



WEB DEVELOPMENT

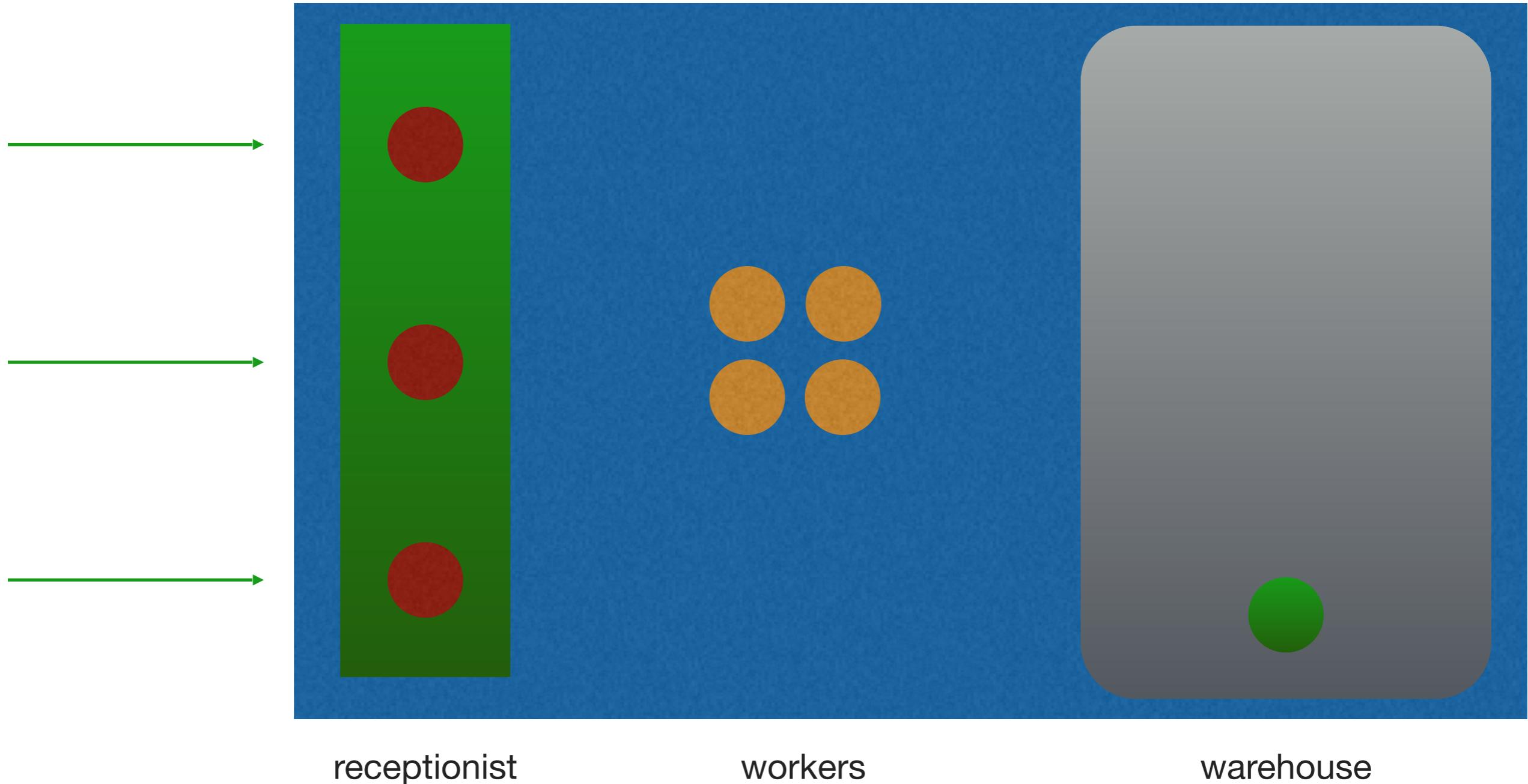
Lesson 1

What is a Web Application?

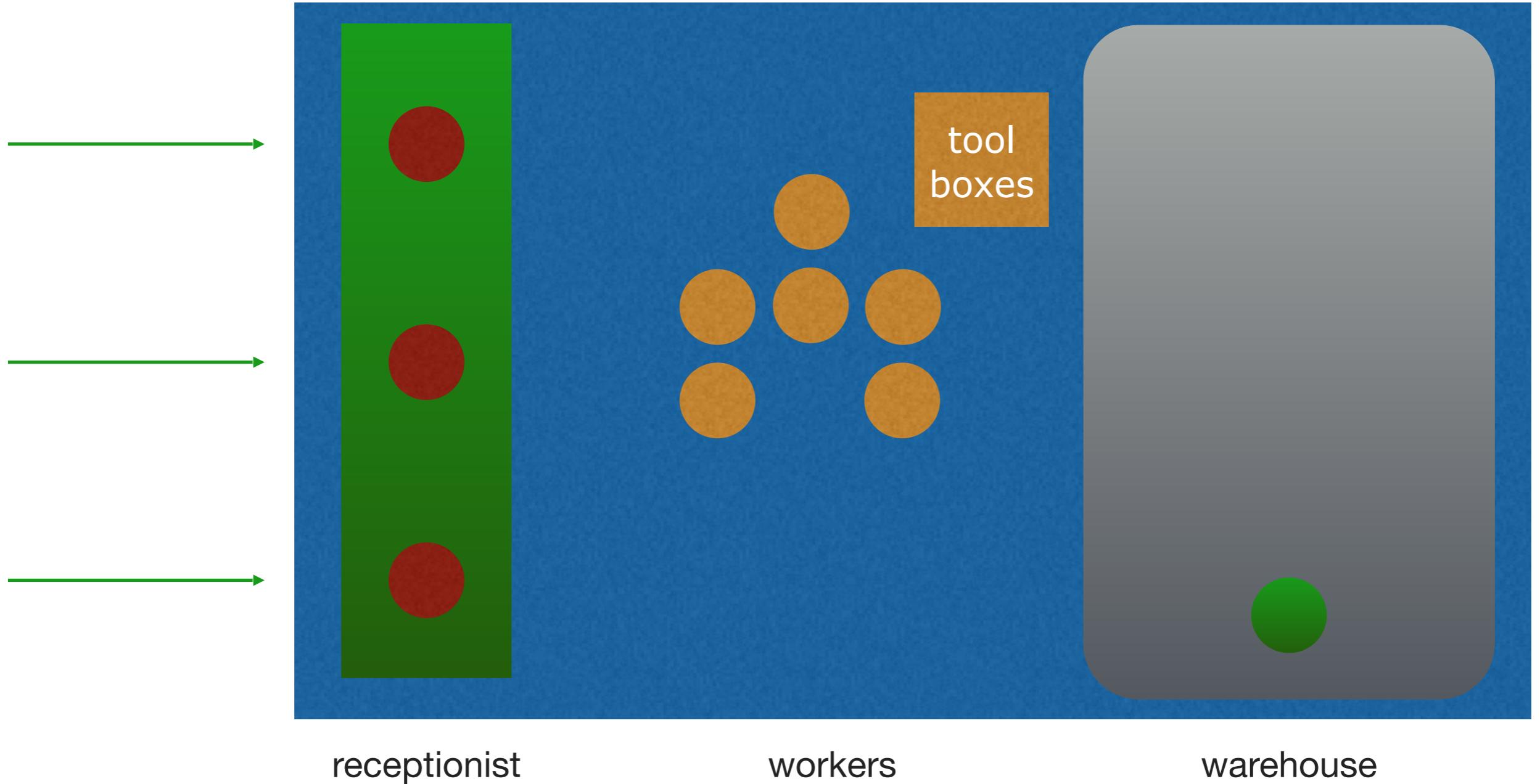
A web application is software that runs on a web server and is accessed via a browser, without needing installation.

How does the Web work?

