Web Apps

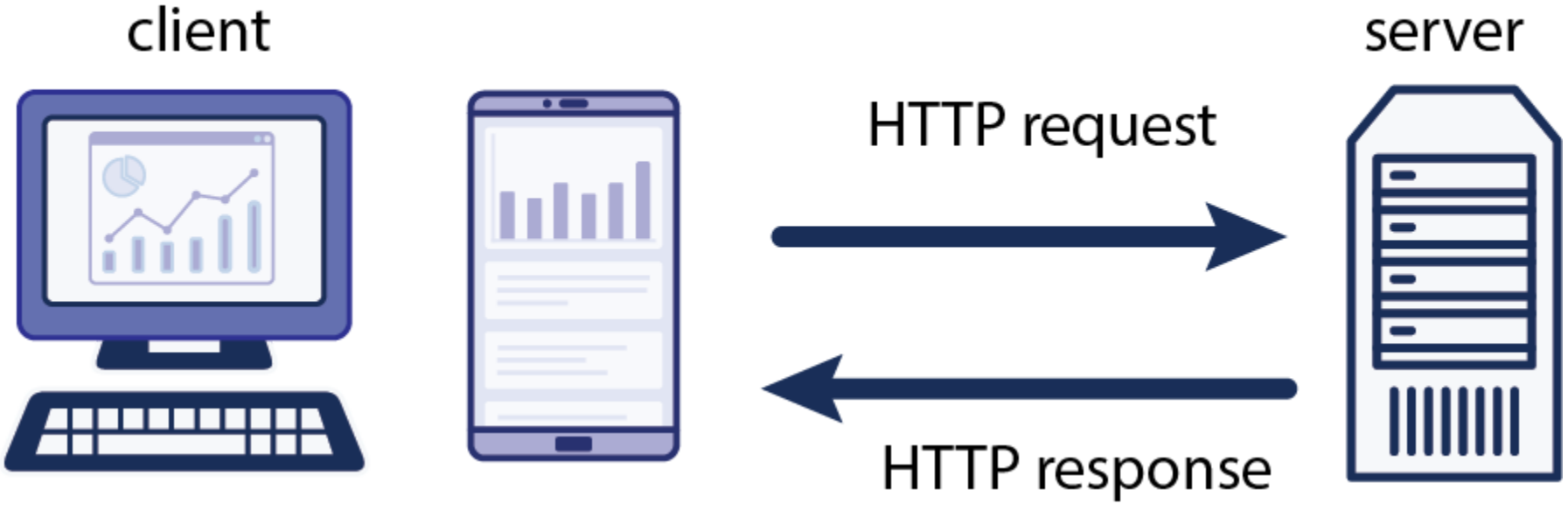
tinyurl.com/61a-webapps

We'll talk about...

- How the web works
- Python for web apps
- Hosting web apps

How the web works

Clients and servers



HTTP

A client sends an HTTP request:

```
GET /index.html HTTP/1.1
Host: www.example.com
```

The server sends back an HTTP response:

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Length: 208
<!DOCTYPE html>
    <html>
    <head>
        <title>Example Domain</title>
    </head>
    <body>
    <h1>Example Domain</h1>
    <p>This domain is to be used for illustrative examples in documents.</p
```

```
    </body>
</html>
```

Webpages

Webpages are made up of three languages:

- **HTML**: Contains the content and uses tags to break it into semantic chunks (headings, paragraphs, etc)
- **CSS**: Contains style rules that apply properties to elements on a page.
- **JavaScript**: Contains code that dynamically accesses and updates the page content to make it more interactive.

What does a server do?

The most basic server just serves up HTML and multimedia files from a file system.

Server-side code is also useful for anything that requires access to persistent data or needs an additional layer of security than allowed in the client.

- User authentication
- Database fetches/updates
- Caching

Server-side Python

Simple HTTP server

The http module in the Python standard library can run a basic server.

It is **not** recommended for production.

It's handy for learning and local development, however...

Example: Simple file server

A file server serves up files and folders according to their path in the file system. Also known as a static server.

Run a file server from any folder:

