SERVERLESS

Doug Lampe

Orlando Code Camp 2018

AGENDA

- Evolution of Application Architecture
- Traditional Architectures
- Containerized Architectures
- Serverless Architectures
- AWS “Serverless” Services
- Sample Application
- Takeaways

Evolution of Application Architecture

Single Procedural Program

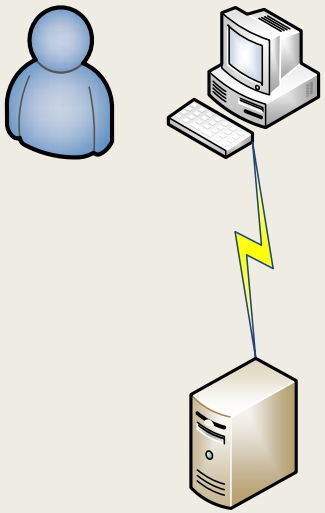
```
 look
Kitchen
You are in the kitchen of the white house. A table seems to have been
used recently for the preparation of food. A passage leads to the west
and a dark staircase can be seen leading upward. A dark chimney leads
down and to the east is a small window which is open.

>climb stairs
You have moved into a dark place.
It is pitch black. You are likely to be eaten by a grue.

>
```

Device with Software and Cloud API

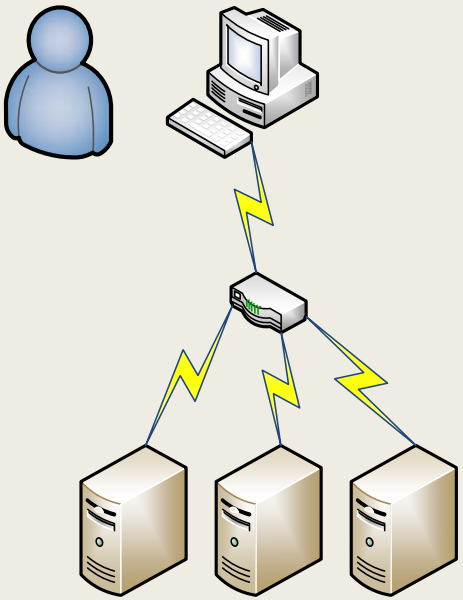
Just Someone Else's Computer

Traditional Architectures

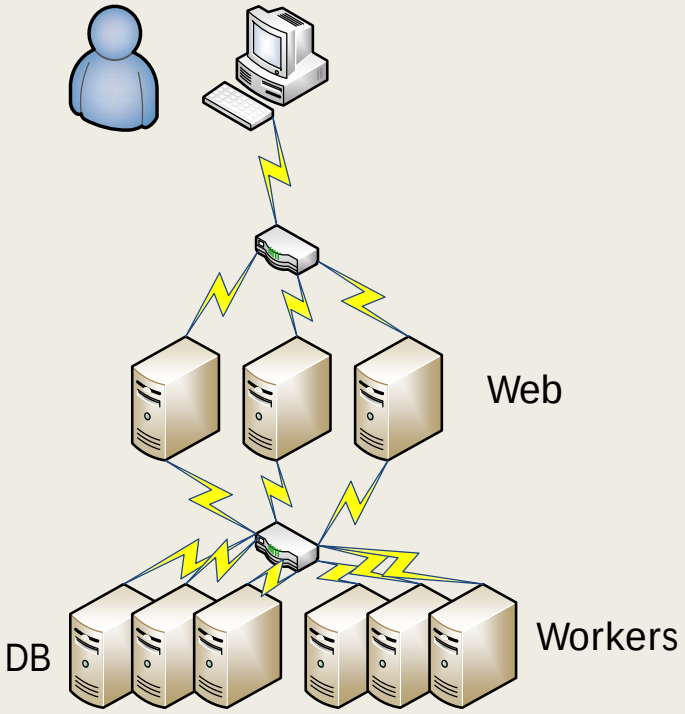
Single Dedicated Server



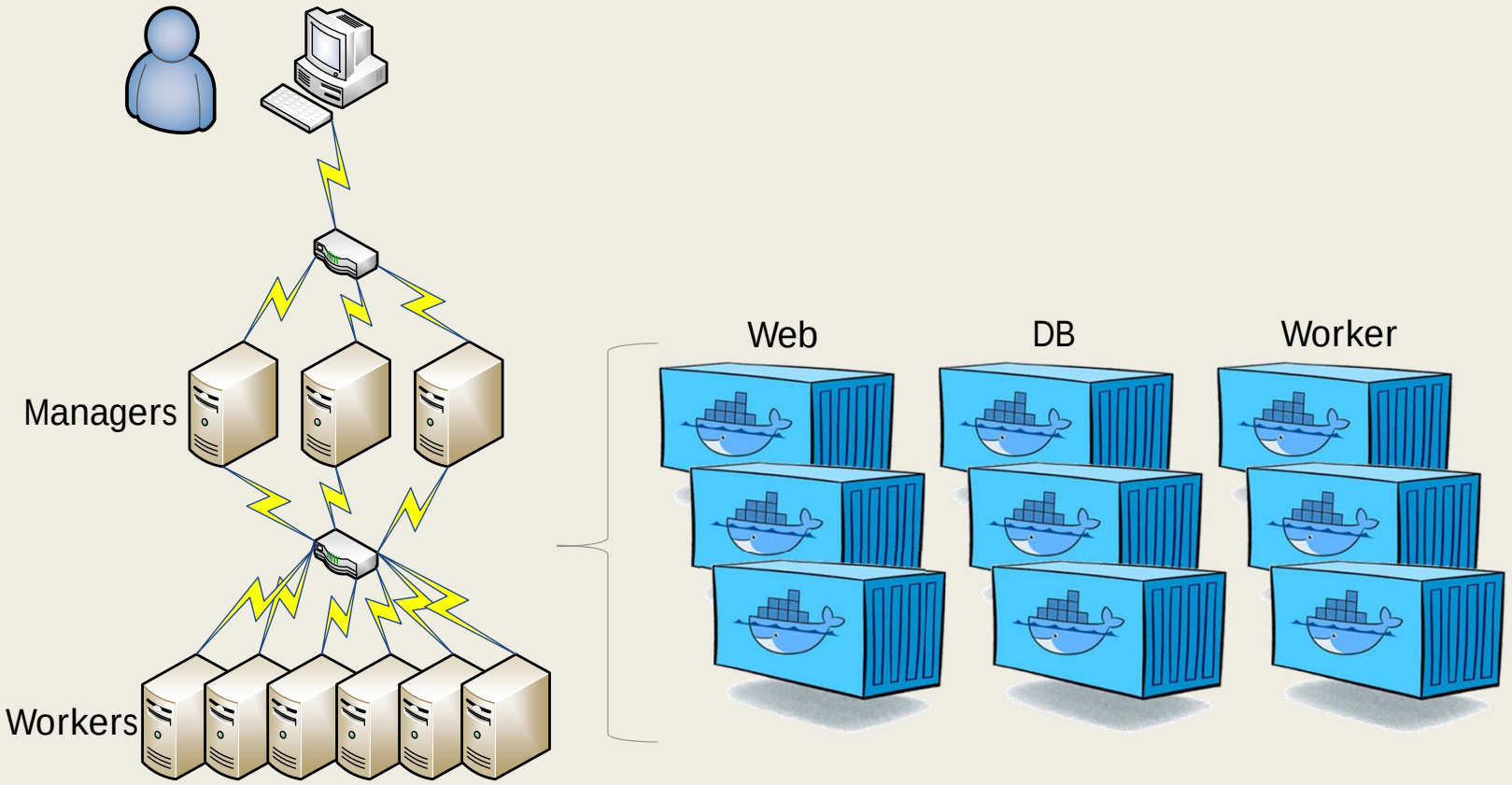
Load Balanced



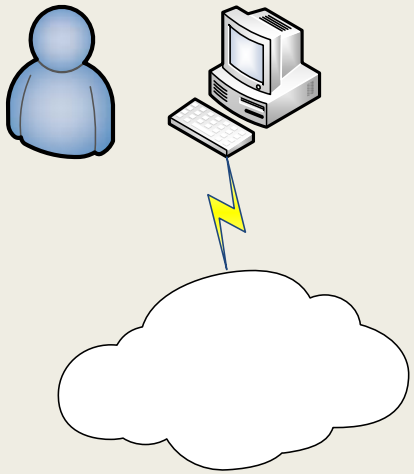
Distributed



Containerized Architectures