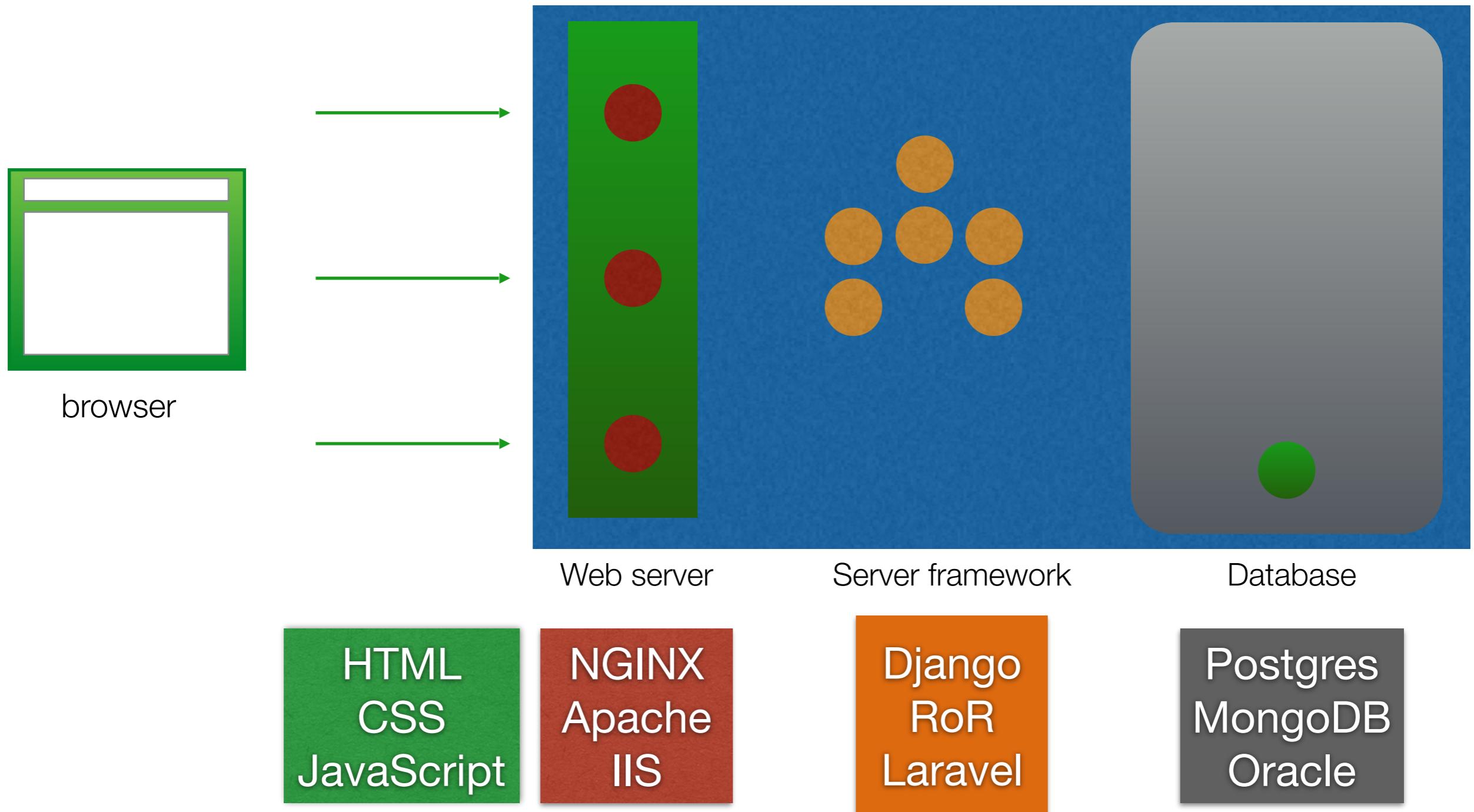
Furniture store

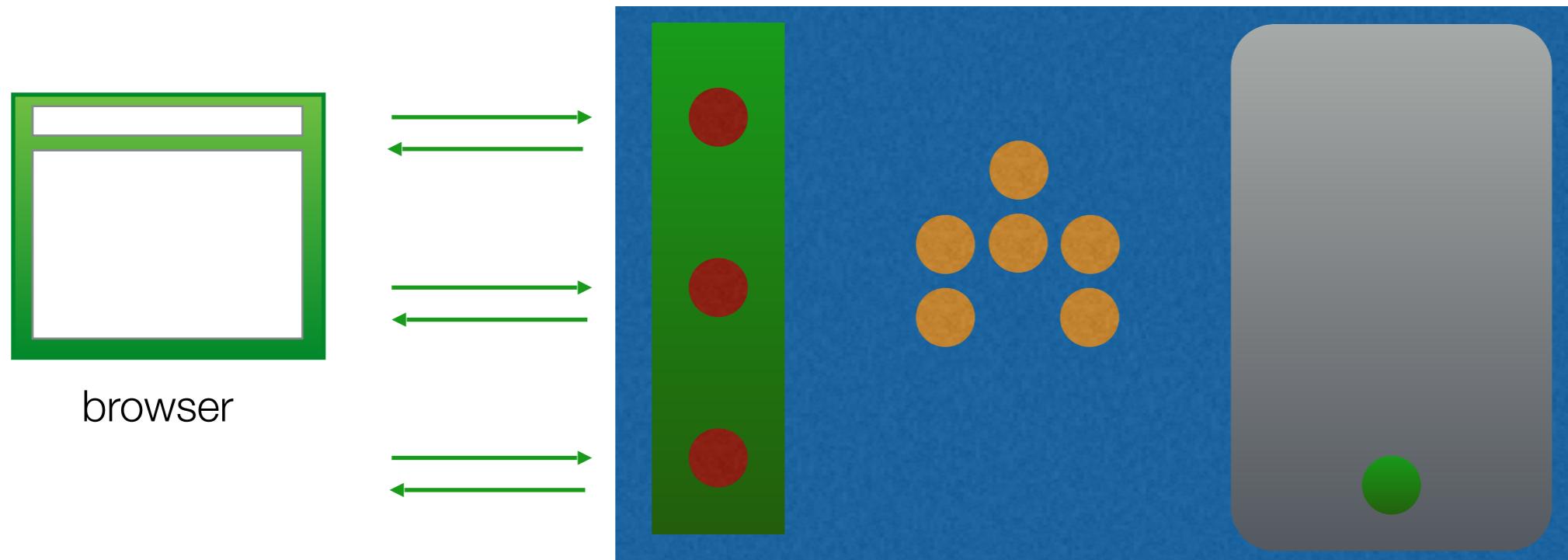


Making the store more responsive



Web development terminology

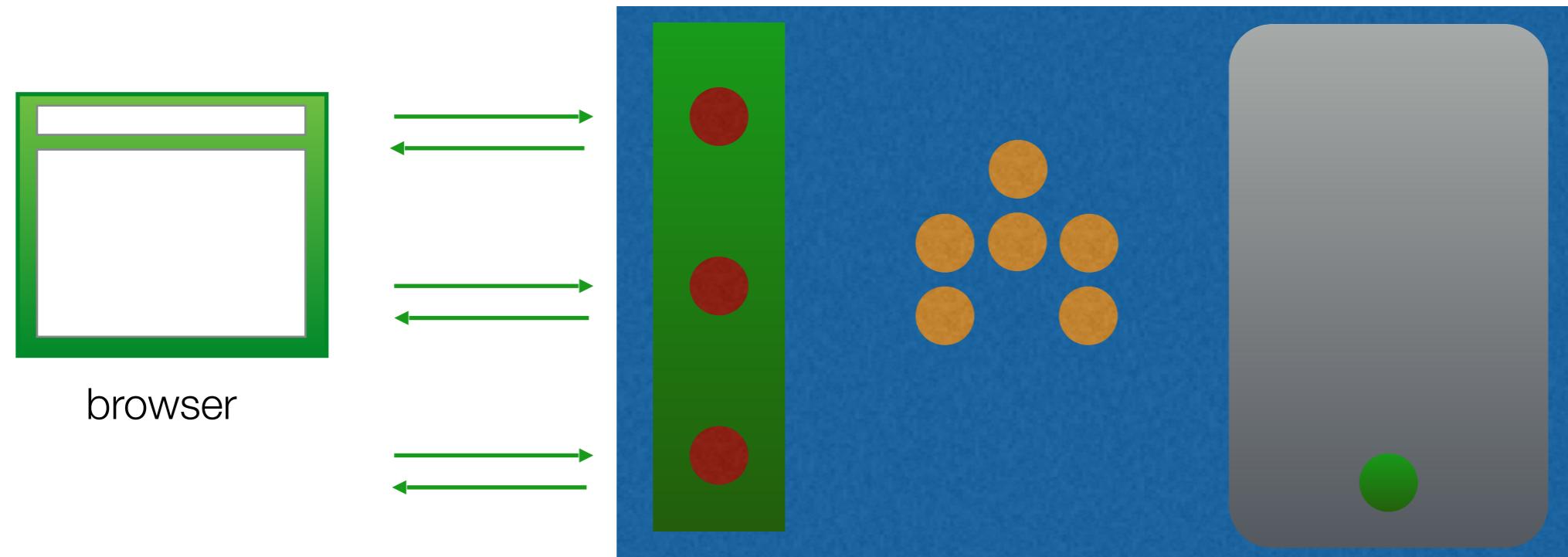




	Web server	Server framework	Database
HTML CSS JavaScript	NGINX Apache IIS Unicorn Thin Puma	Python: Django, FastAPI, Flask Ruby: Rails, Sinatra Java: Spring, Play Go: Gin, Fiber C#: <u>ASP.NET</u>	MySQL Postgres Microsoft SQL SQLite Oracle MongoDB Cassandra Redis Memcached
HAML LESS SASS CoffeeScript TypeScript		PHP: Laravel, Yii	
jQuery React Angular Vue Next.js Svelte Nuxt.js			

