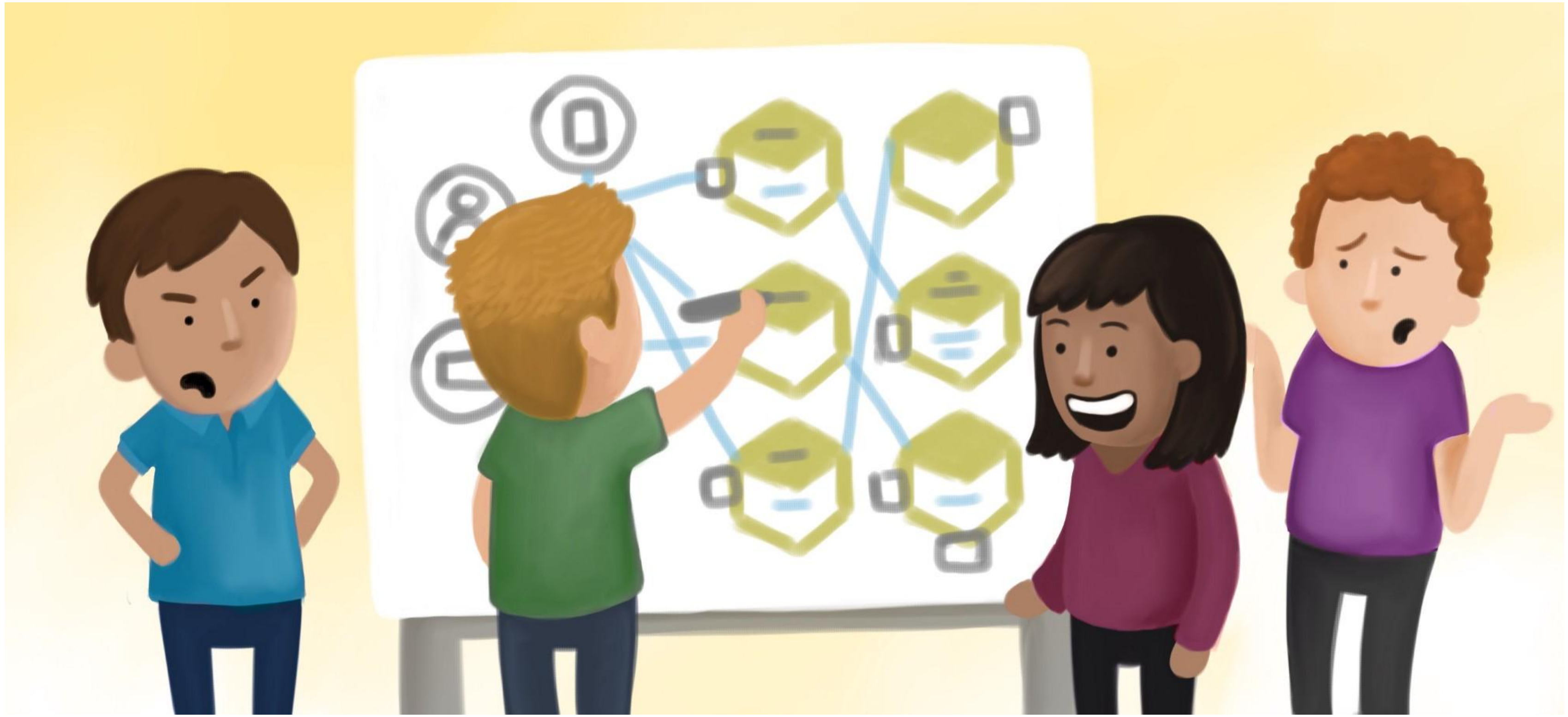
Node.js: Microservices and Testing

Vitalii Melnychuk
Software Engineer at Wise Engineering

The base problem is misunderstanding



