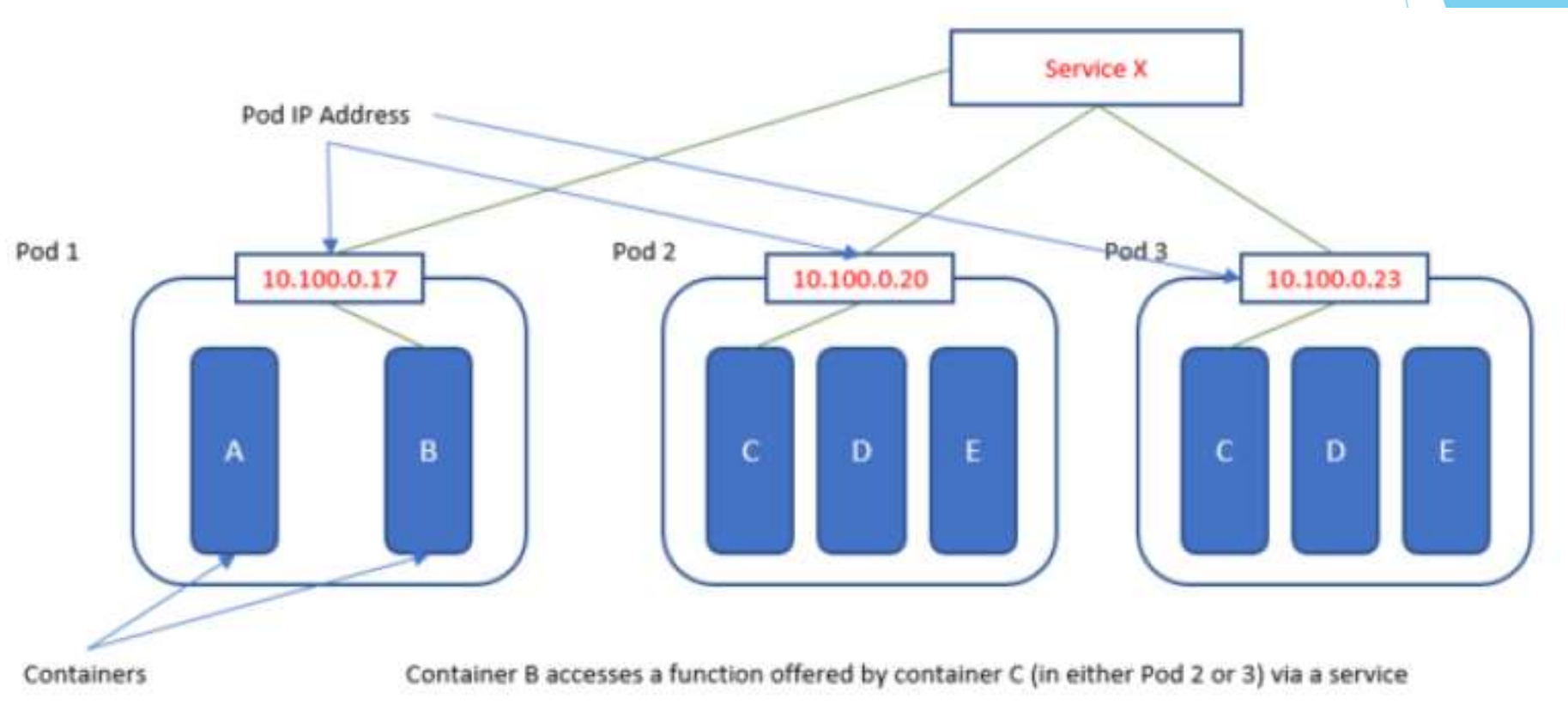
kubernetes

Kubernetes

- ▶ Забезпечує всі аспекти управління:
 - ▶ Маршрутизацію
 - ▶ Масштабування
 - ▶ Високу доступність
 - ▶ Автоматичне розгортання
 - ▶ Управління конфігурацією
 - ▶ І більше...