Hosting

AWS
Microsoft Azure
Heroku
ps.kz
hoster.kz



browser

Hosting

- AWS
- Microsoft Azure
- Heroku
- ps.kz
- hoster.kz

front end	Web server	Server framework	Database
HTML CSS JavaScript HAML LESS SASS CoffeScript TypeScript jQuery React Angular Ember Vue	NGINX Apache IIS Unicorn Thin Puma	Python: Django, FastAPI, Flask Ruby: Rails, Sinatra Java: Spring, Play Go: Gin, Fiber C#: <u>ASP.NET</u> PHP: Laravel, Yii	MySQL Postgres Microsoft SQL SQLite Oracle MongoDB Cassandra Redis Memcached
client side			

front end

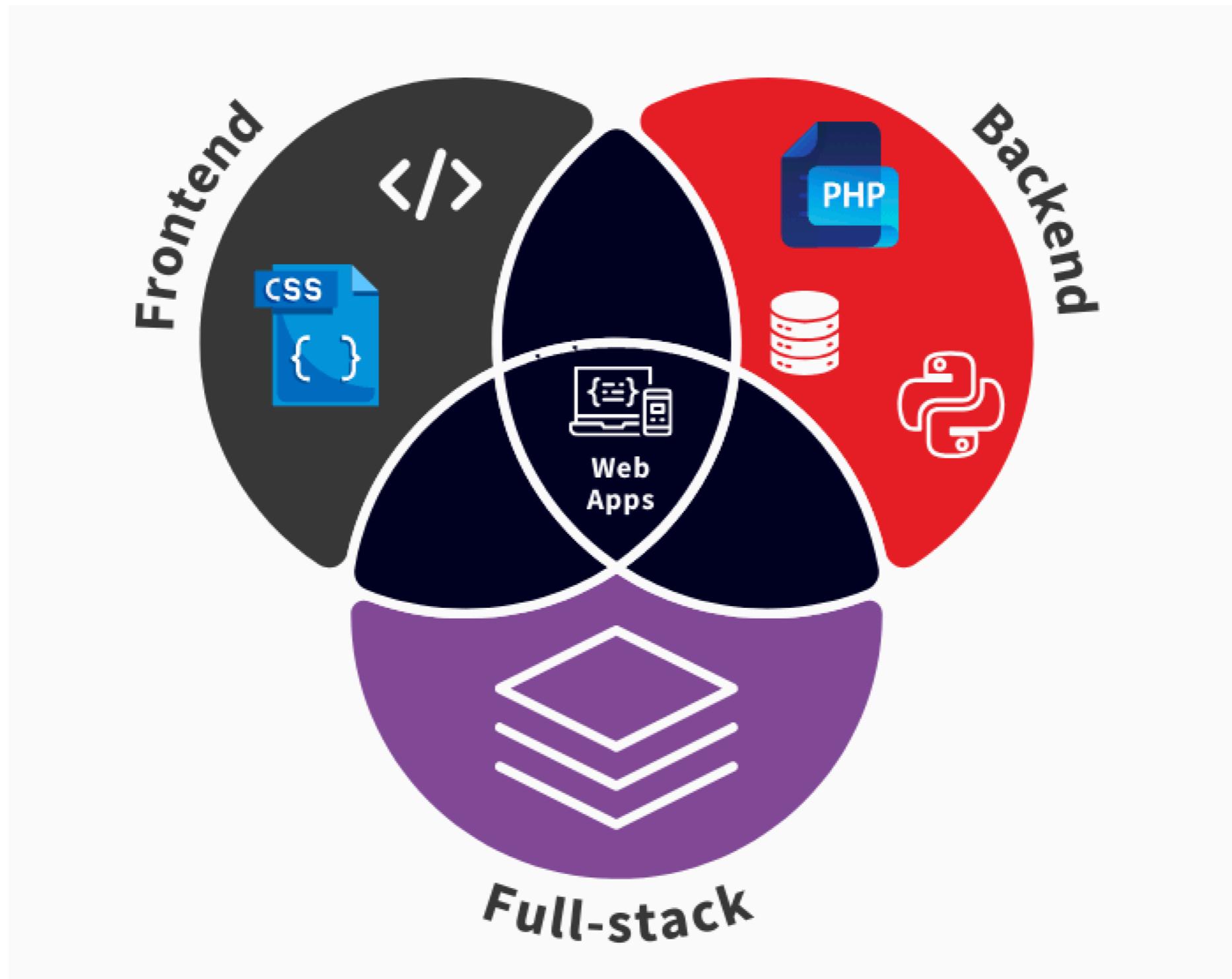
back end

server

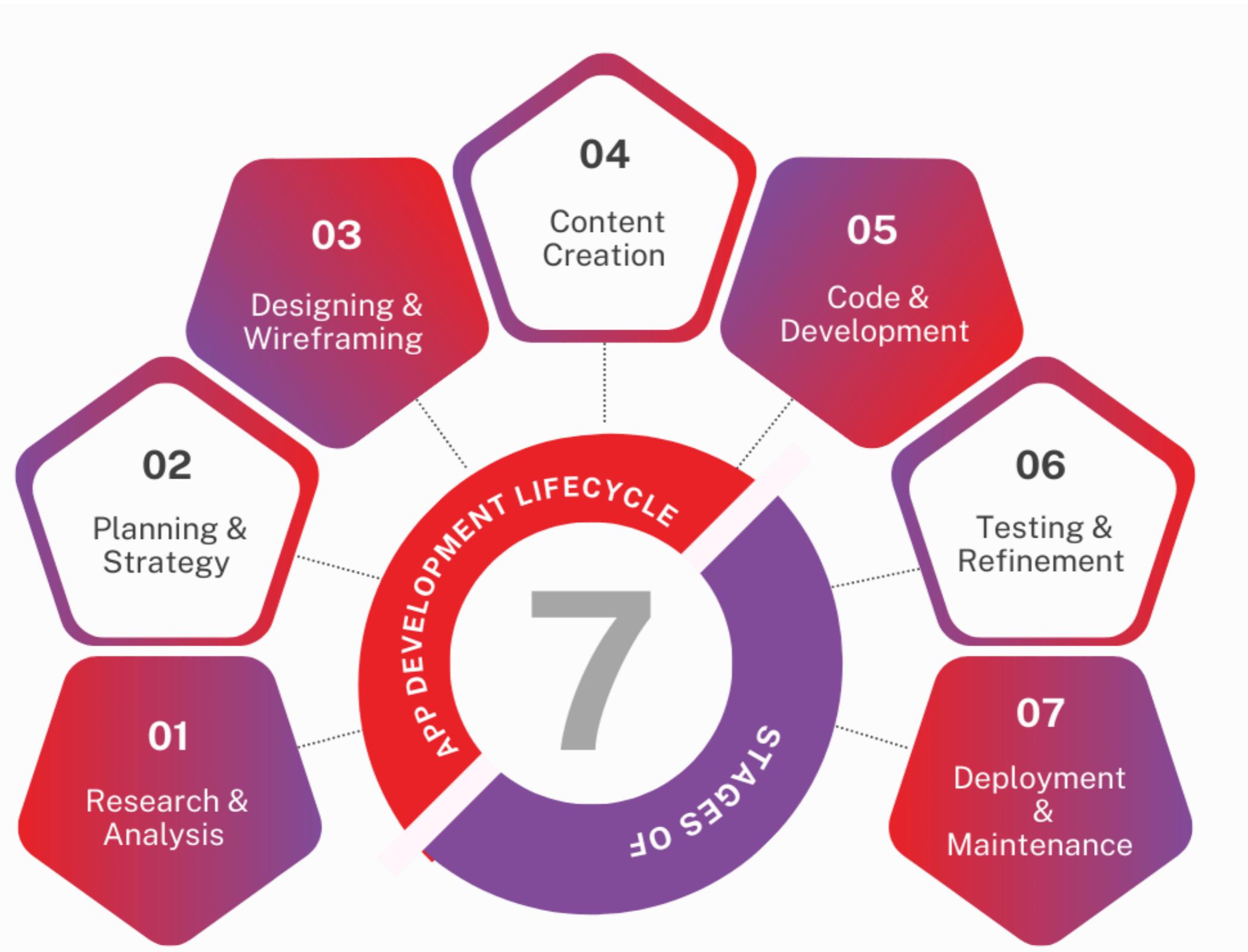
client side

server side

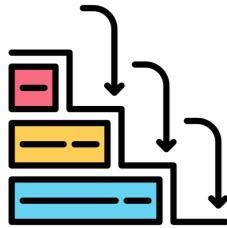
What is Web Application Development?