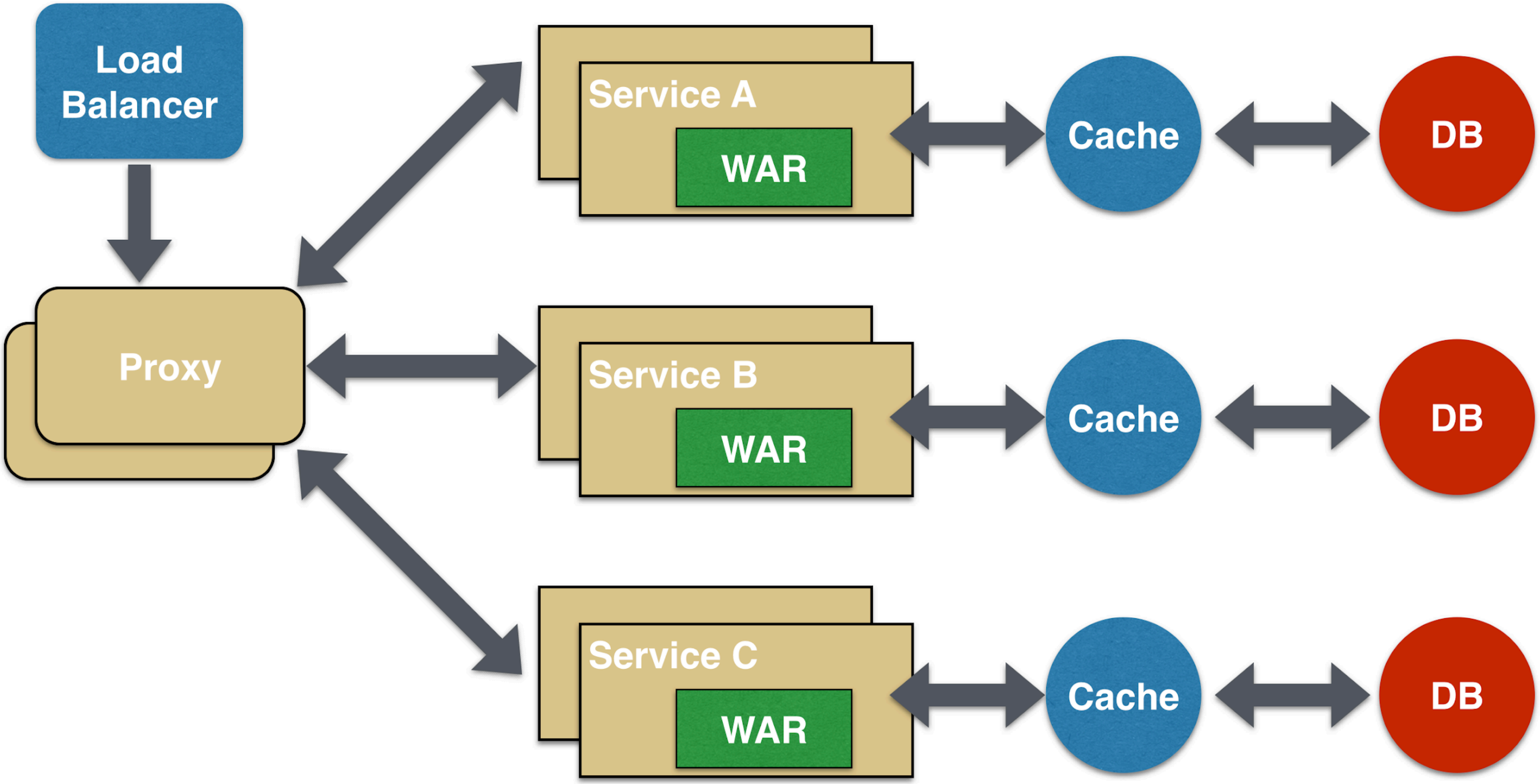
Where we are going on?

- Microservices, advantages and disadvantages
- How to test communication between Microservices