```
python3 -m http.server 8080
```

Example: Simple dynamic server

 Repo: github.com/pamelafox/python-simple-server-example/

The server code is in server.py. Uses the `http` module to dynamically generate responses.

Run the server:

```
python3 server.py
```

Flask framework

Flask, an external package, is a lightweight framework for server requests and responses.

Apps written in Flask:

- cs61a.org
- Khan Academy (originally)
- Reddit
- Netflix

Example: Simple Flask website

👀 Demo: tinyurl.com/simple-flask-website

 Repo: github.com/pamelafox/simple-flask-server-example/

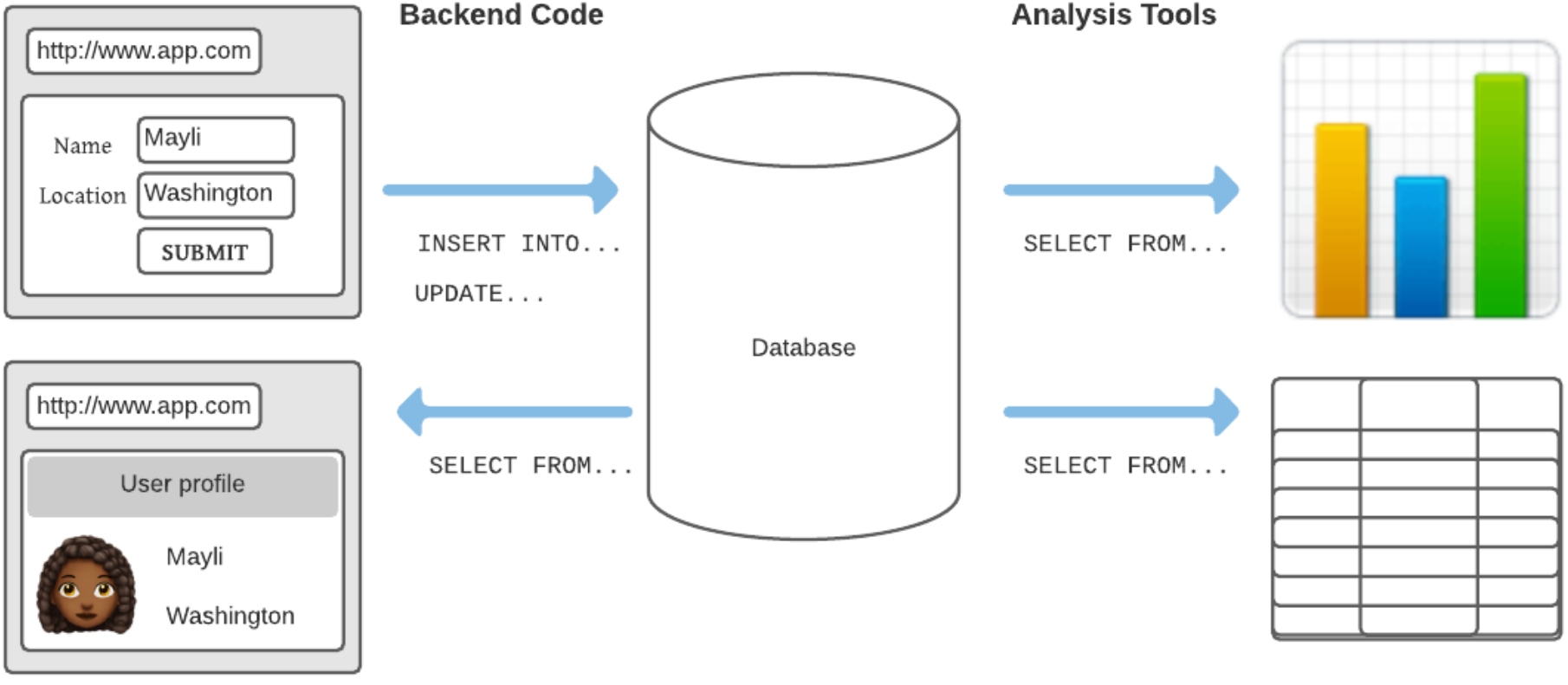
Most of the server code is in app.py. Uses Flask to generate responses for each route

Run the server:

