Web-Based Applications at Scale

Outline

- Web Application Architecture - Recap
- Accessibility and Performance
- Scalability
- What is next for the Bookstore app?

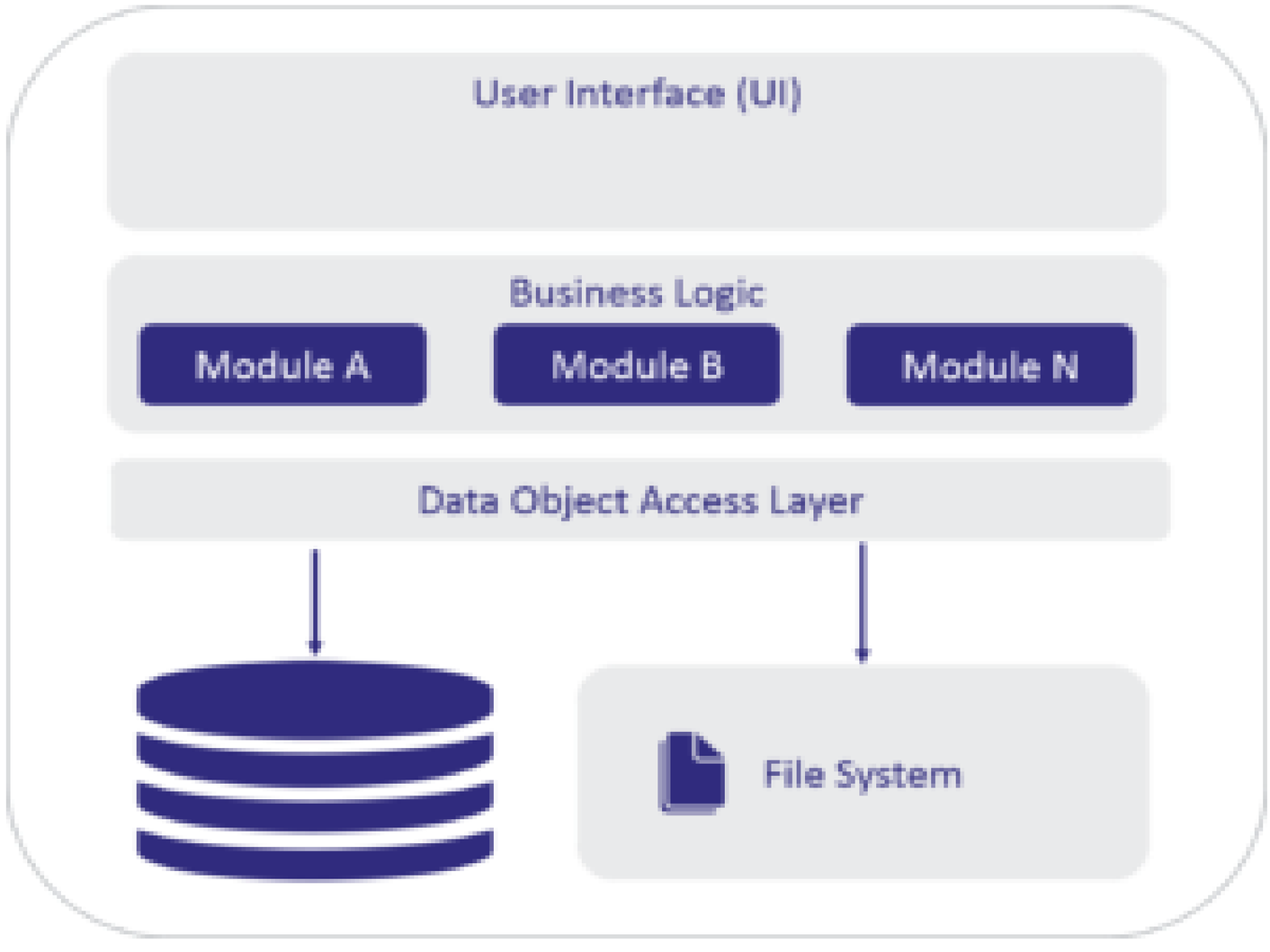
Web Application Architecture - Recap

- Web Application (Software) Architecture
  - Client Side:
    - Single-Page Applications (SPA)
    - Multi-Page Applications (MPA)
    - Progressive Web Applications (PWA)
  - Server Side:
    - Microservices
    - Monolith
    - Serverless

Web Application Architecture - Recap

- Server: **Monolith**
  - Built as a single unit that runs all or most of the functions
  - Advantages
    - Faster to design, develop, test and deploy
  - Disadvantages
    - Scalability
    - Reliability
    - Agility
    - Modifiability
- Popular choices include Java, Python, and PHP