To Microservices - the cause of and solution to all of life’s problems

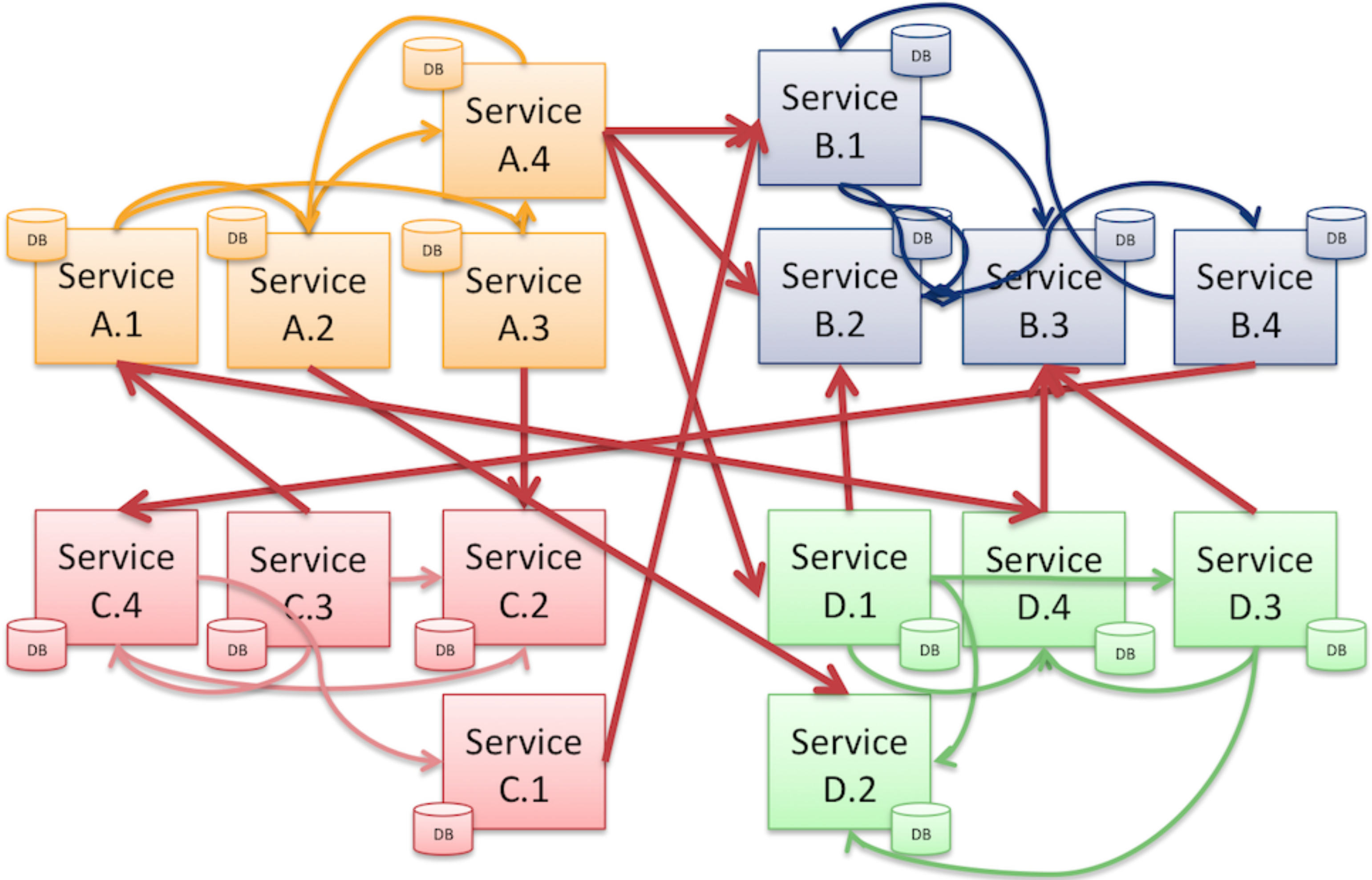
One source - a lot of Microservices

How to get all usages of the column?


Load Balancer
Proxy
Service A
WAR
Cache
DB
Service B
WAR
Cache
DB
Service C
WAR
Cache
DB