Serverless Architecture



AWS “Serverless” Services


Simple Storage Service (S3) – Cloud storage


CoudFront – Content delivery (CDN)


DynamoDB and RDS Aurora – NoSQL and Relational DB


Identity & Access Management (IAM) – Manage user access


Cognito – Identity Management and authentication


Simple Queue Service (SQS) – Message queueing


Simple Notification Service (SNS) – Push notifications and SMS


Lambda – Run code only when you need it

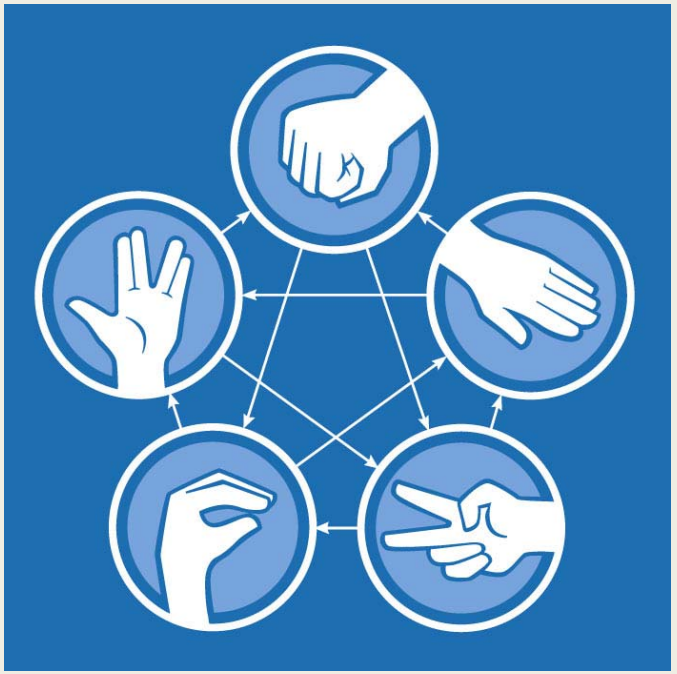

API Gateway – Manage APIs (including integration with Lambda)