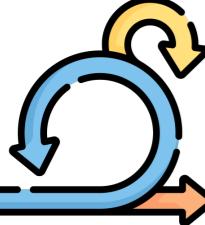


What is Web Application Development?



Web Application Development Methodologies



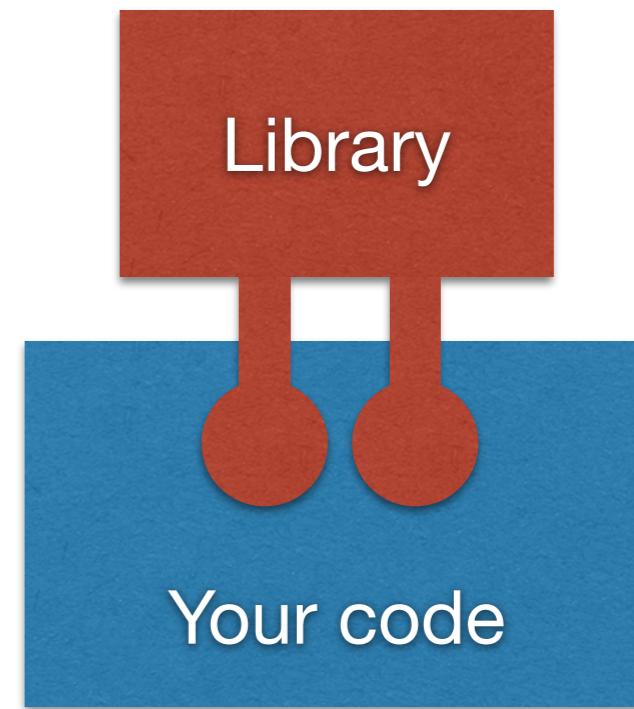
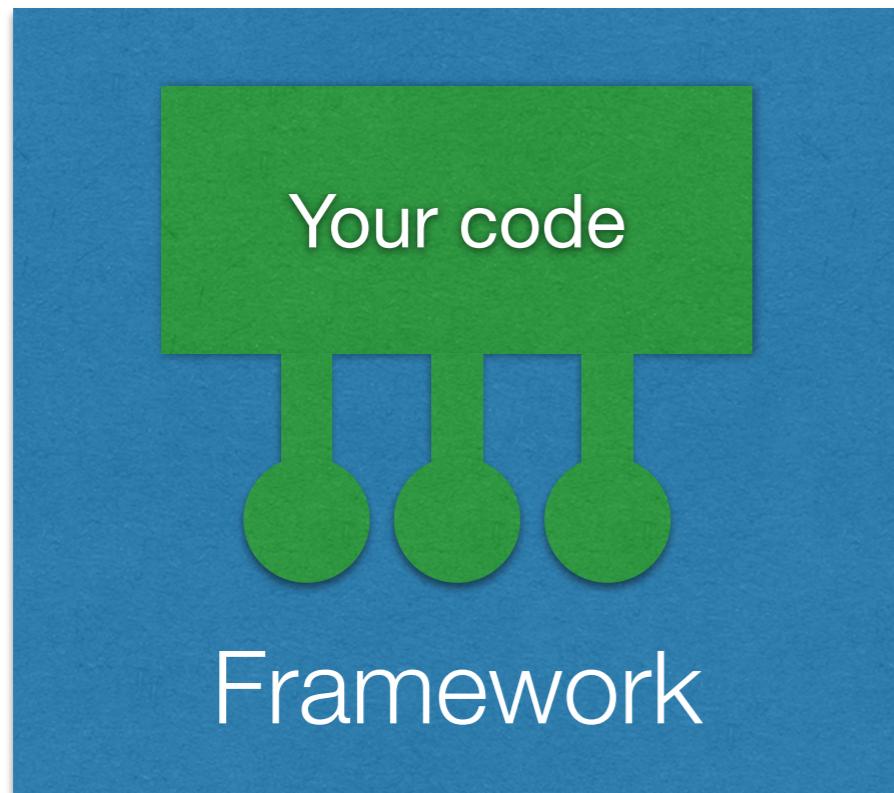
1. **Waterfall:** A sequential process where each phase (requirements, design, development, testing) is completed before moving to the next. Best for fixed, well-defined projects.
2. **Agile:** An iterative and flexible approach emphasizing collaboration, regular feedback, and incremental deliveries. Ideal for evolving requirements.
3. **Scrum:** A framework within Agile using sprints (short, time-boxed iterations) and daily meetings to ensure progress and adaptability.
4. **Spiral:** Combines iterative and waterfall methods, emphasizing risk analysis and gradual refinement. Suitable for high-risk projects.
5. ...

Technologies

Frontend technologies

	Angular	React	Vue.js	Next.js	Svelte
Type	Framework	Library	Framework	Meta-framework	Compiler
Language	TypeScript	JS / TS	JS / TS	JS / TS	JS / TS
Learning Curve	Steep	Moderate	Easy	Moderate	Easy
Size	~130 KB	~45 KB	~35 KB	~90 KB	~2 KB
State Management	Built-in (Signals)	External (Zustand, Jotai)	Built-in (Pinia)	React + Server State	Built-in (Runes)
Use Cases	Enterprise apps	Dynamic UIs	Rapid development	Full-stack apps	Performance-critical