```
python3 app.py
```

Webapps with Databases

Web apps store any data in databases that needs to be shared across multiple users/computers.



Example: Flask Quiz

👀 Demo: tinyurl.com/python-flask-quiz

 Repo: github.com/pamelafox/flask-db-quiz-example/

Most of the server code is in app.py. Uses SQLAlchemy for database interaction.

Run the server:

```
python3 app.py
```

Django framework

Django, an external library, is a fairly "opinionated" framework for server-side code. Includes an ORM for database interaction.

Apps written in Django:

- Coursera (originally, now Scala+Play)
- Instagram
- Pinterest (originally, now Flask)
- Eventbrite

Example: Django Quiz

👀 Demo: tinyurl.com/python-django-quiz

 Repo: github.com/pamelafox/django-quiz-app/

Important server files:
models.py , urls.py , views.py , admin.py

Run DB migrations and server:

```
python manage.py migrate
python manage.py runserver
```

Hosting web apps

Hosting options

When your website is **hosted** on a server, it means other users on the Internet can access it.

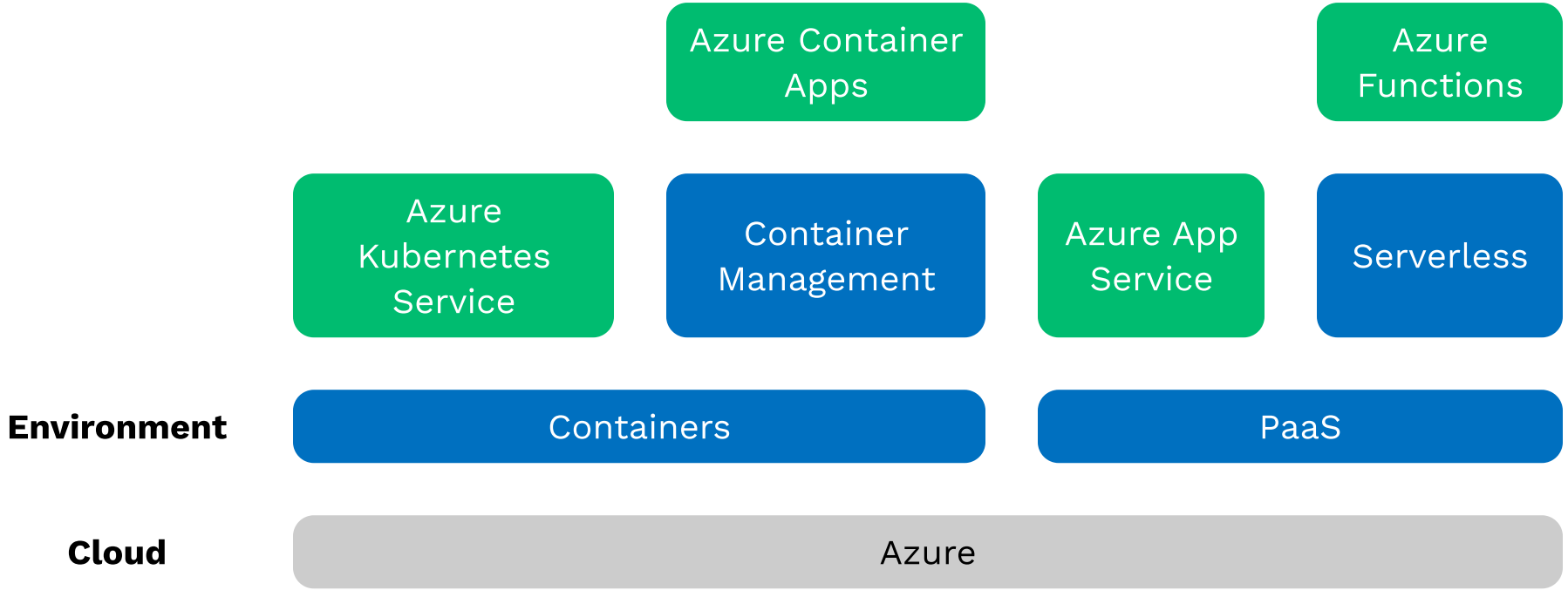
Many possible hosts:

- A Raspberry Pi in your house
- A rented computer in a data center
- A virtual machine
- A PaaS (platform as a service)

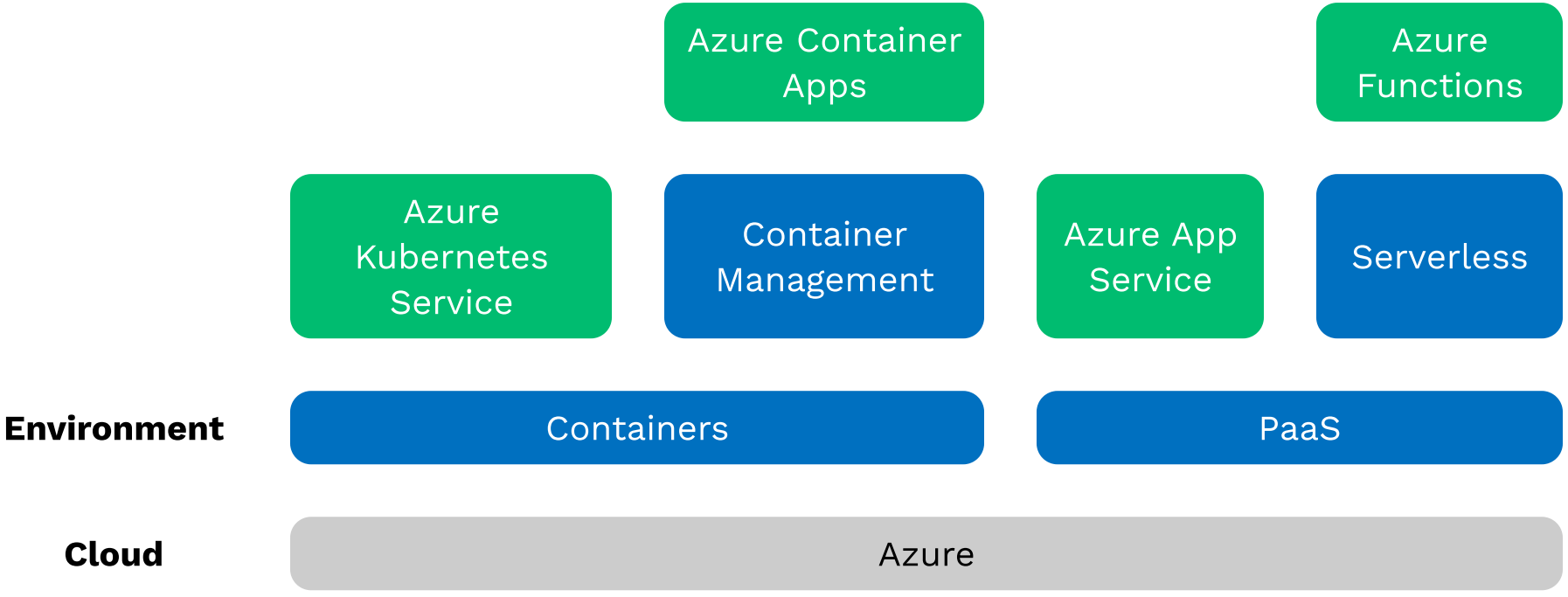
Consider:

- How much control do you want?
- How much do you enjoy administering systems?
- Do you need it to scale up/out?

Azure hosting options



Azure hosting options



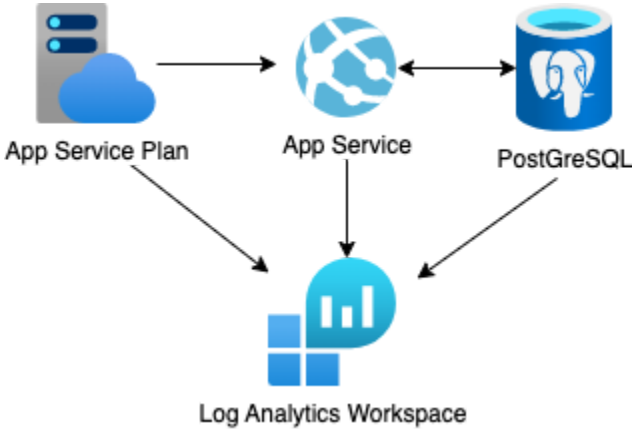
For Flask/Django, App Service is easiest way to get started.

But wait, there's more!