Route 53 – Domain name management

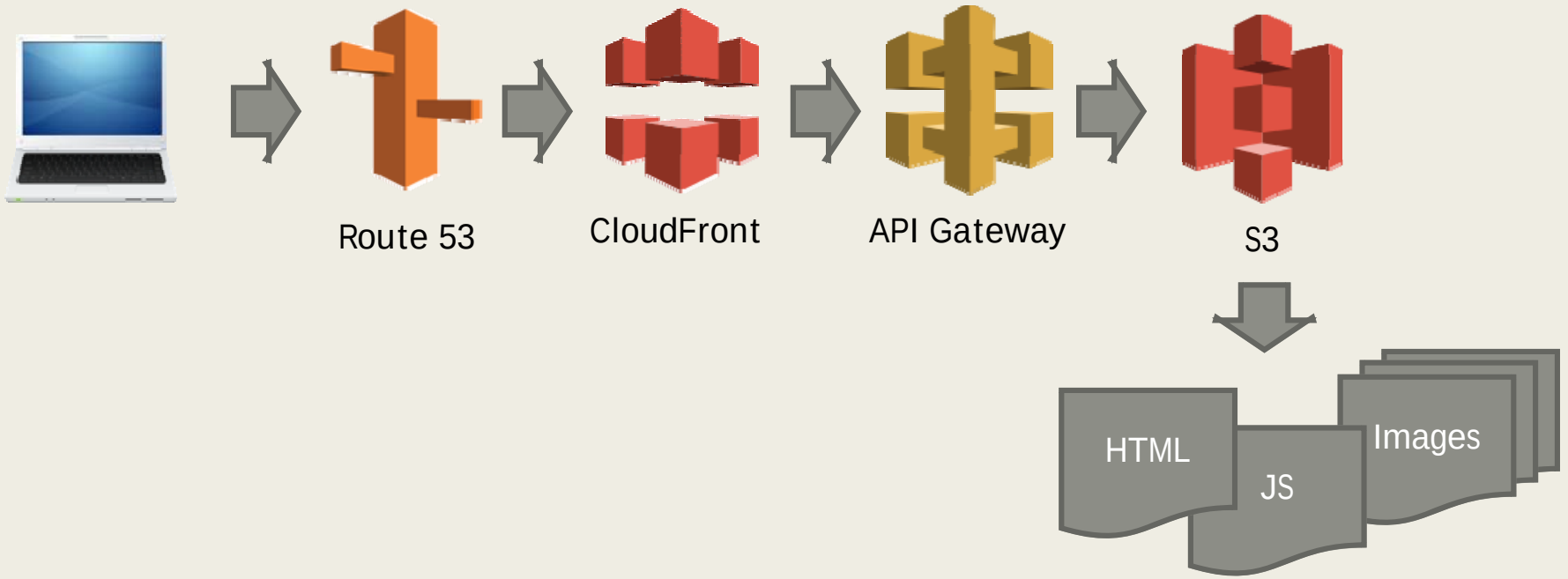

Simple Email Service (SES) – Send and receive e-mail

Sample Application:
Rock Paper Scissors Lizard Spock

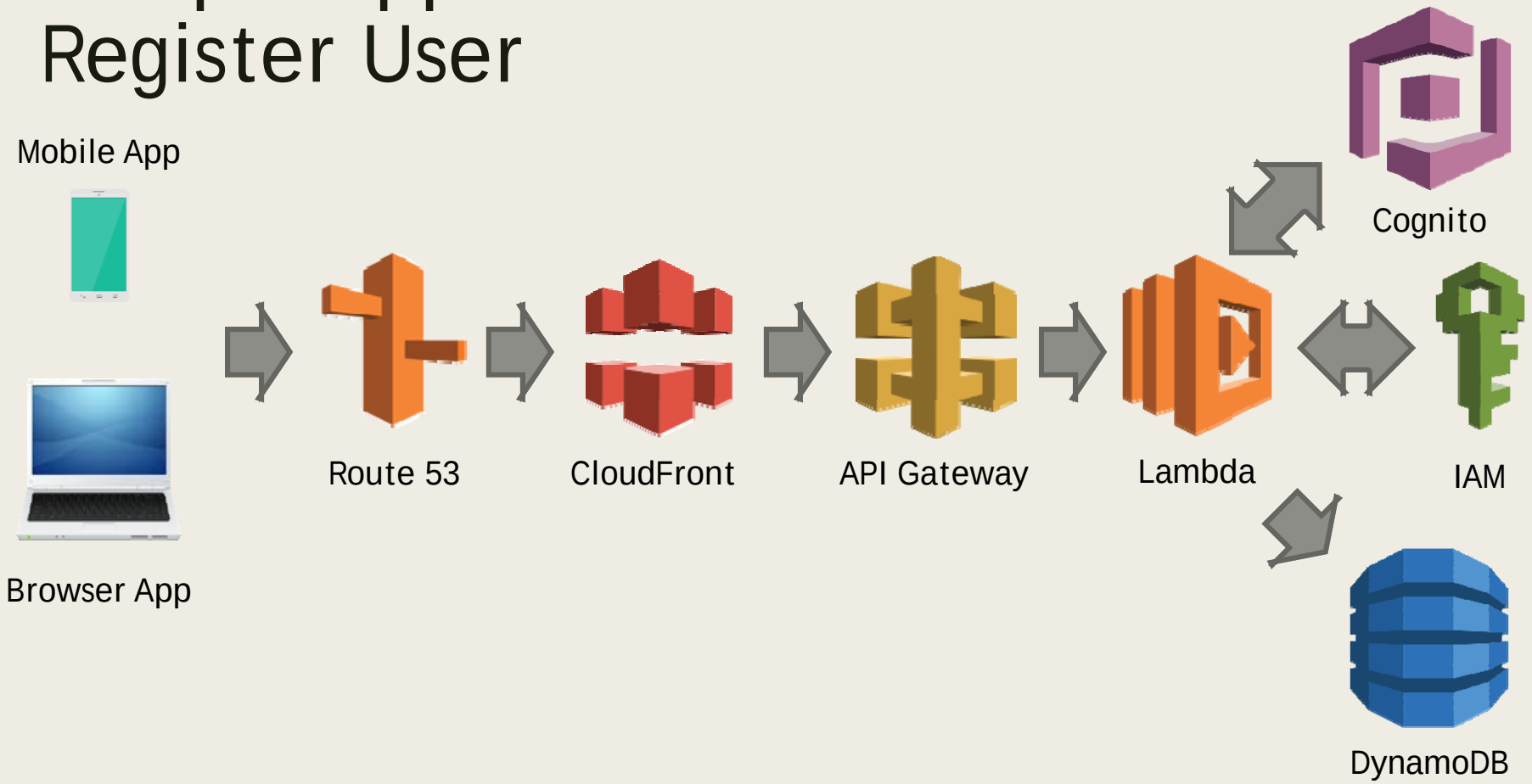
- User registration and login with 3rd party login support
- Find friends by phone or e-mail
- Challenge AI, friends, or strangers
- Record guesses
- Determine results
- Record results
- Track leader board