Framework & Library

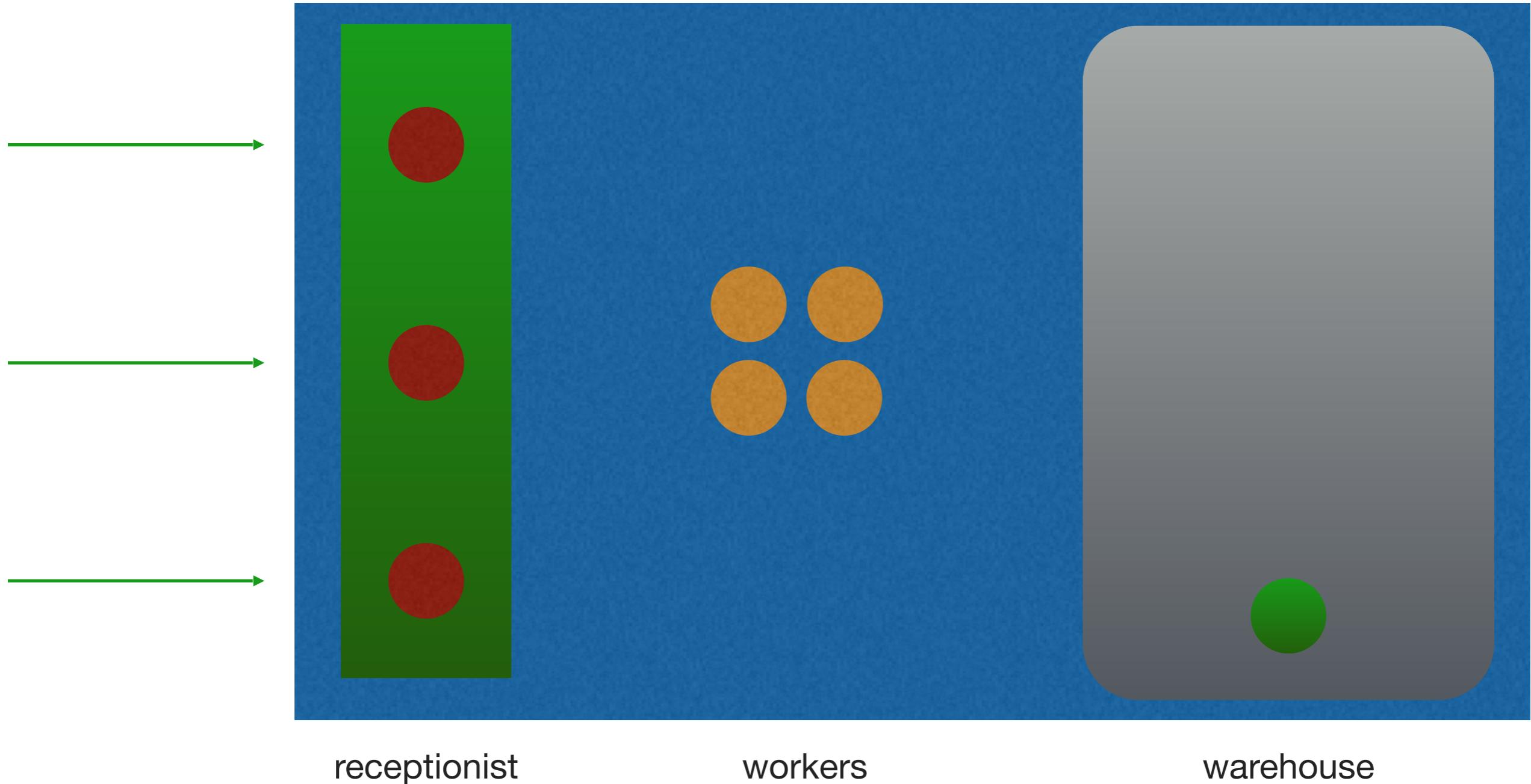


Backend technologies

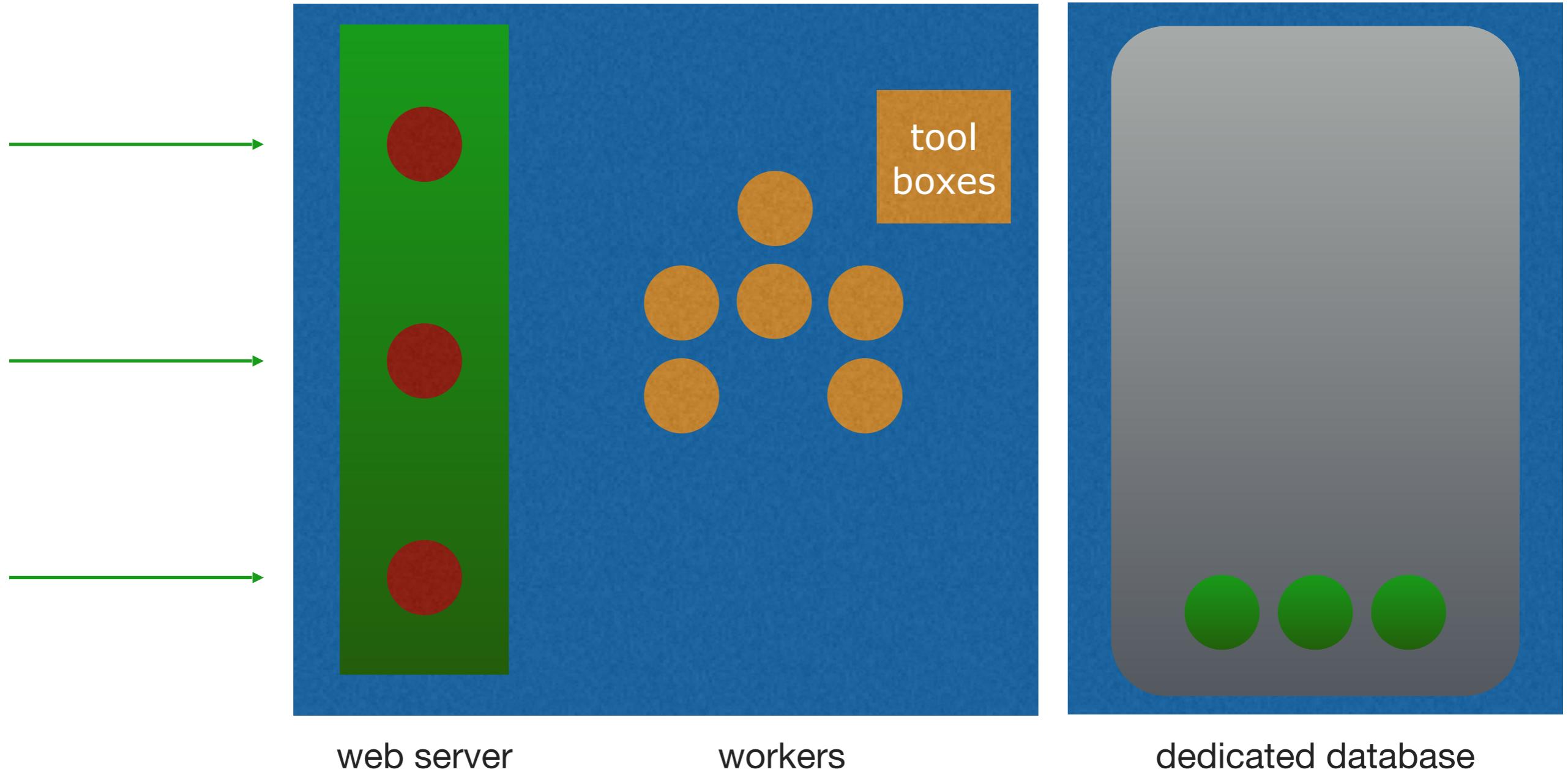
	Type	Language	Performance	Flexibility	Best For	Examples
Django	Full-stack	Python	High	Flexible	REST APIs, large apps	Instagram, Disqus
FastAPI	Async API	Python	Very High	Very flexible	Modern APIs, microservices	Netflix, Uber
Flask	Micro	Python	High	Very flexible	Prototypes, small apps	Pinterest, LinkedIn
Express.js	Minimalist	Node.js	Very High	Very flexible	APIs, real-time apps	PayPal, Uber
NestJS	Full-stack	Node.js/TS	Very High	Moderate	Enterprise Node apps	Adidas, Roche
Ruby on Rails	Full-stack	Ruby	High	Less flexible	E-commerce, MVPs	Shopify, GitHub
Spring Boot	Full-stack	Java	Very High	Moderate	Enterprise, banking	Many Fortune 500
Go (Gin/Fiber)	Minimalist	Go	Extremely High	Flexible	High-perf microservices	Google, Twitch

