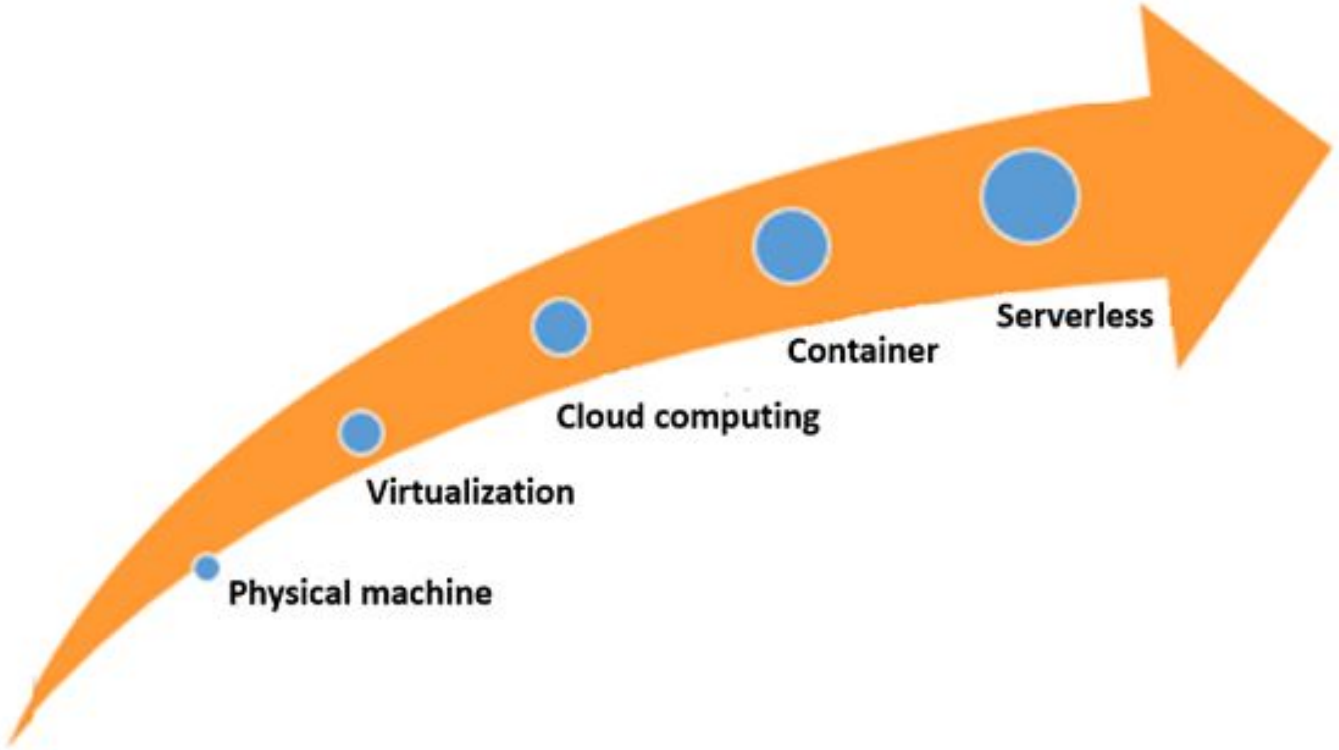
Serverless Applications

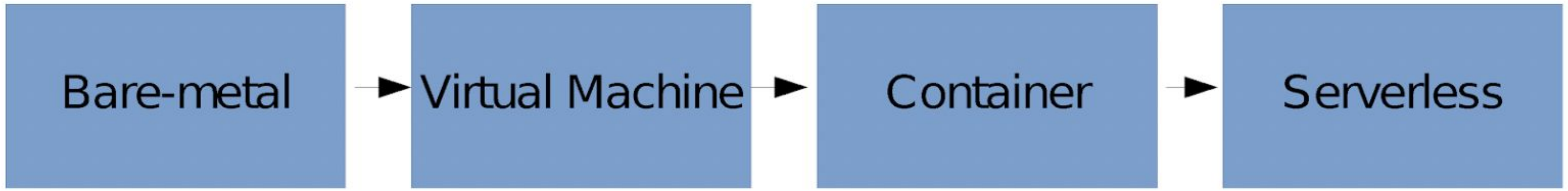
What is serverless?