Undocumentable applications

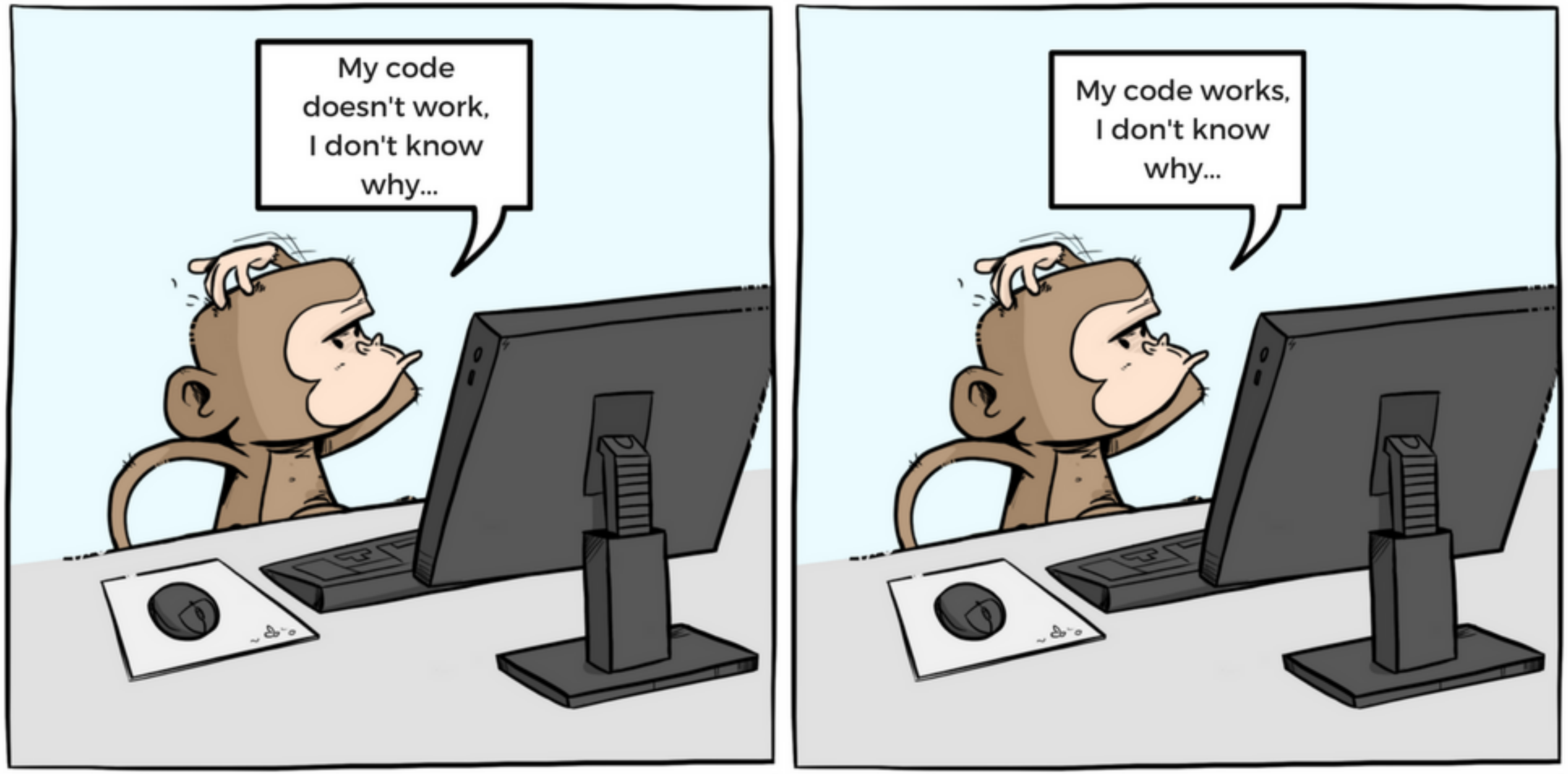
Do you think about people who will be working with it?



Service A.4
DB
Service A.1
DB
Service A.2
DB
Service A.3
DB
Service B.1
DB
Service B.2
DB
Service B.3
DB
Service B.4
DB
Service C.4
DB
Service C.3
DB
Service C.2
DB
Service C.1
DB
Service D.1
DB
Service D.4
DB
Service D.3
DB
Service D.2
DB

Tests Coverage is 0% =(

Who cares?


My code doesn't work, I don't know why...
My code works, I don't know why...

How to solve?

Best approaches to build your own applications

- Documentation -> tests -> code
- One source -> one MS
- MS API

Why Microservice API is important?

- Validate data then process
- Each module of application is controllable
- Will be able to test part of application

Disadvantages:

- For first time, It’s too boring
- Not useful for startups
- Deployment process is complex

Advantages:

- Easy to understand
- Easy to support
- Scalable

So, how it would be on the real project?