Архітектура Kubernetes



<https://en.wikipedia.org/wiki/Kubernetes>

Kubernetes in 6 minutes



<https://youtu.be/TLHvYWVUZyc?si=i5V6AfBV9uO-BB0k>

Висновки з розгортання мікросервісів

- ▶ Автоматизоване розгортання є обов'язковим для ефективної архітектури мікросервісів
- ▶ Docker і Kubernetes є де-факто галузевими стандартами
- ▶ Архітектор не несе відповідальності за розгортання, але повинні його розуміти

Основні типи додатків

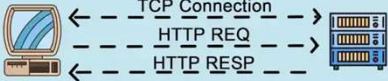




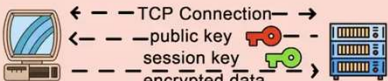




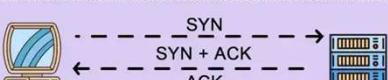


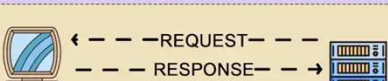



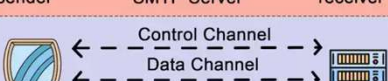

- ▶ Web-додатки
- ▶ Web API
- ▶ Мобільні додатки
- ▶ Консольні додатки
- ▶ Сервіс
- ▶ Desktop додатки



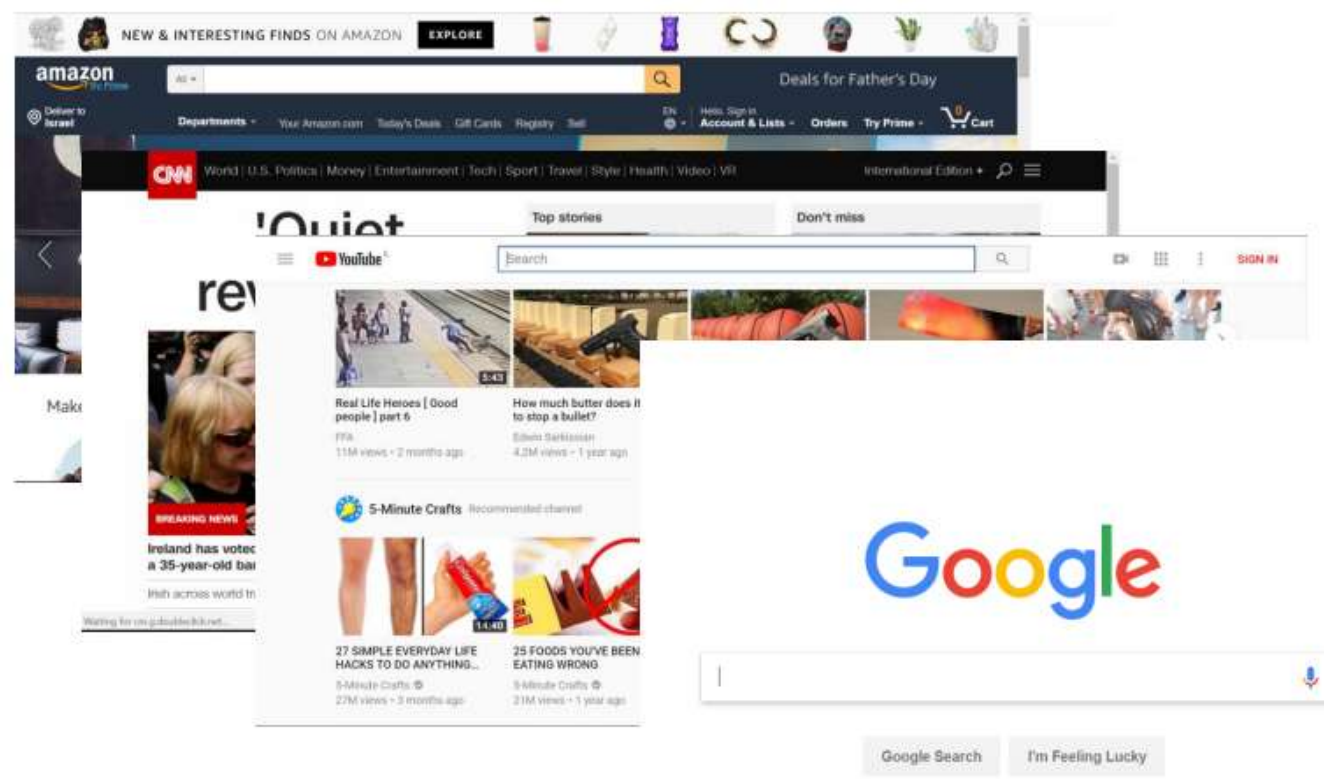
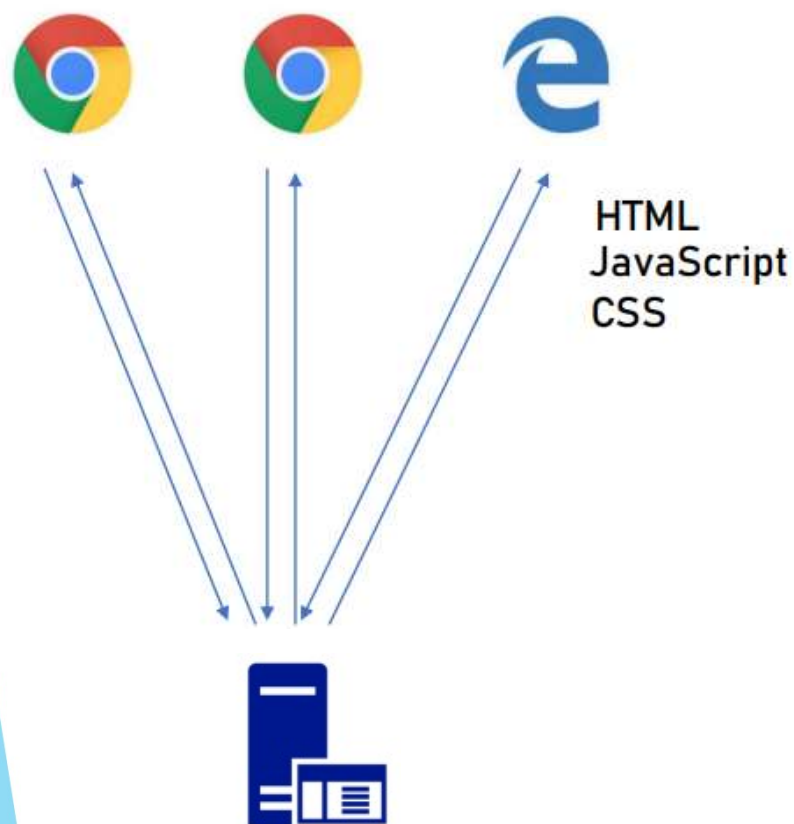
Мережеві протоколи