Web Application Architecture - Recap

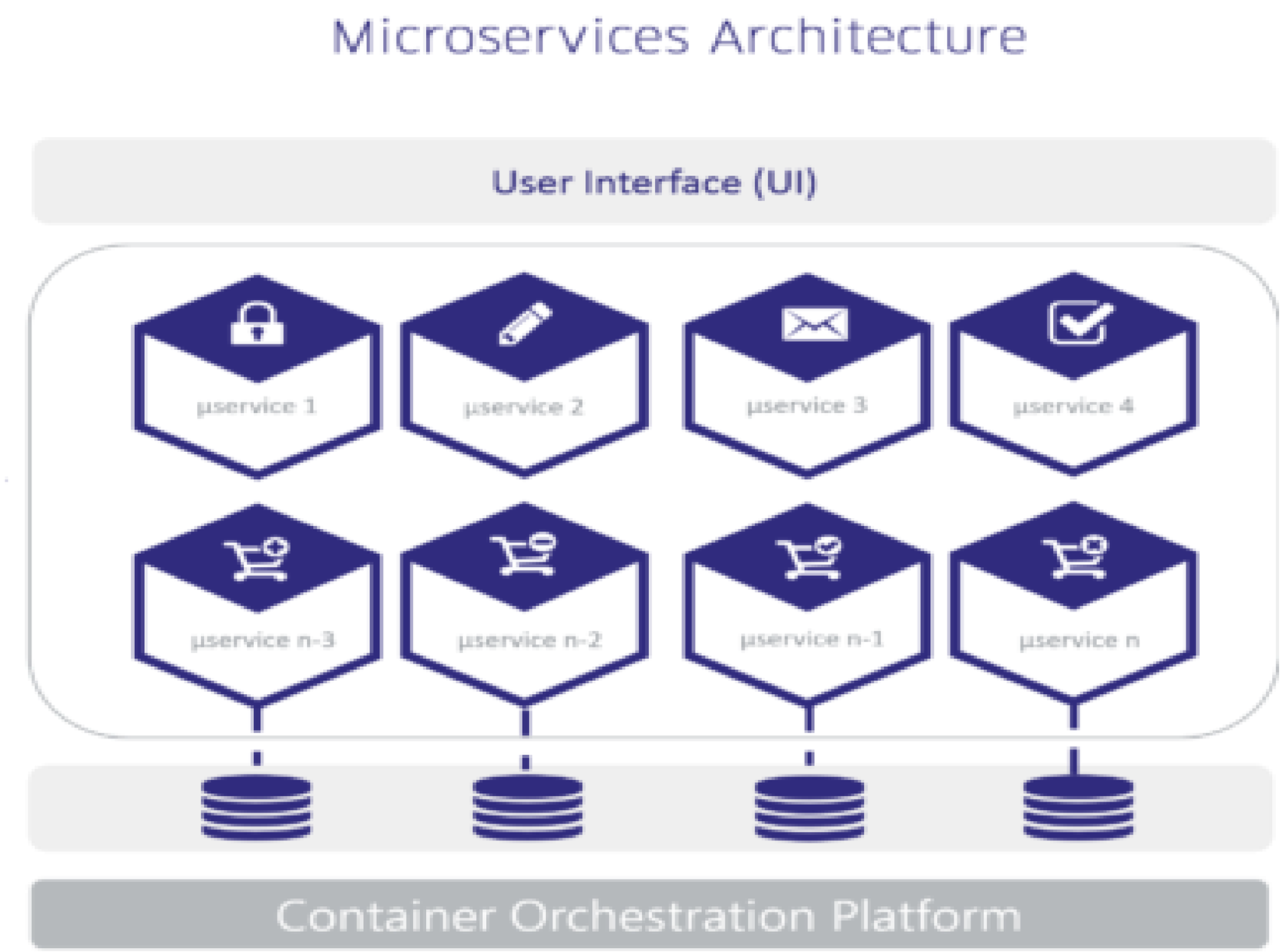


Source: https://www.sumerge.com/microservices-vs-monolithic-architecture/

Web Application Architecture - Recap

- Server: **Microservices**
  - Based on the principles of decentralization and modularity
  - Each microservice is built around a single business function and deployed independently
  - The most popular widely-applicable approaches to build highly-scalable web applications
  - Advantages
    - High reliability
    - Scalability
    - Faster time-to-market
    - Modifiable & Agile
  - Disadvantage
    - Development
    - Testing
  - Popular choices include Node JS, Express JS, Python, and Go