Scaling concepts

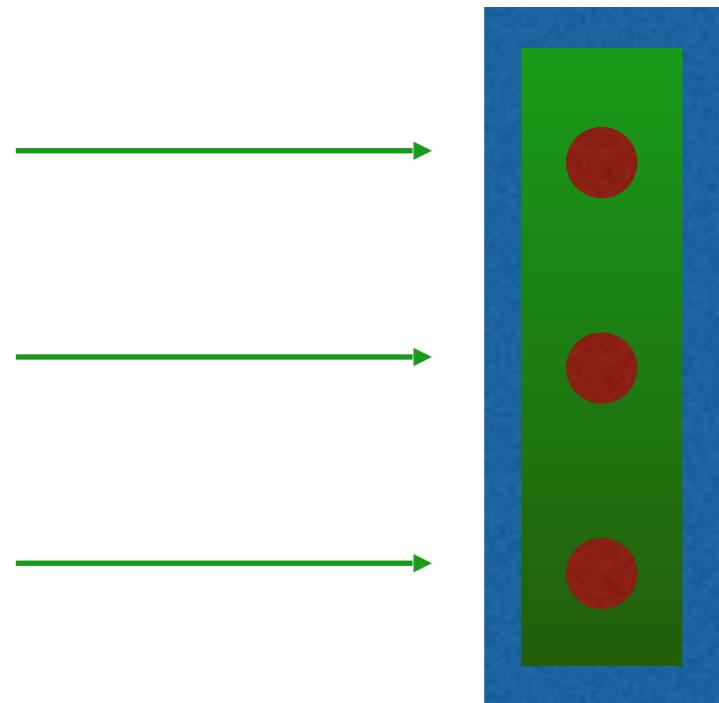
Furniture store



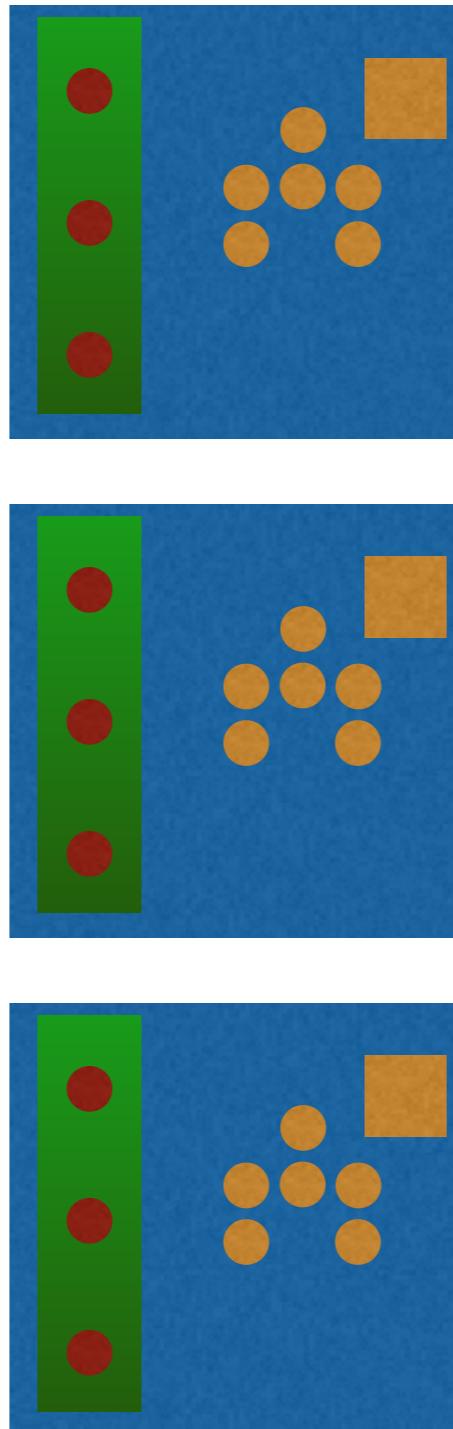
Separating the data store



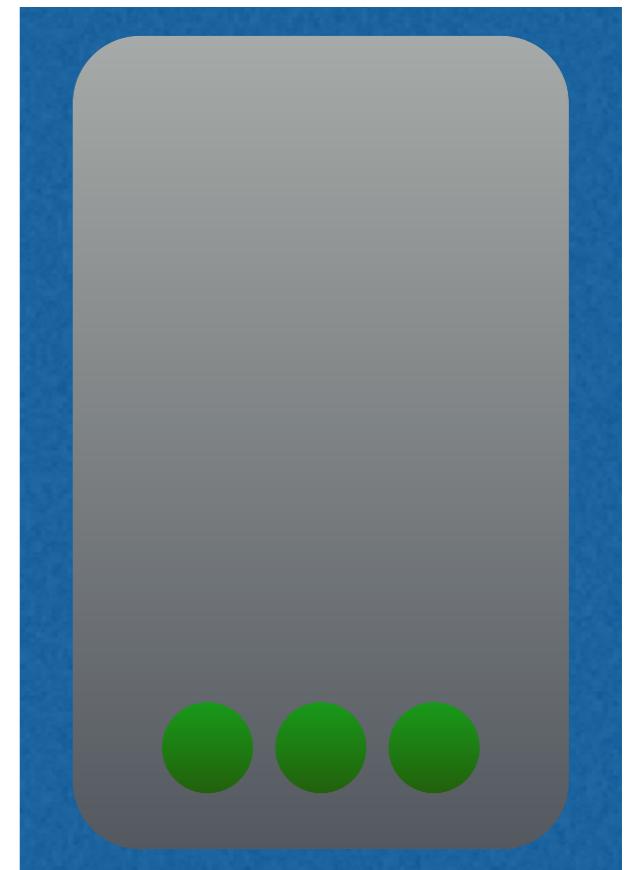
Scaling the server



main receptionists

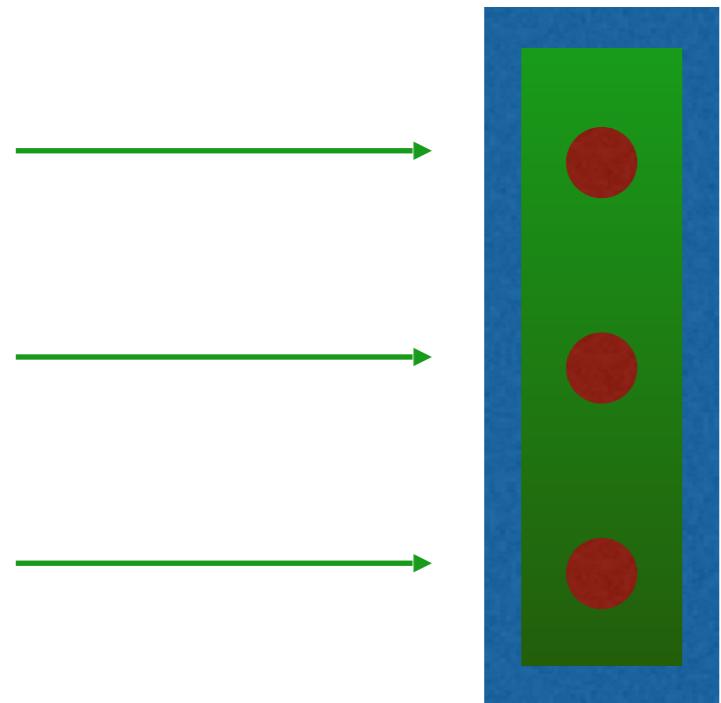


servers

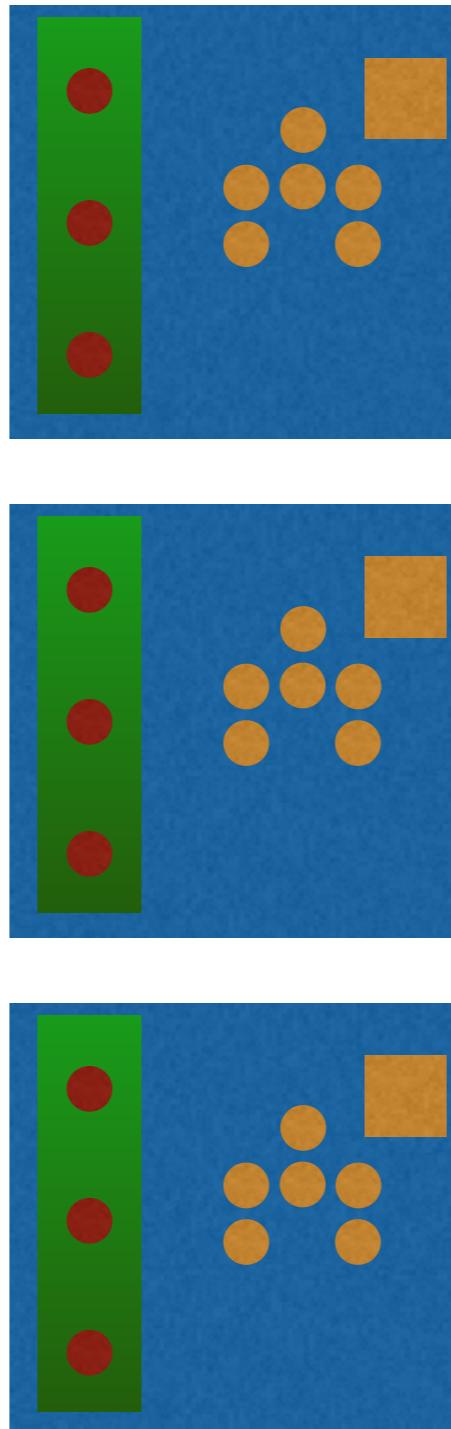


dedicated database

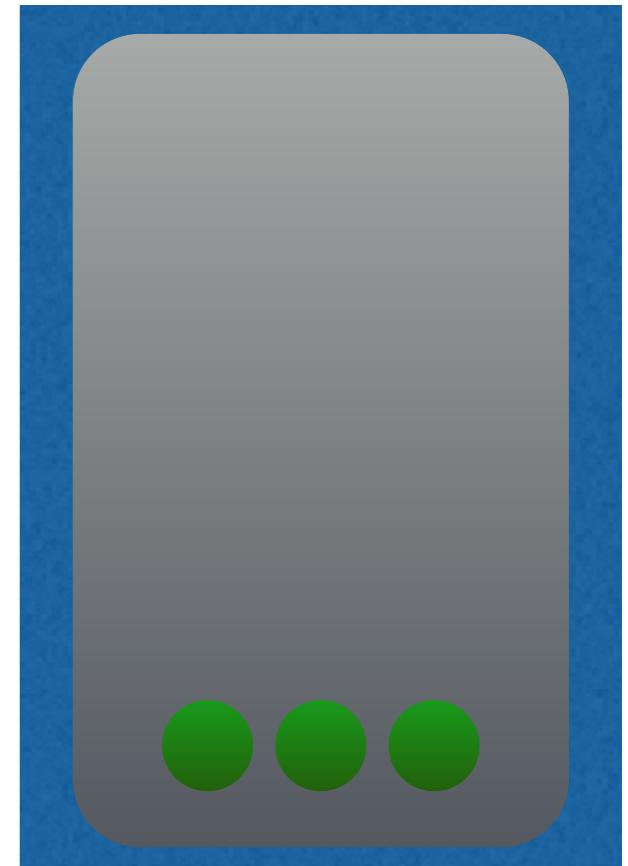
Scaling the server



load balancer

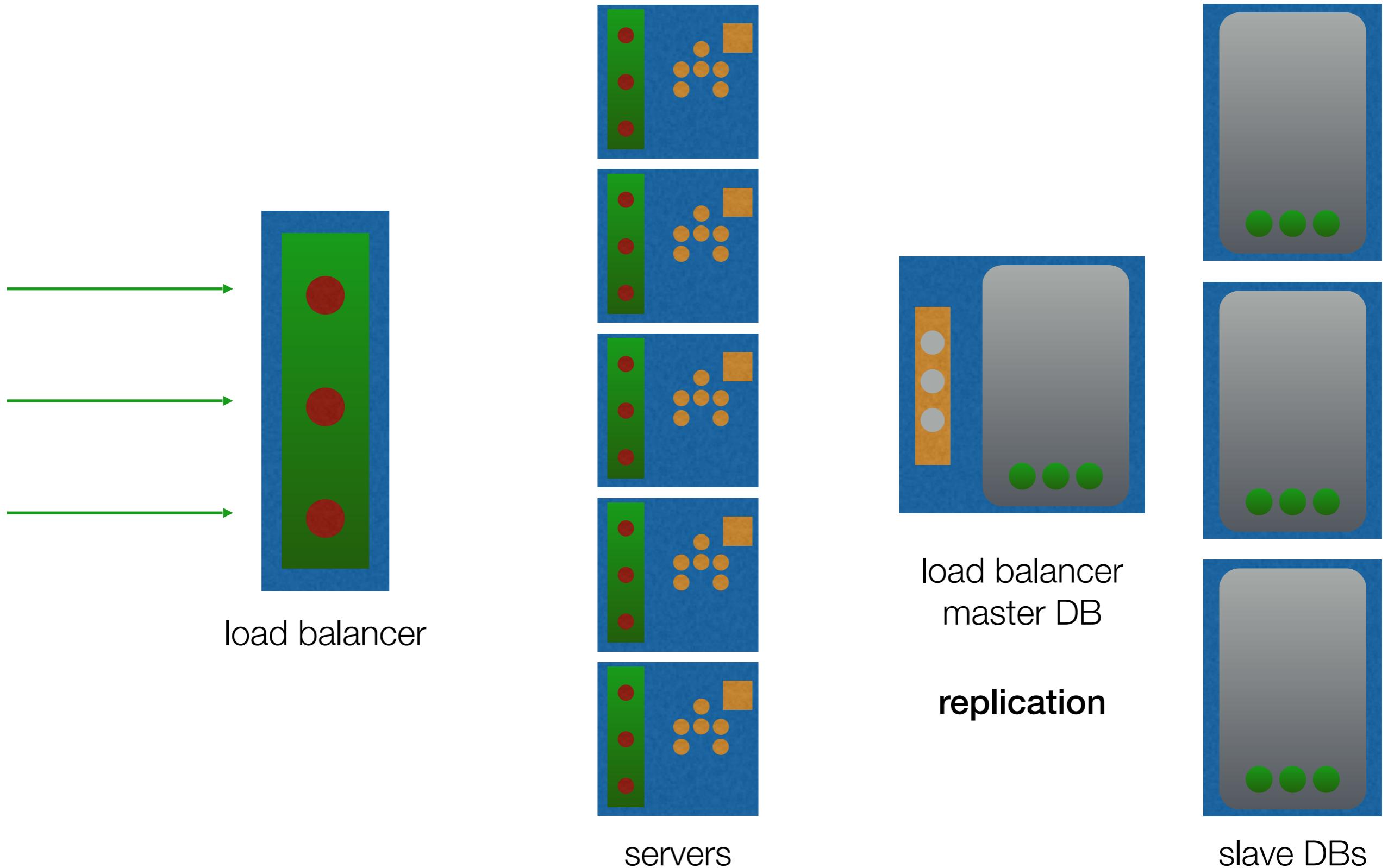


servers



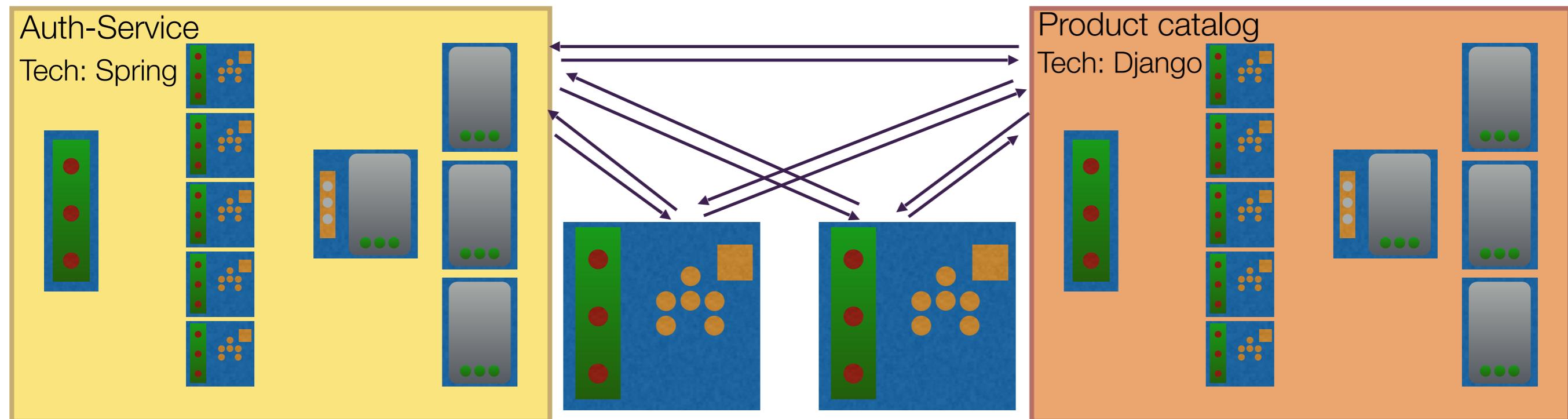
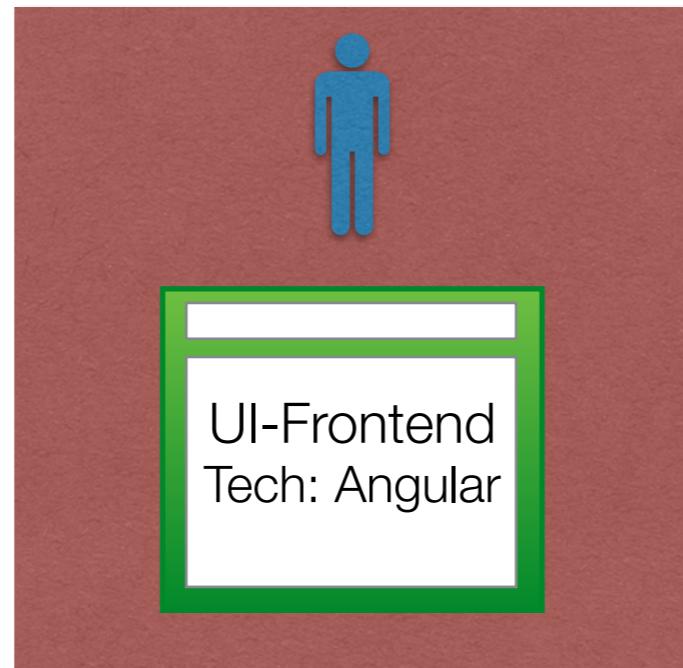
dedicated database

Scaling the data store



Modularization and APIs

Microservices



What is an API?



API — is like an artist performing on stage, and its users are the audience



RESTful API

1. REST (REpresentational State Transfer) – is an architectural style for developing web services
2. API (Application Program Interface) – is code that allows two software programs to communicate with each other

API endpoint for Companies

- 1. /getAllCompanies
- 2. /addNewCompany
- 3. /showCompanyDetail?id=23
- 4. /deleteCompany?id=23



The URL is a sentence, where resources are nouns and HTTP methods are verbs.

- | | |
|------------------|----------|
| 1. /companies | (GET) |
| 2. /companies | (POST) |
| 3. /companies/23 | (GET) |
| 4. /companies/23 | (DELETE) |



Data formats

- JSON

```
{  
  "root": {  
    "age": "18",  
    "isStudent": "true",  
    "name": "Nick"  
  }  
}
```