Databases	PostGreSQL, MySQL, CosmosDB, ...
Storage	Blob Storage, Files, Archive Storage, ...
Networking	DNS Zone, Virtual Network, VPN Gateway, ...
Caching	CDN, Front Door, ...
Security	Key Vault, Security Center, ...
Machine Learning	Translator, Bot Service, Computer Vision, ...

...and more!

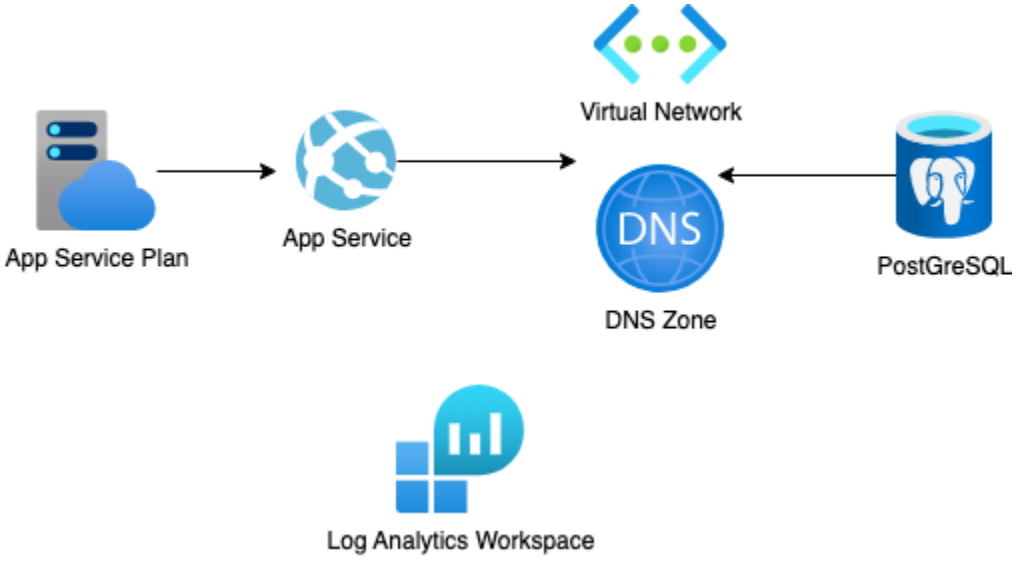
Hosting the Flask quiz on Azure



Using the Azure Dev CLI:

```
azd up
```

Hosting the Django quiz on Azure



Using the Azure Dev CLI:

```
azd up
```

More Azure resources

- Deploy a Python (Django or Flask) web app to Azure App Service
- Deploy a Python (Django or Flask) web app with PostgreSQL in Azure
- Hosting Python apps on Azure
- Getting Started with Python in VS Code
- MS Python Discord: aka.ms/python-discord

Any questions?