Web Application Architecture - Recap

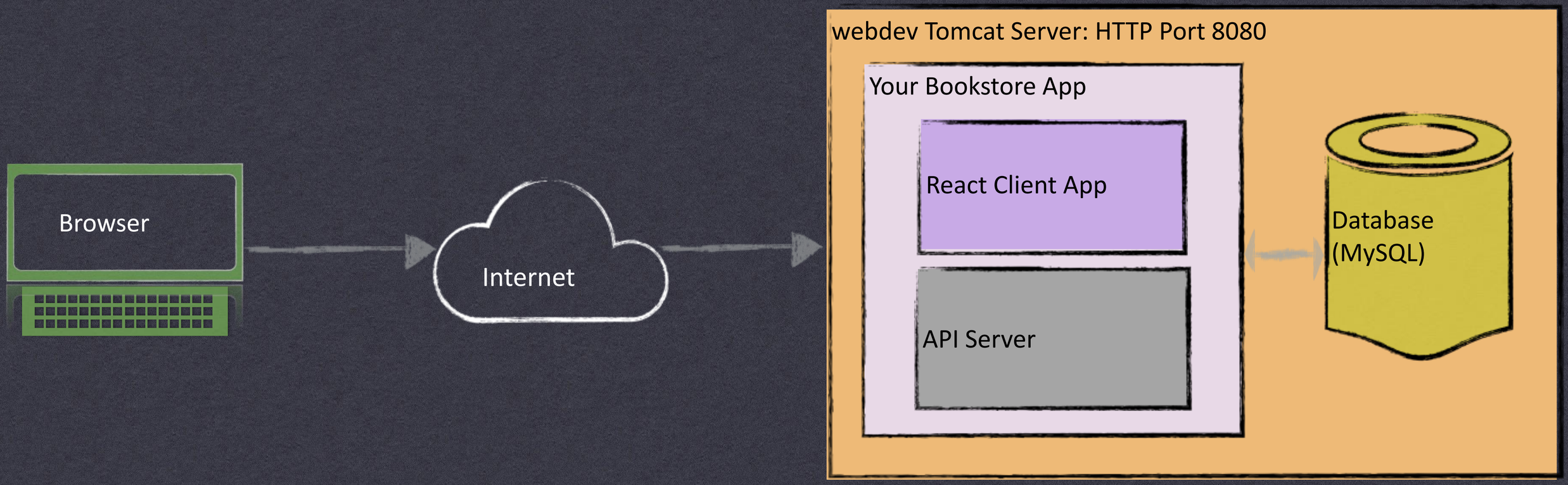


Source: https://www.sumerge.com/microservices-vs-monolithic-architecture/

Web Application Architecture - Recap

- Server: **Serverless**
  - Applications are developed as a set of functions
  - No need to maintain server infrastructure
    - Handled by Cloud service provides (AWS, Google Cloud,…)
  - Applications with varying workloads, rapid development needs, and unpredictable traffic
- Advantage:
  - Cost
  - Scalability
- Popular choices include Node JS, Express JS, Python, and Go

WHAT WE HAVE BUILT, AND WHY?



WHAT HAPPENS WHEN WE GROW?

Credit: Steve Atkinson (Dr. A)

Web Application at Scale

- Web based applications should be optimized for:
  - Accessibility
  - Performance
  - Scalability

Web Application at Scale

- Accessibility
  - look and Feel
    - Contrast, proximity, . . .
  - Accessible for diverse people
    - Physical ability, technology skills, age, . . .
- Internationalization
  - Language, culture, . . .
    - Intl (JavaScript object)
    - lang(html)
    - . . .

Web Application at Scale

- Performance
  - Efficiency, speed, and responsiveness
    - Loading time, perceived performance, . . .
    - Caching
      - Database, web, CDN, ….
    - Many automated tools for measurement
      - Lighthouse, Core Web Vitals, Google Analytics, . . .

Web Application at Scale

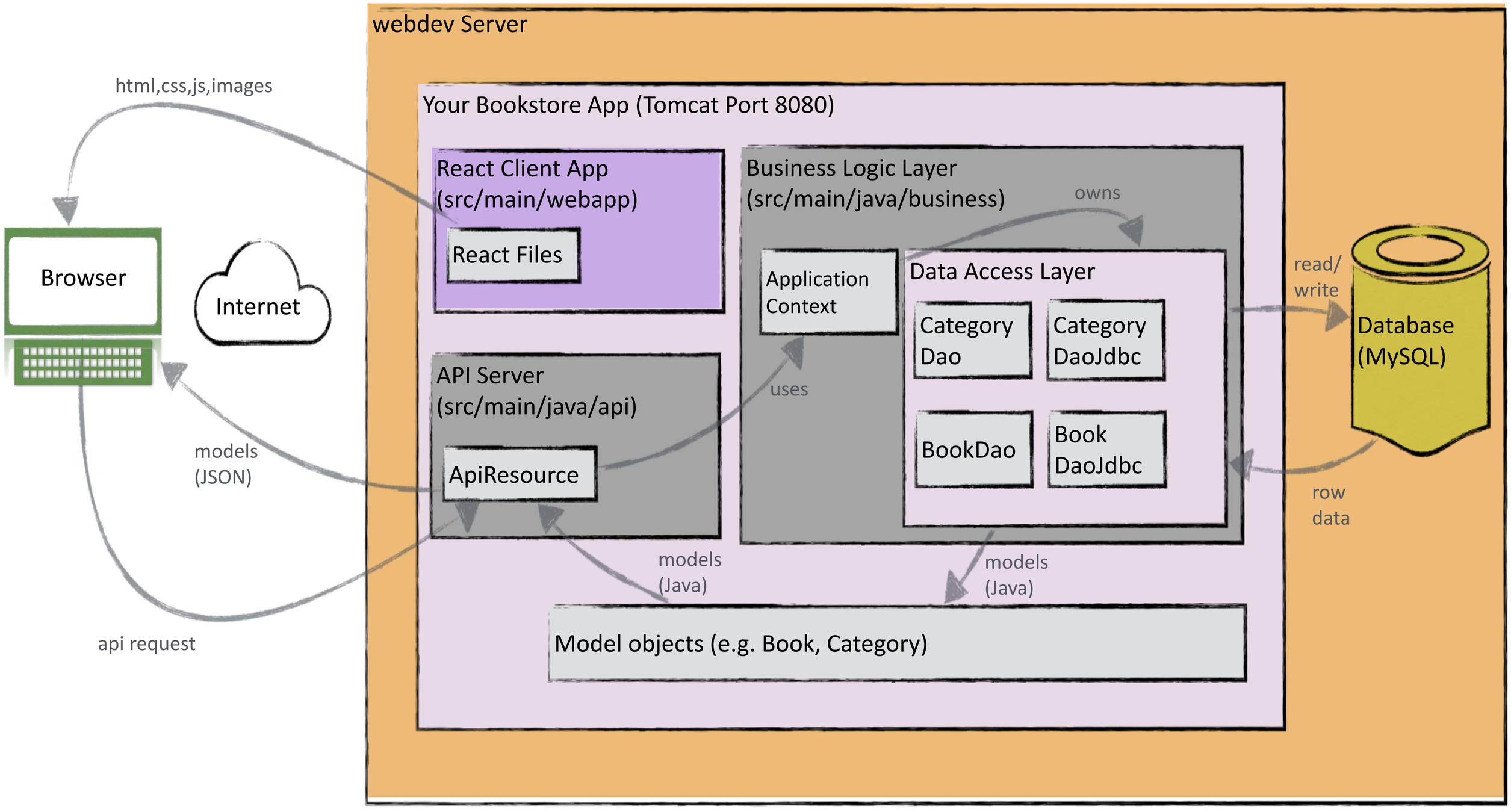
- Scalability
  - Ability to handle increased traffic and data, while preserving the application's efficiency and reliability
  - Can be :
    - Vertical
      - Increasing the resources on a server.
- Horizontal
  - The addition of more computational servers and parallel resources
  - Additional load balancing may be needed to distribute requests

Web Application at Scale

Vertical Scaling

Horizontal Scaling

What have we built?