Create our first Microservice

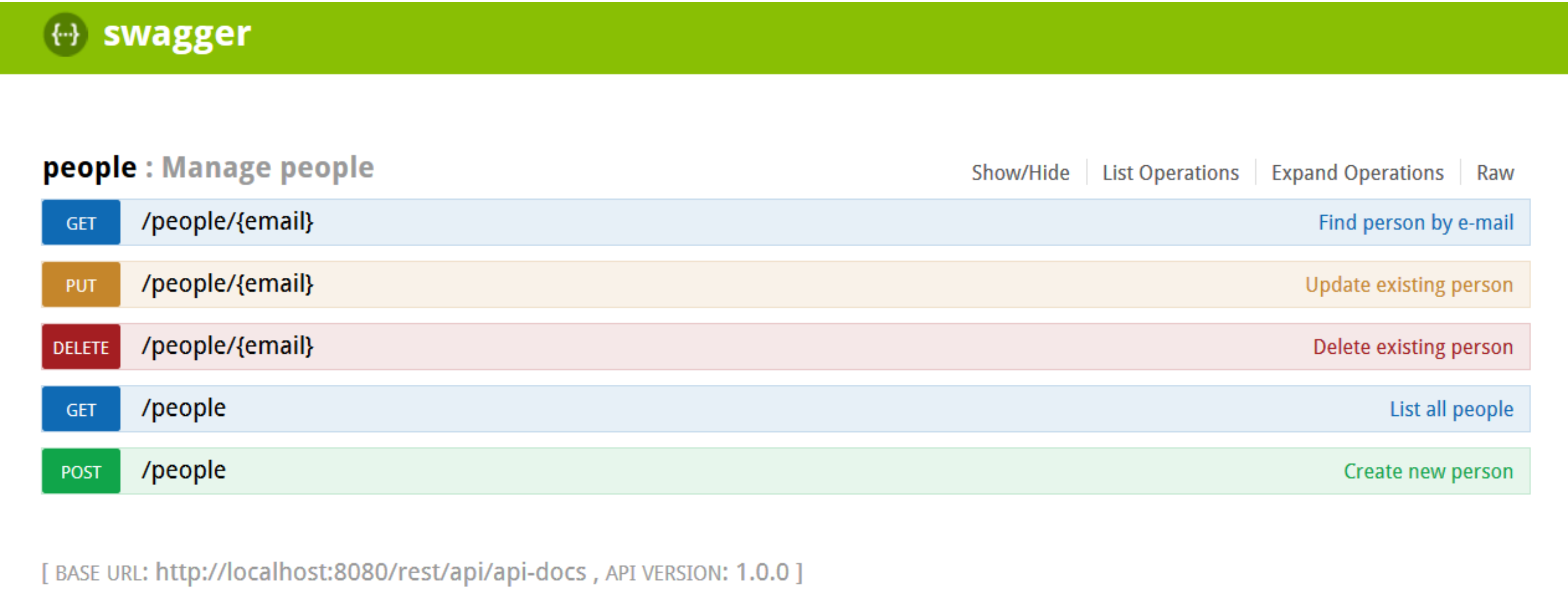
What were your first steps after getting the idea?

1. What we have to expect?
2. Describe API schema
3. Describe each particular module
4. Write tests
5. Write code

What we have to expect?

- Count of requests
- Request params
- Response params

Swagger as tool



people : Manage people

GET	/people/{email}	Find person by e-mail
PUT	/people/{email}	Update existing person
DELETE	/people/{email}	Delete existing person
GET	/people	List all people
POST	/people	Create new person

[BASE URL: http://localhost:8080/rest/api/api-docs , API VERSION: 1.0.0]

Describe API schema

- Use the same objects for validation, docs, tests (jsonschema)
- 100% tests coverage

Describe each particular module

- Create folders structure
- Describe components that will be used
- Provide feature flow of module
- Send on team review

Writing tests

Main difficulties:

- I have a remote connection
- There are methods like \`setInterval\`, \`setTimeout\`
- Emm, console.log()?

Mocks are your friends

- Use DAO as pattern to work with db
- Pass all dependencies to your module from outside
- Don’t use global variables
- Create stubs and test only part of application

Is it time to write code?

If have Microservice and don’t have any tests or docs. You do have nothing.

- To think about people who will be working with you in the feature
- To edit documentation easier than code
- To think twice when you start new application with a lot of microservices