- Ability to deploy without thinking about servers
- Don’t pay for instances
- Just pay per execution
  - How many times was my code run?

History


Physical machine
Virtualization
Cloud computing
Container
Serverless

Levels of Abstraction



Serverless

- Instead of deployments, you write functions
- Functions are the units – the lowest-level building block
- Typical workflow:
  - Create function (defining)
  - Write + Upload Code
  - Run function
    - Triggered by API call or other event

Pros of Serverless

- No servers to manage
- No thinking about patches and security
- Autoscaling seamless – and not developers' problems
- Pay per invocation
  - When no traffic, no charges
  - Simple pricing
- Service integrations
  - Very easy to integrate with other services within cloud provider
- Easy to debug and use

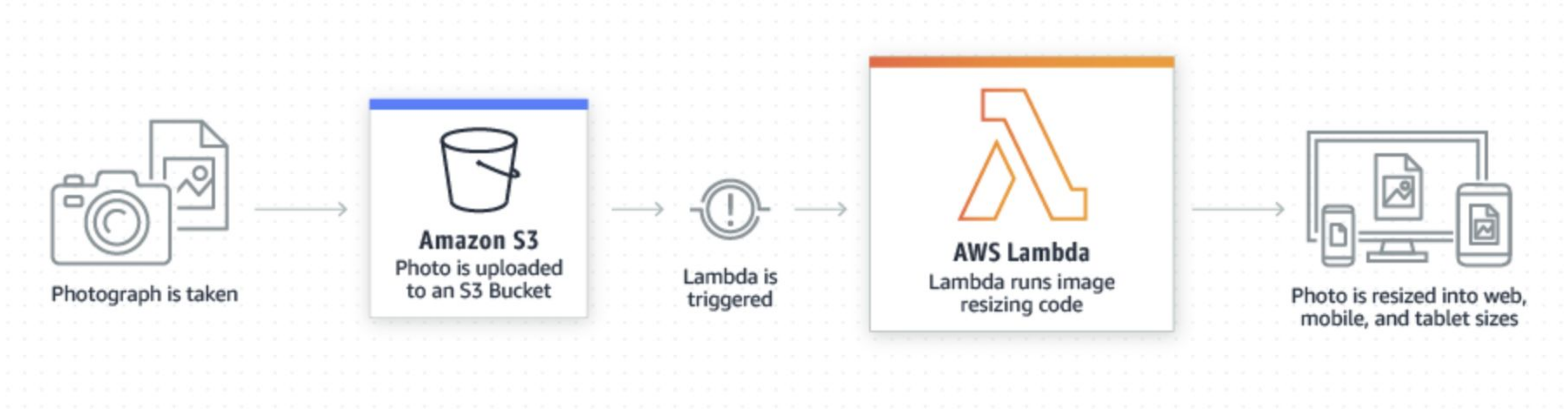
Cons of Serverless

- Gain flexibility and ease of use **at the expense of control**
- Do not have visibility into infra
- Depending upon cloud provider to do things for you
  - Large companies will be partly serverless, partly not