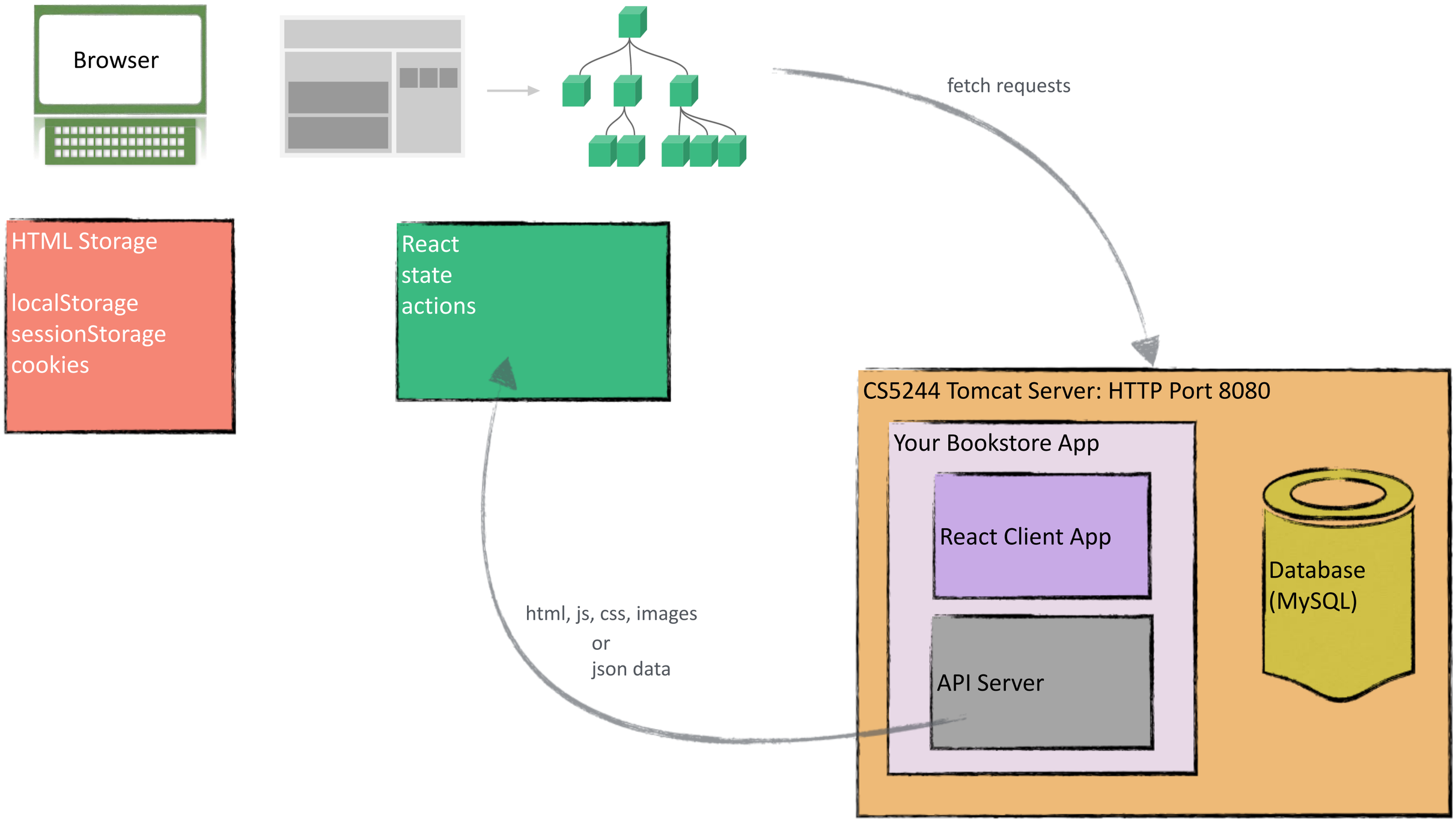


Credit: Steve Atkinson (Dr. A)

Why have we built this way?

- The API allows us to serve clients other than React browser clients (e.g. mobile phones)
    - It can be versioned and support many formats
- The business layer is separated from the API
    - model objects and logic (e.g. validation) can be re-used in many different APIs and applications.
- If we wanted to change our database, changes would be isolated to our Data Access Layer.
    - This involves carefully wrapping exceptions to hide details