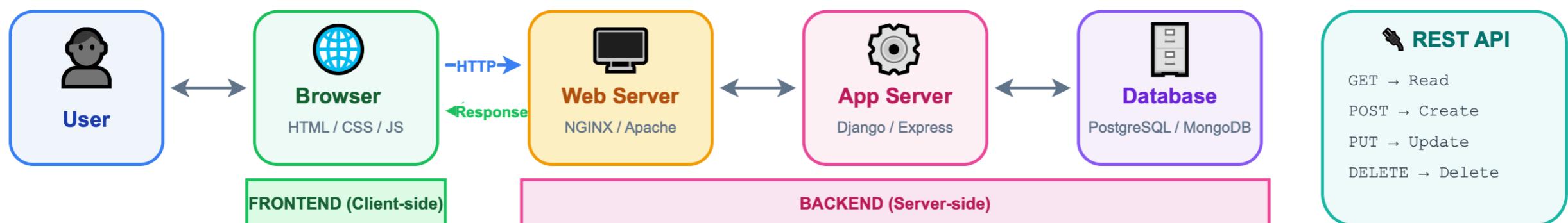
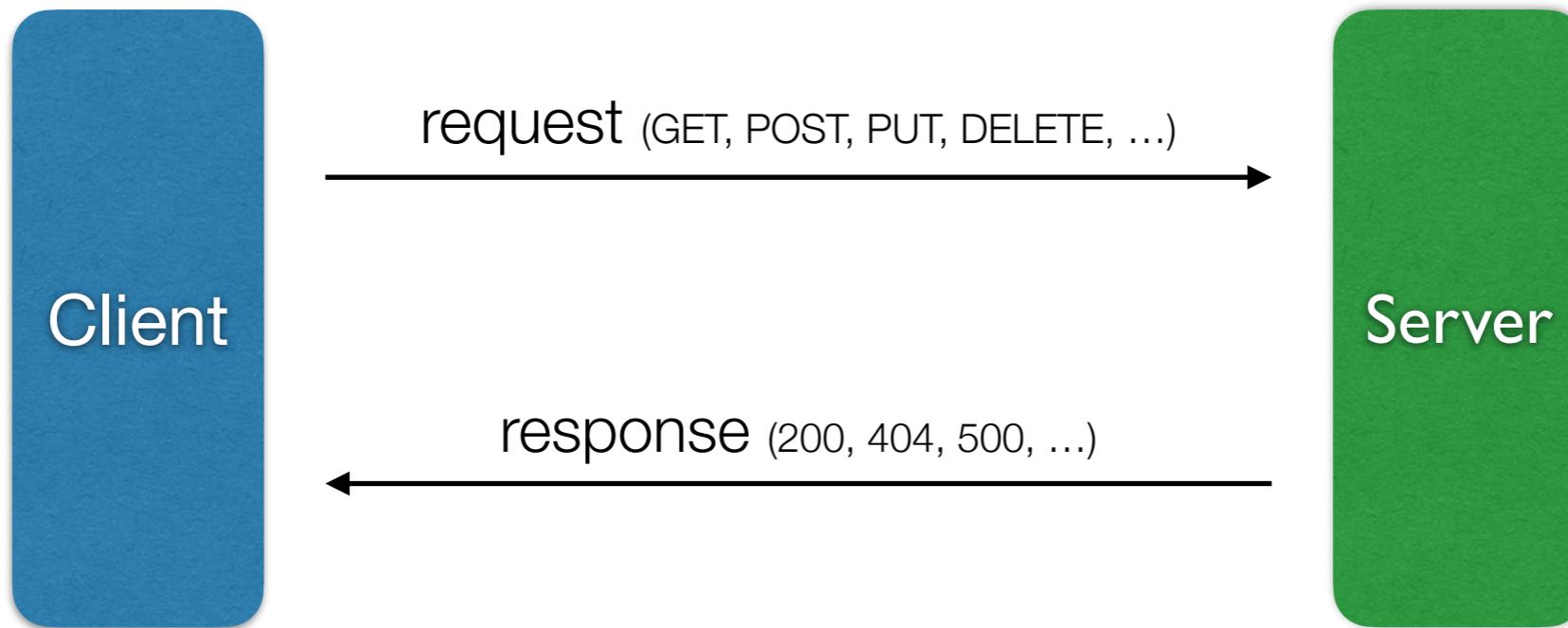
- XML

```
<?xml version="1.0" encoding="UTF-8"?>  
<root>  
  <age>18</age>  
  <isStudent>true</isStudent>  
  <name>Nick</name>  
</root>
```

- CSV

name	age	isStudent
Nick	18	true

Client Server Communication

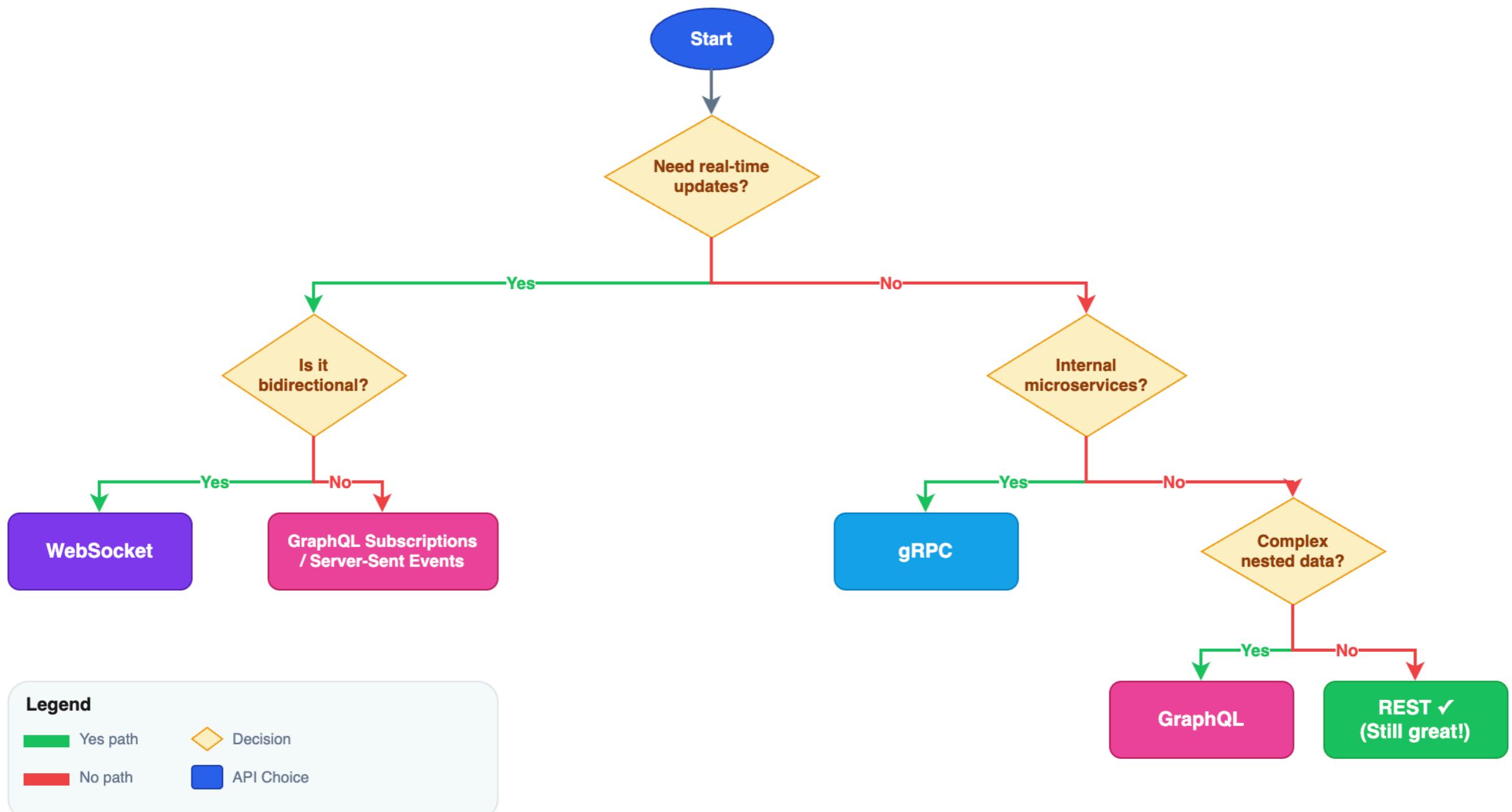


Beyond REST: Modern API Approaches

API Style	Protocol	Data Format	Best For	Real-time	Examples
REST	HTTP	JSON/XML	General purpose, CRUD ops	No (polling)	Most web apps
GraphQL	HTTP	JSON	Complex data, mobile apps	Yes (subscriptions)	Facebook, GitHub, Shopify
gRPC	HTTP/2	Protocol Buffers	Microservices, internal APIs	Yes (streaming)	Google, Netflix, Slack
WebSocket	WS/WSS	Any	Real-time, bidirectional	Yes (native)	Chat apps, trading, gaming

Beyond REST: Modern API Approaches

When to Use What: API Decision Tree



Protocols

- TCP/IP – Transmission Control Protocol / Internet Protocol
 - communication among computers on Internet
- HTTP – Hyper Text Transfer Protocol
 - Communicates with browsers to send web page packets
- HTTPS – Hyper Text Transfer Protocol Secure
 - HTTP with Secure Sockets Layer (SSL)
- FTP – File Transfer Protocol
 - Used by FTP Clients to transfer file packets

HTTP response status codes

- 2xx – Success category
 - 200 Ok
 - 201 Created
- 3xx – Redirection Category
 - 304 Not Modified
- 4xx – Client Error Category
 - 400 Bad Request
 - 401 Unauthorized
 - 403 Forbidden
 - 404 Not Found
- 5xx – Server Error Category
 - 500 Internal Server Error
 - 503 Service Unavailable

Summary

Topic	What You Learned
Web Application	Software running on a server, accessed via browser — no installation needed
Client-Server Model	Browser (client) sends requests → Server processes → Returns response
Frontend	What users see — HTML (structure), CSS (style), JavaScript (interactivity)
Backend	Server logic — Python/Django, Node/Express, Java/Spring + Database
Full-Stack	Frontend + Backend + Database combined
Scaling	Load balancers distribute traffic; DB replication handles data growth
REST API	Standardized way for systems to communicate using HTTP methods

Questions?