8 Popular Network Protocols

blog.bytebytego.com

Protocol	How does It Work?	Use Cases
HTTP	 <p>TCP Connection HTTP REQ HTTP RESP</p>	 Web Browsing
HTTP/3 (QUIC)	 <p>UDP Connection 1 2 3 4</p>	 IoT  Virtual Reality
HTTPS	 <p>TCP Connection public key session key encrypted data</p>	 Web Browsing
WebSocket	 <p>HTTP Upgrade Full Duplex</p>	 Live Chat  Real-Time Data Transmission
TCP	 <p>SYN SYN + ACK ACK</p>	 Web Browsing  Email Protocols
UDP	 <p>REQUEST RESPONSE</p>	 Video Conferencing
SMTP	 <p>sender SMTP Server receiver</p>	 Sending/Receiving Emails
FTP	 <p>Control Channel Data Channel</p>	 Upload/Download Files

Web додатки



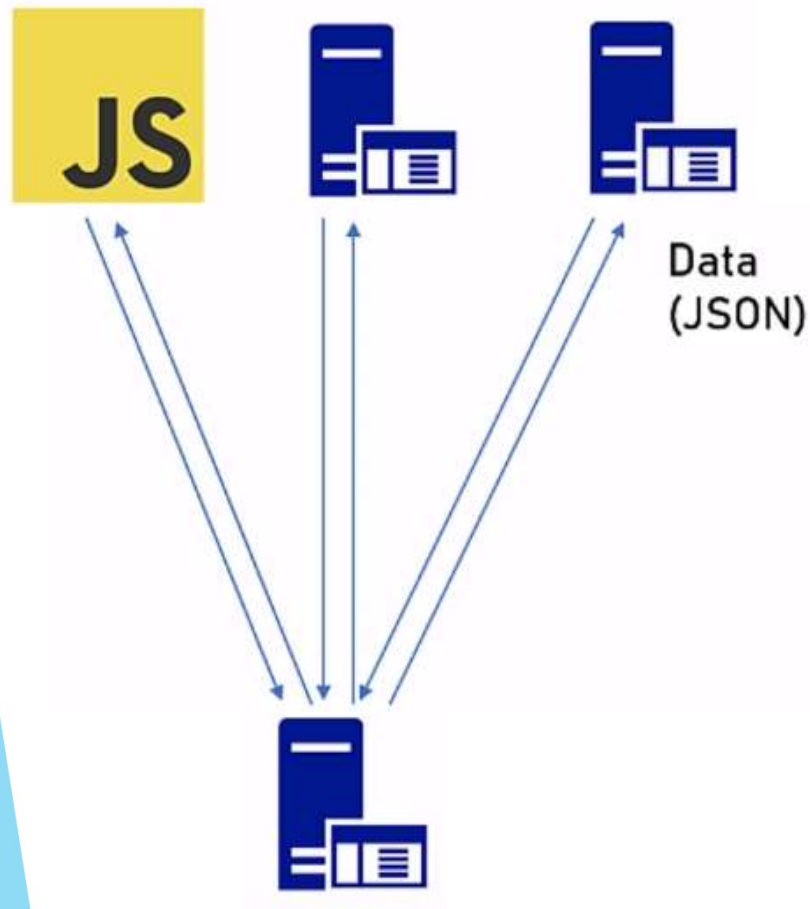
Web додатки

Найкраще підходить для систем, які потребують:

- ▶ Інтерфейс користувача
- ▶ Дії, ініційовані користувачем
- ▶ Короткі, цілеспрямовані операції
- ▶ Основані на запиті-відповіді



Web API



facebook for developers

Docs Tools Support Search developers.facebook.com

Bas
Acc
Two

Graph API
Overview
Using th

Google Search for services, methods, and recent requests... Loading

Microsoft Developer Network

Dev Center Explore Get Started Code Samples Resources Documentation

Last Updated: 5/25/2018

Office 365 API reference

Applies to: Exchange Online | Office 365 | OneDrive for Business

Office 365 API functionality is also available through the **Microsoft Graph**, a unified API that includes APIs from other Microsoft services such as Outlook, OneDrive, OneNote, Planner, and Office Graph, accessible through a single endpoint and with a single access token. We recommend using the Microsoft Graph in your apps when possible.

The Office 365 APIs enable you to provide access to your customer's Office 365 data, including the things they care about most--their mail, calendars, contacts, users and groups, files, and folders--all right from within your app itself.

You can access the Office 365 APIs from solutions across all mobile, web, and desktop platforms. No matter your development platform or tools. So whether you're building web applications using .NET, PHP, Java, Python, or Ruby on Rails, or creating apps for Windows Universal Apps, iOS, Android, or on another device platform, it's your choice.

Table of contents

- Introduction
- Authenticating apps
- Accessing and syncing data
- Integrating non-Microsoft file types
- Making your app easy to use and find
- OneNote
- Office 365 REST API reference
 - Office 365 APIs
 - Use the Outlook REST API
 - Batch Outlook REST requests
 - Outlook Mail

Web API

- ▶ Зазвичай, оснований на Стандарті REST-архітектури
- ▶ Є комбінацією:
 - ▶ URL (<https://www.mysite.com/api/orders>)
 - ▶ Параметрів (`date=10/10/2017`)
 - ▶ HTTP дієслово (GET, POST, DELETE, PUT)
- ▶ Приклади:
 - ▶ GET <https://www.mysite.com/api/users/17>
 - ▶ DELETE <https://www.mysite.com/api/orders/156>
- ▶ Повертає дані, а не HTML
- ▶ Дуже доступний