Front End Architecture



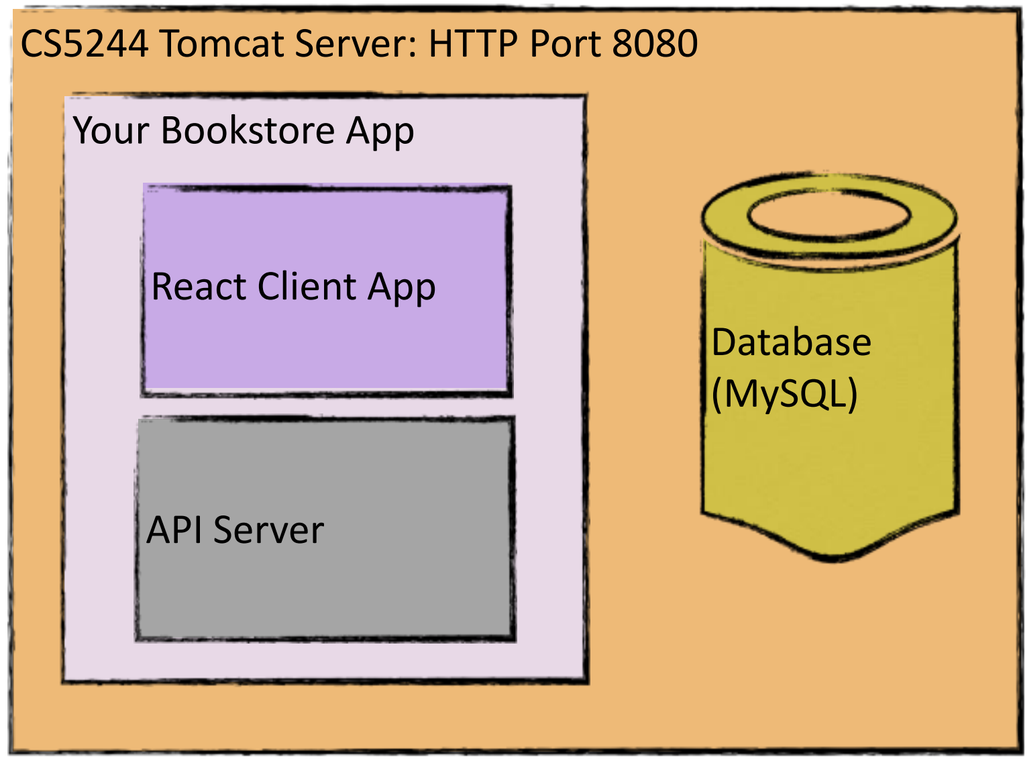
Credit: Steve Atkinson (Dr. A)

Why have we built this way?

- The React client is separately build-able, and deployed separately
  - Web developers can work independently via the API
  - React Components are reusable
  - React Components are responsive

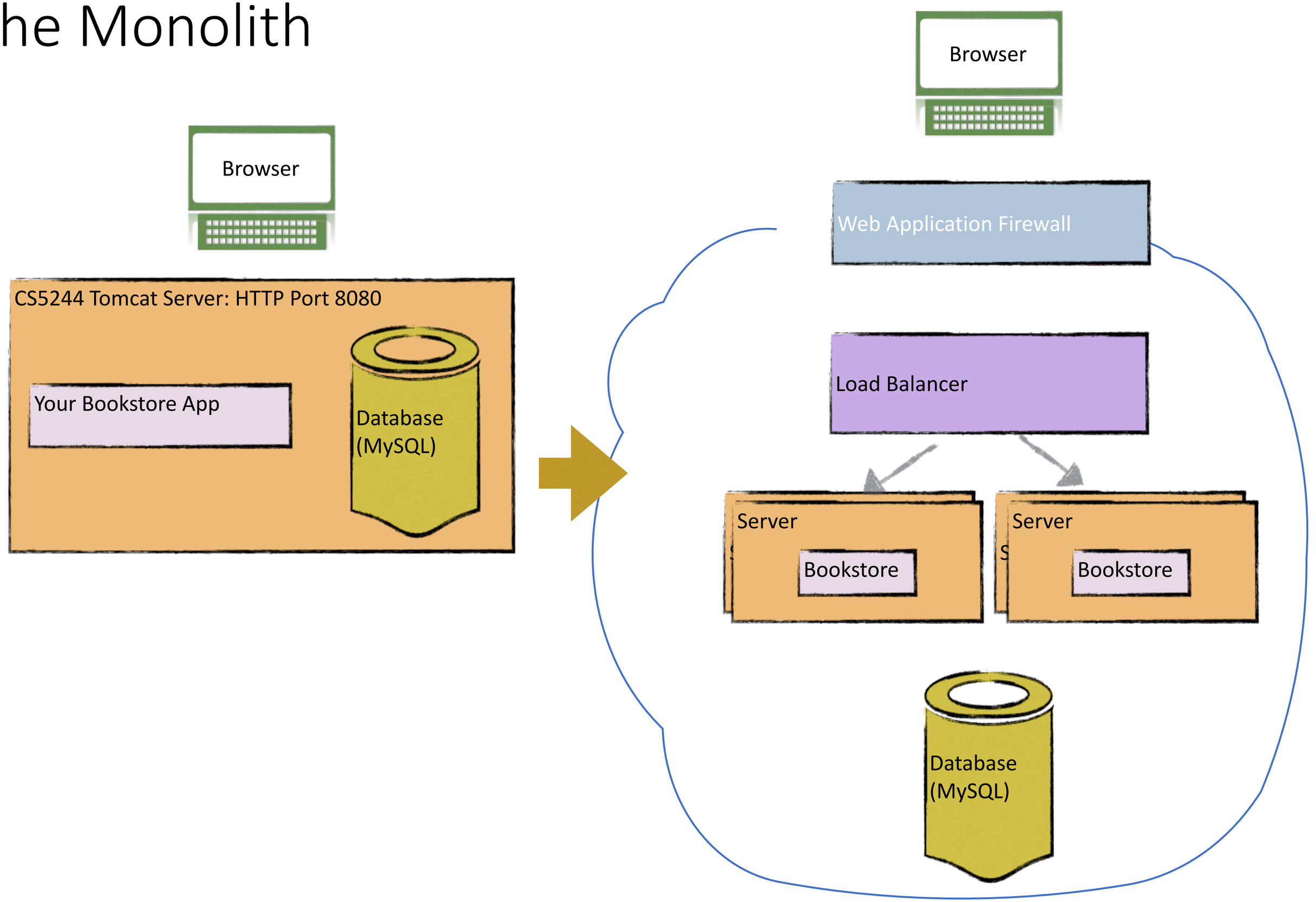
Operating the Mini-Site

- One central server machine is cost effective
- Data:
  - use a virtual machine with snapshots.
- Mitigations:
  - install a TLS certificate,
  - run a replica mySQL database