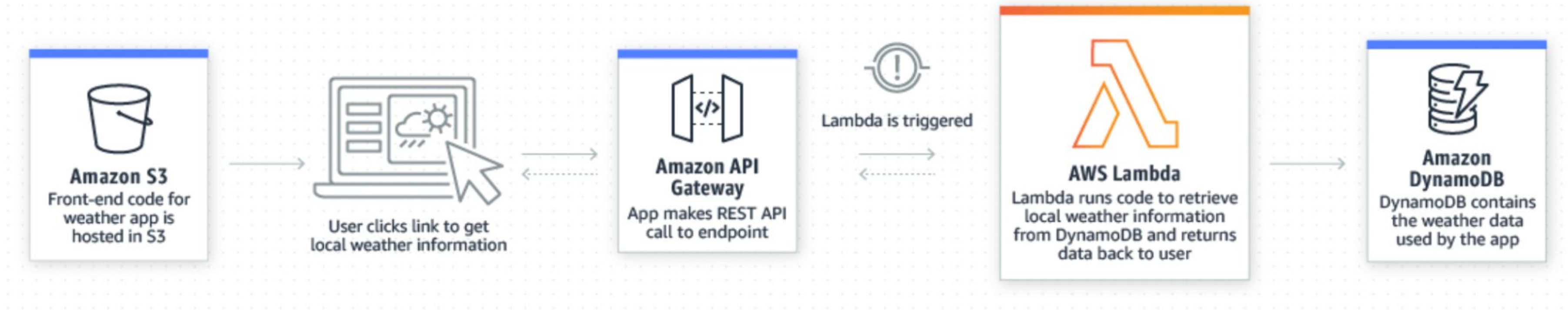
Cloud Providers: Serverless

- https://cloud.google.com/serverless
- https://aws.amazon.com/lambda/
- https://azure.microsoft.com/en-us/solutions/serverless/
- https://www.digitalocean.com/blog/introducing-digitalocean-functions-serverless-c
omputing
- Others following suit

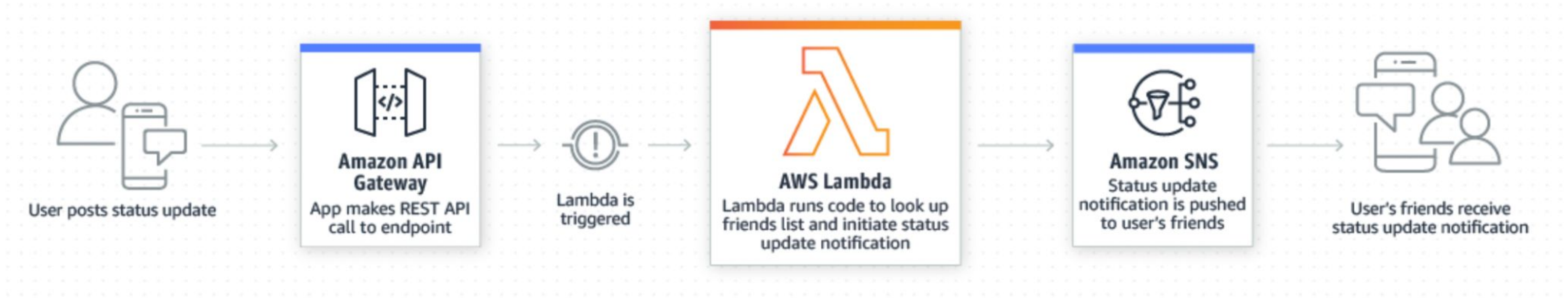
AWS Lambda Examples


Photograph is taken
Amazon S3
Photo is uploaded to an S3 Bucket
Lambda is triggered
AWS Lambda
Lambda runs image resizing code
Photo is resized into web, mobile, and tablet sizes

AWS Lambda Examples


Amazon S3
Front-end code for weather app is hosted in S3
User clicks link to get local weather information
Amazon API Gateway
App makes REST API call to endpoint
Lambda is triggered
AWS Lambda
Lambda runs code to retrieve local weather information from DynamoDB and returns data back to user
Amazon DynamoDB
DynamoDB contains the weather data used by the app

AWS Lambda Examples


User posts status update
Amazon API Gateway
App makes REST API call to endpoint
Lambda is triggered
AWS Lambda
Lambda runs code to look up friends list and initiate status update notification
Amazon SNS
Status update notification is pushed to user's friends
User's friends receive status update notification

Django: MTV

- The Model-View-Template (MVT) is slightly different from MVC.
- The main difference between the two patterns is that Django itself takes care of the Controller part (Code that controls the interactions between the Model and View)
- In Django-land, a “view” is a Python callback function for a particular URL, because that callback function describes which data is presented.
- Furthermore, it’s sensible to separate content from presentation – which is where templates come in.
- In Django, a “view” describes which data is presented
- a view normally delegates to a template, which describes **how** the data is presented.
- Where does the “controller” fit in, then? In Django’s case, it’s probably the framework itself: the machinery that sends a request to the appropriate view, according to the Django URL configuration.

In-Class Exercise

- https://aws.amazon.com/getting-started/hands-on/build-serverless-web-app-lambda-apigateway-s3-dynamodb-cognito/