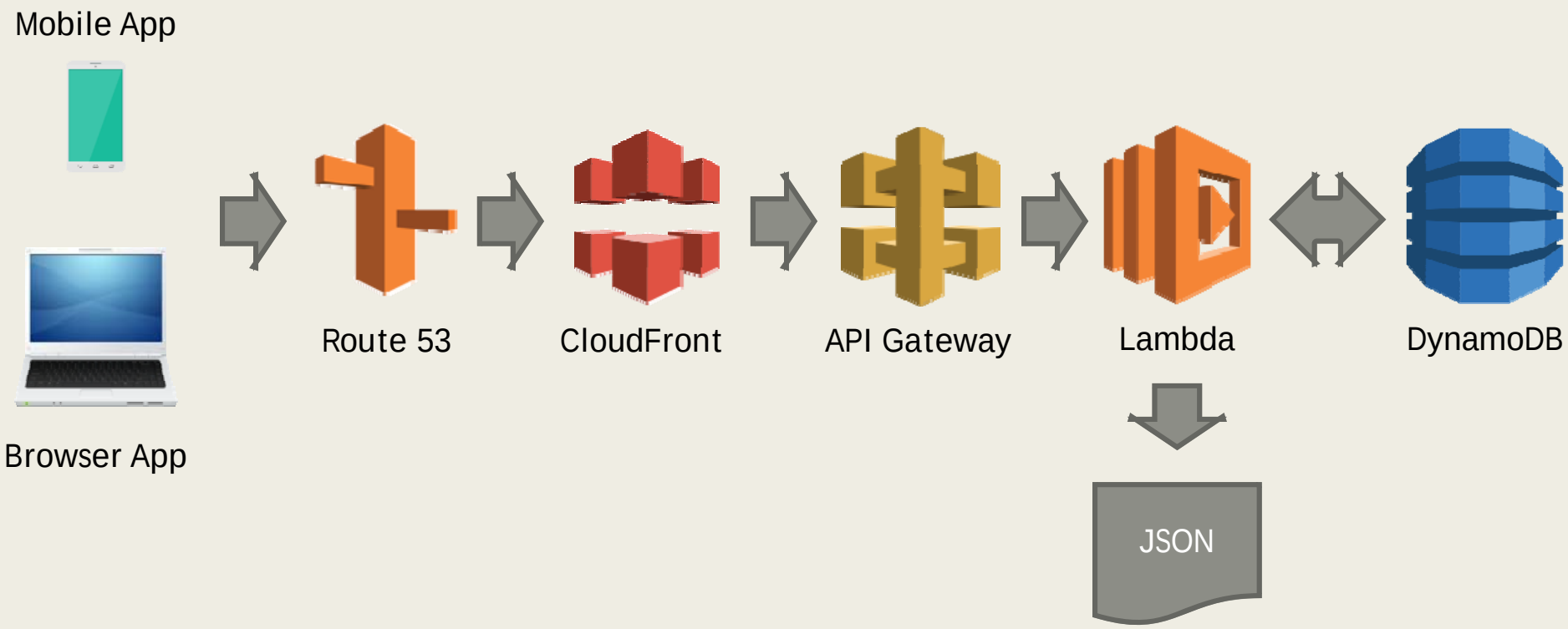
Sample Application Browser App Load



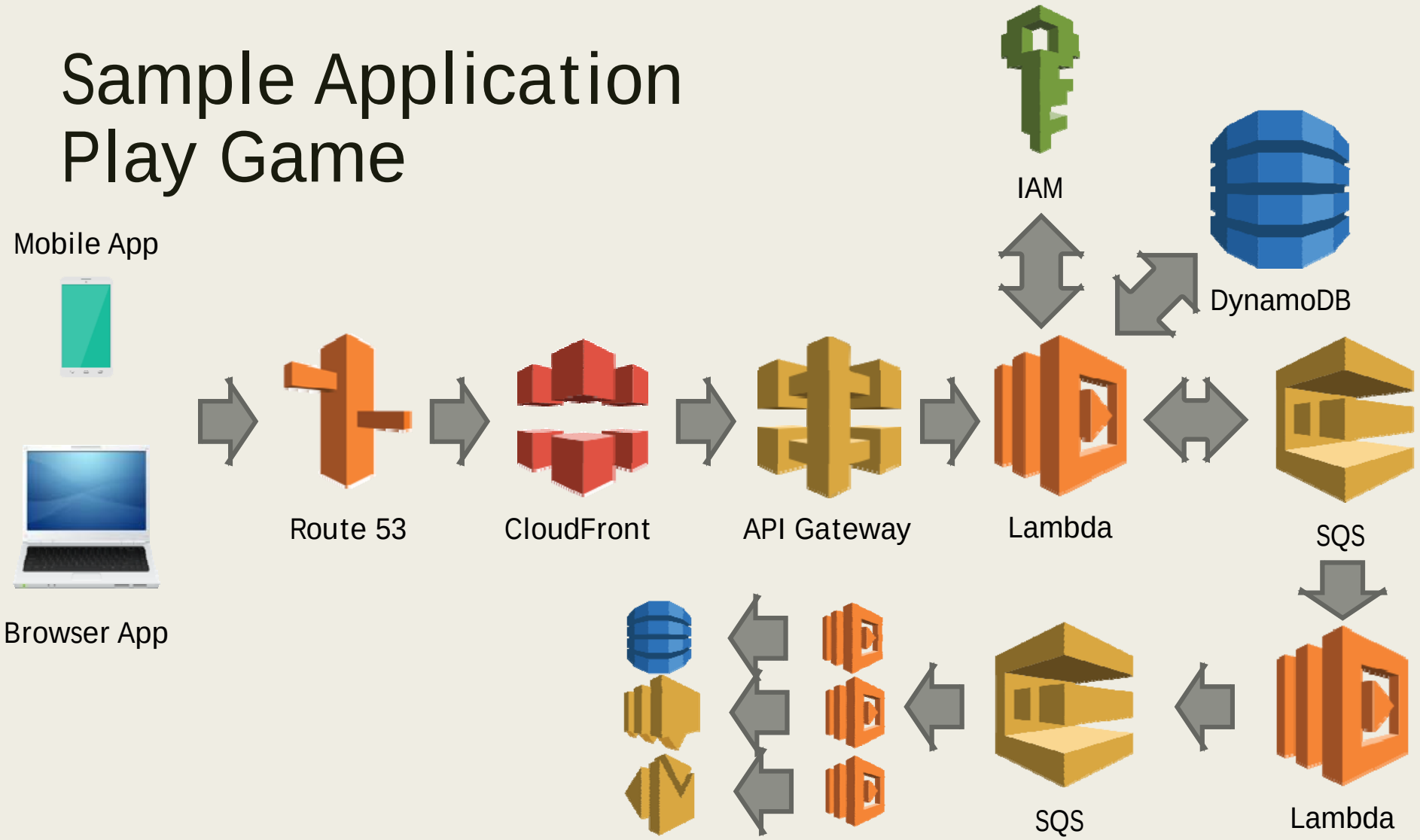
Sample Application
Register User



Sample Application Query Data



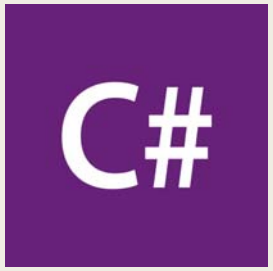
Sample Application Play Game

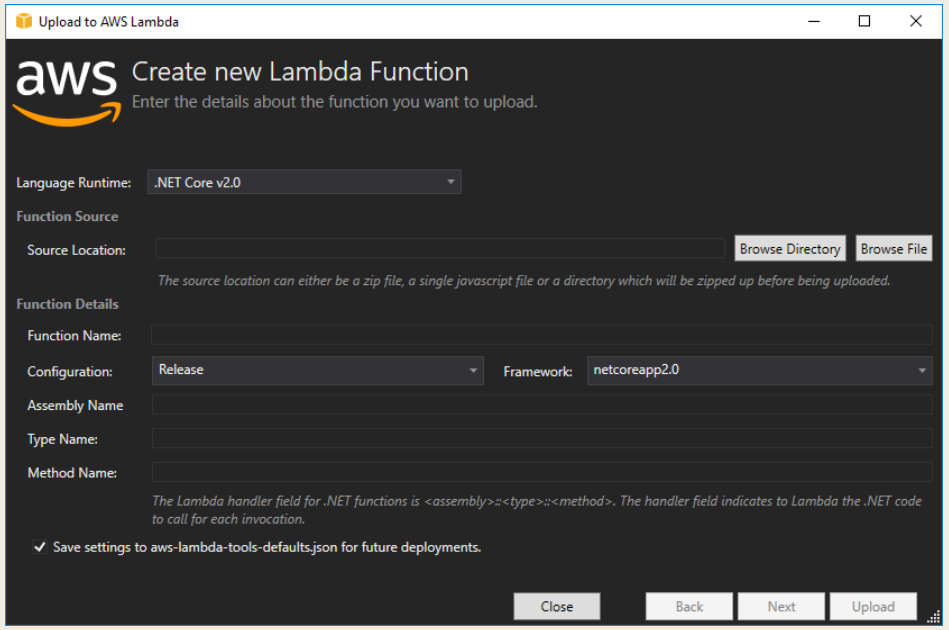


Takeaways

- Serverless architectures:
  - *Allow you to pay only for the computing resources you use*
  - *Encourage "good" design patterns that support resiliency, performance, and scale*
- But they also:
  - *Create opportunities for bad design patterns by locking into specific implementations*
  - *Can actually end up costing more than "serverfull" architectures*

AWS Lambda Supports C#





AWS SDK Extension for VS 2017

Appendix:
AWS “Less-Server” Services

- AWS Batch – Spin up server, run job, dispose of server
- Elastic Beanstalk – Spin servers up and down as demand changes
- Elastic Container Service (ECS/EKS) – AWS spins servers up and down as demand changes