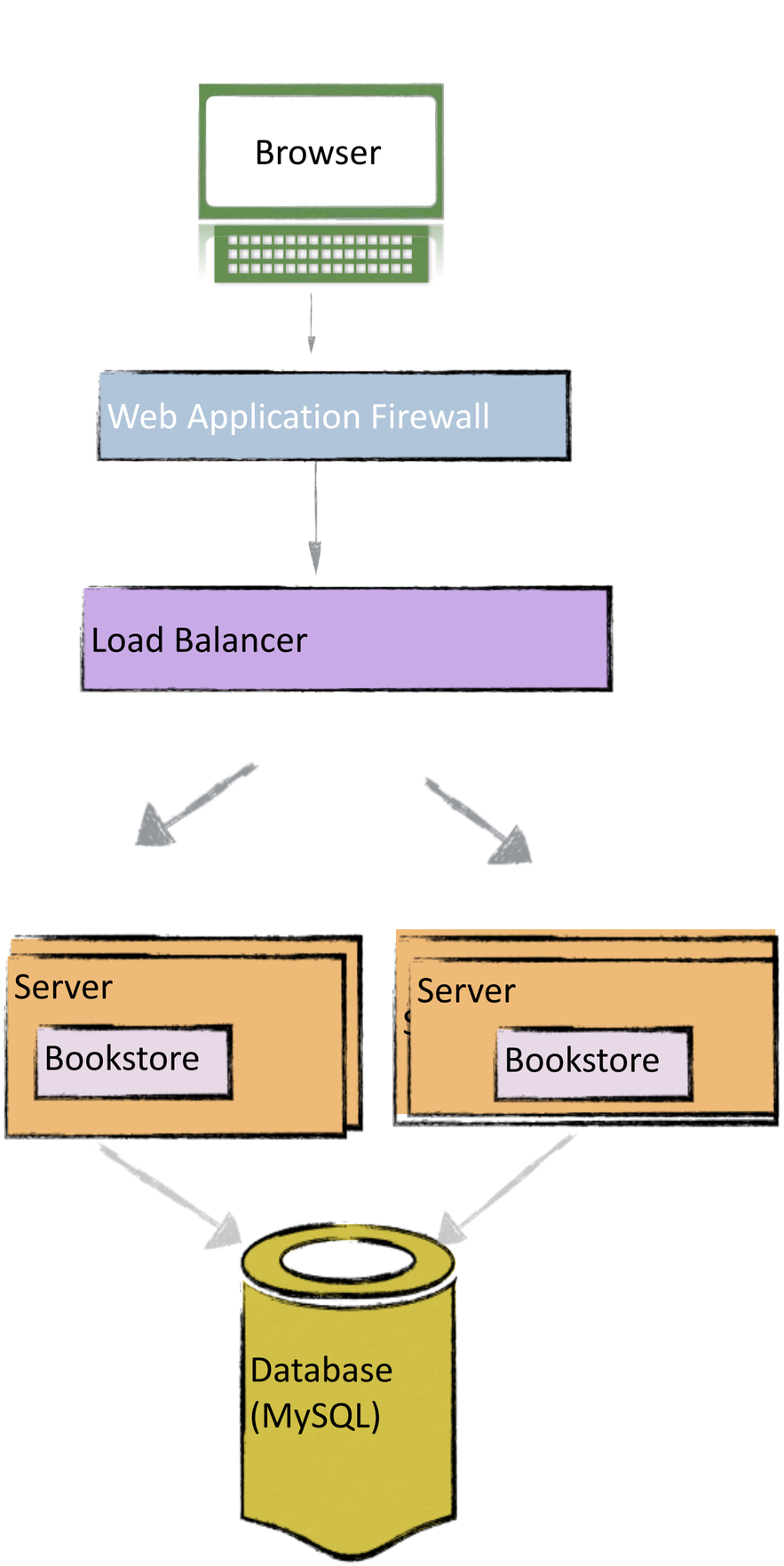
Credit: Steve Atkinson (Dr. A)

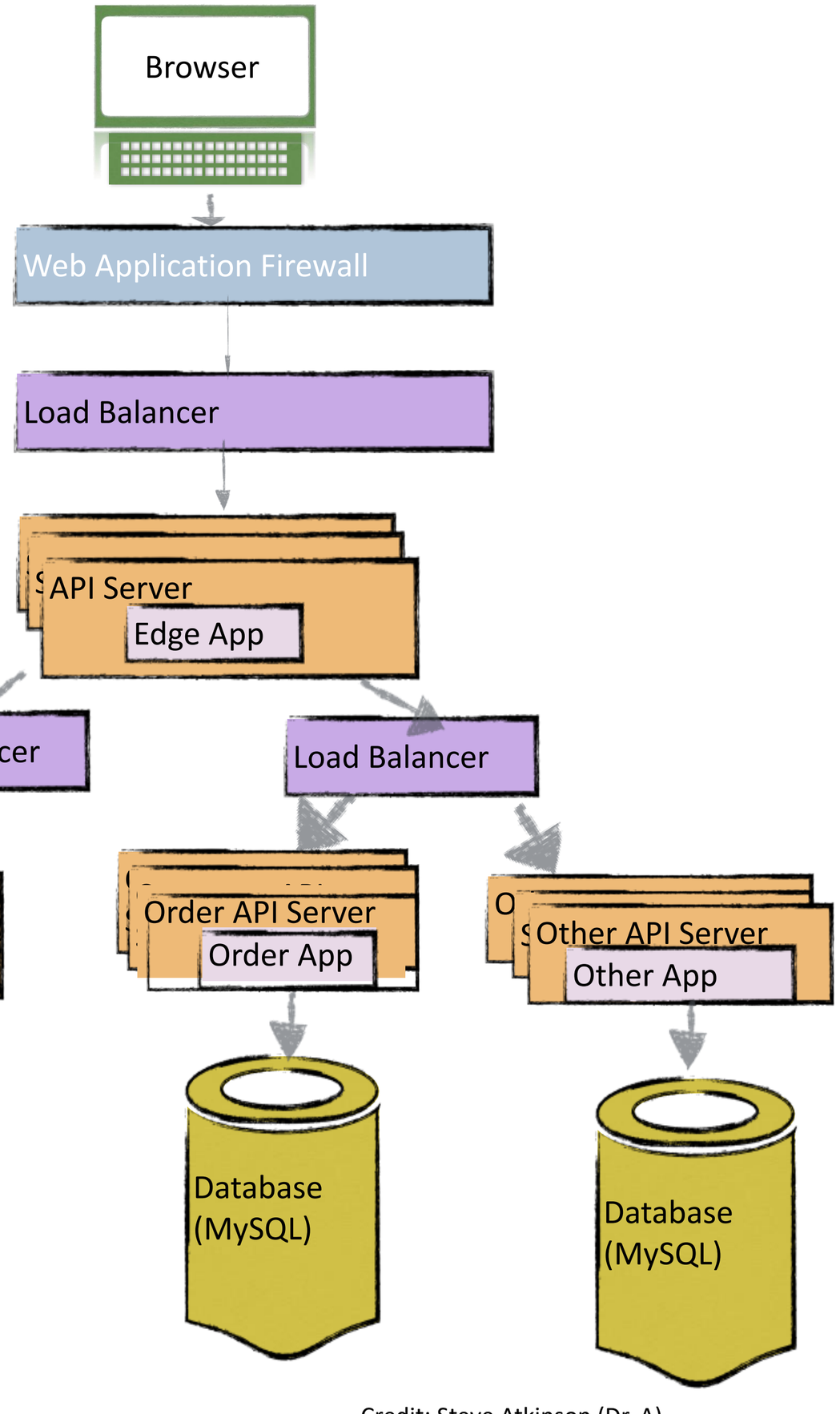
The Monolith



Credit: Steve Atkinson (Dr. A)

MICRO-SERVICES



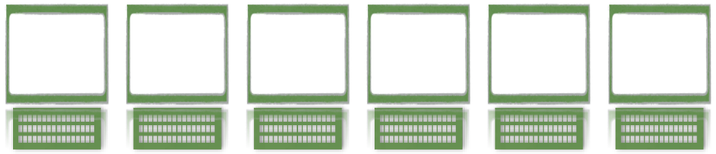


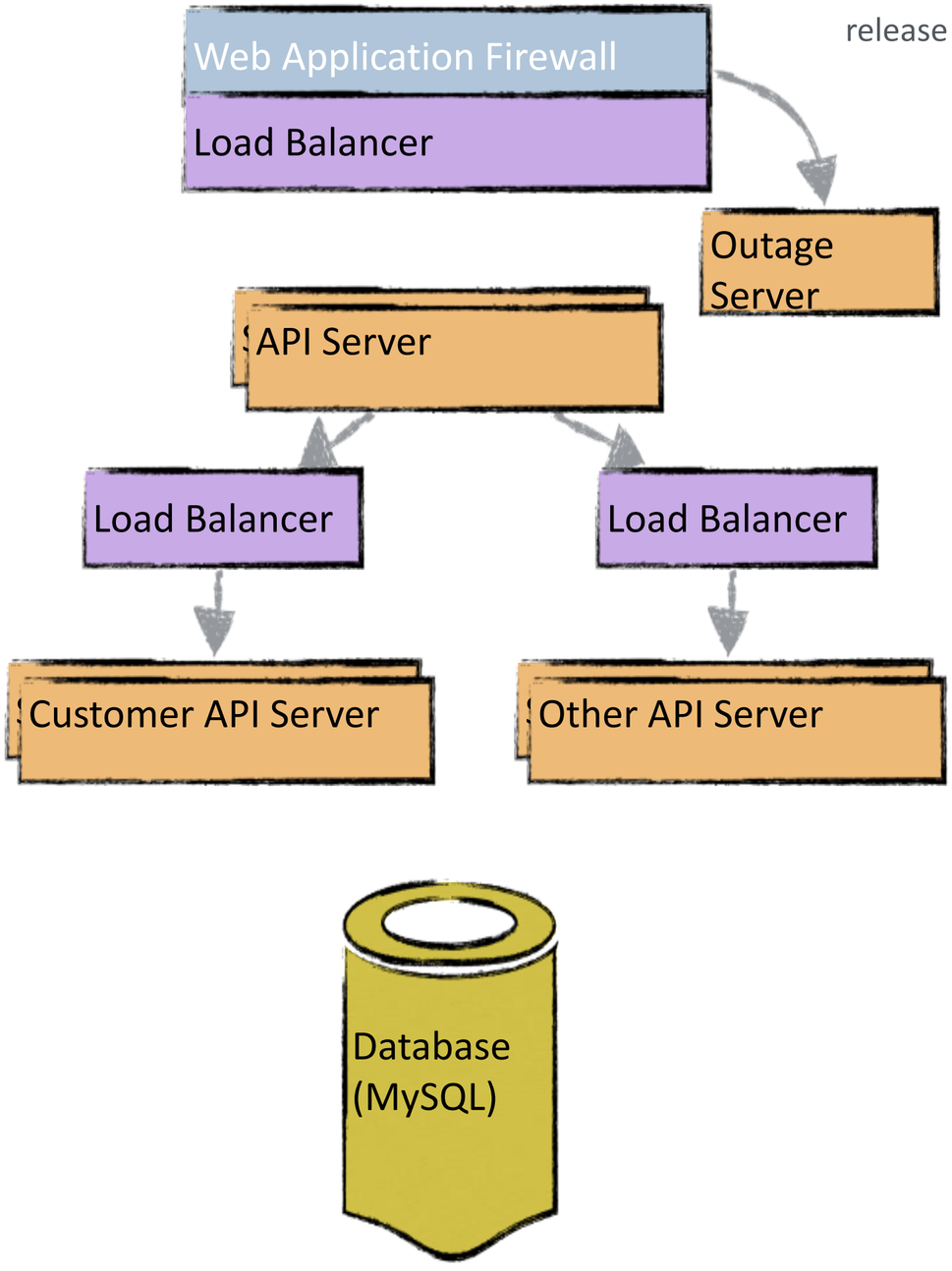
Credit: Steve Atkinson (Dr. A)

Web Application at Scale

- Deployment infrastructure
  - On-premise
  - Cloud
  - Hybrid

When you grow globally...





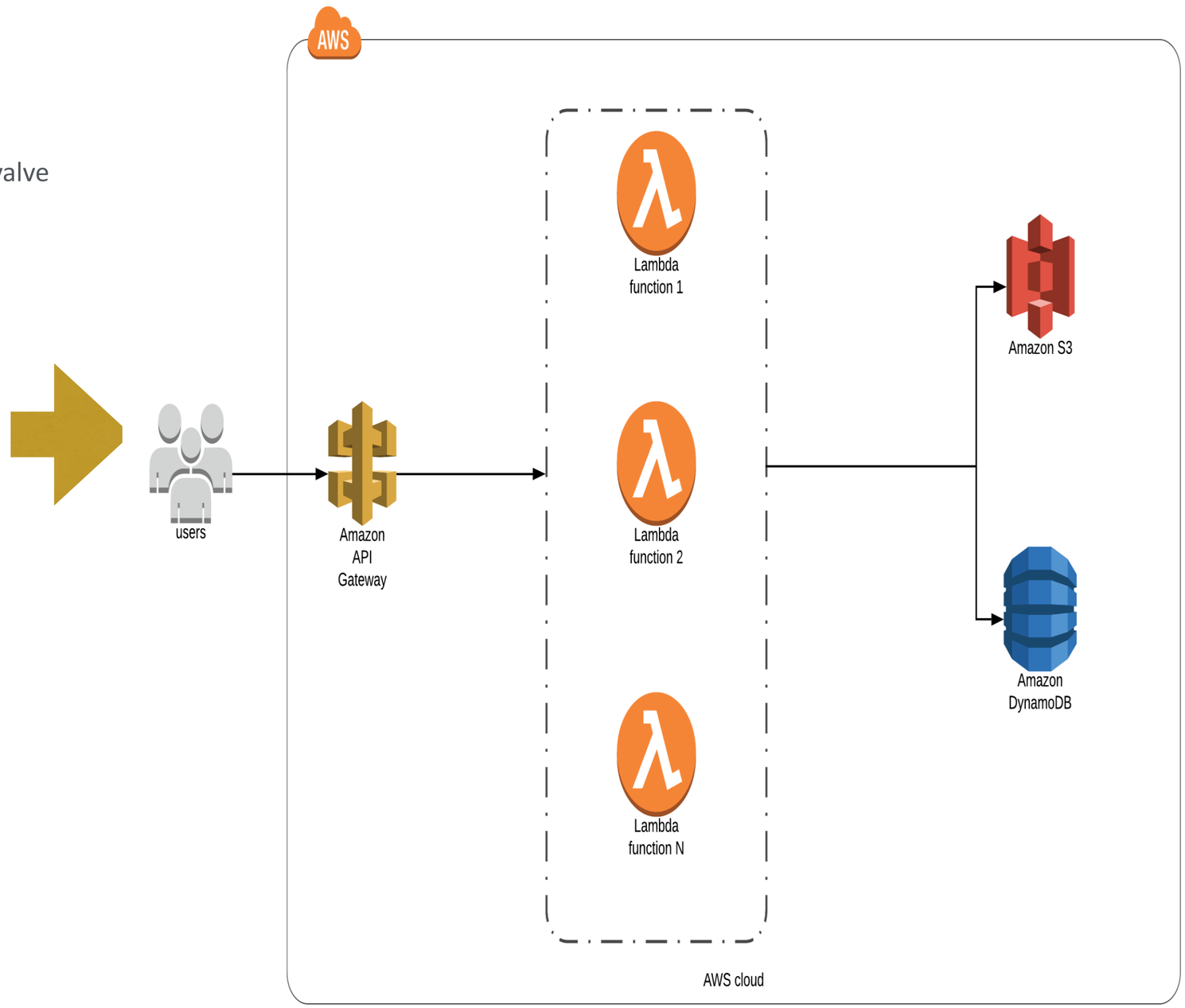


Figure:Attendance System
AWS Lambda

Credit: Steve Atkinson (Dr. A)

Web Application at Scale

- Different cloud solutions:
  - Google cloud function
  - Amazon Lambda
  - Microsoft Azure

References

- https://developers.google.com/learn/pathways/solution-three-tier-cloud-run
- https://github.com/donnemartin/system-design-primer