

Tutorial - Quantum Web Services: development and deployment

Introduction

Javier Romero-Álvarez
Jaime Alvarado-Valiente
Enrique Moguel
José García-Alonso



JUNTA DE EXTREMADURA
Consejería de Economía, Ciencia y Agenda Digital

SPILab
by Quercus SEG

Software Engineering Group
QUERCUS
UNIVERSIDAD DE EXTREMADURA

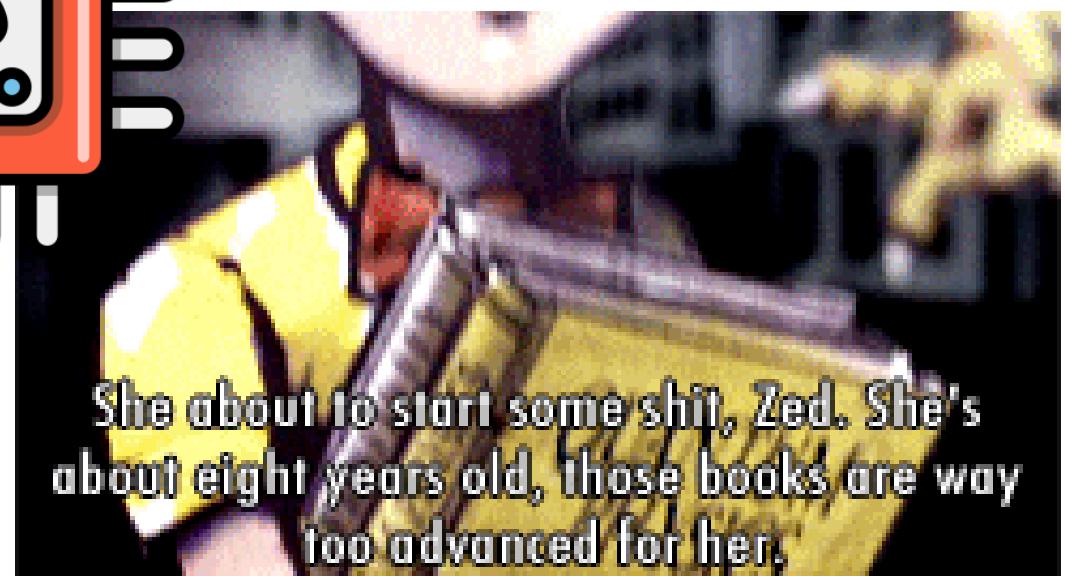
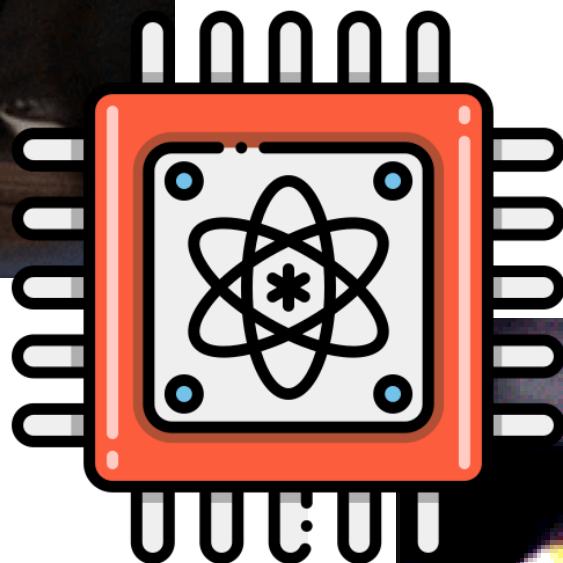
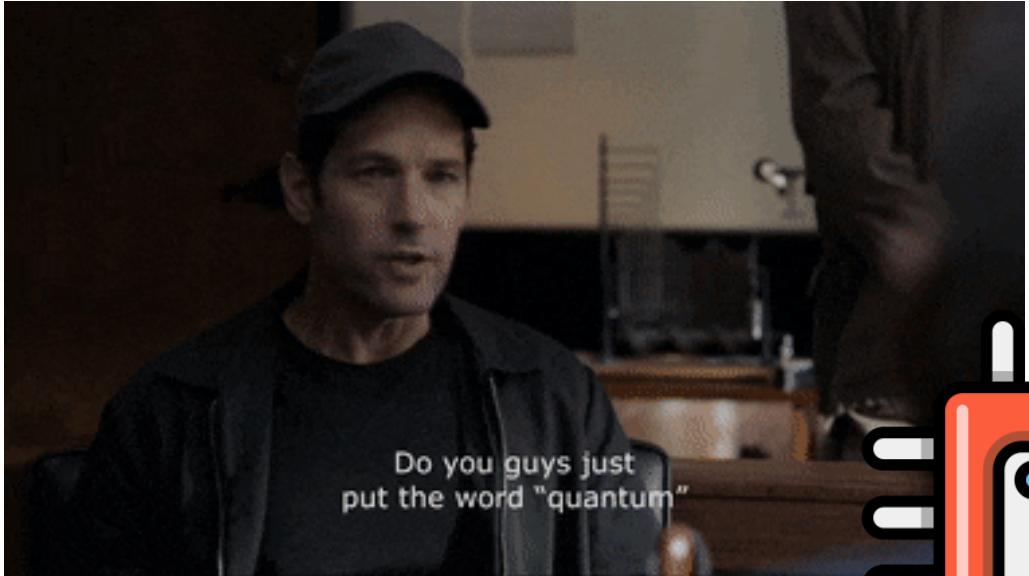
Q
EX
UNIVERSIDAD DE EXTREMADURA



OUTLINE

1. Introduction to quantum computing
2. Quantum Programming
3. Quantum Web Services with AWS
4. Shor Service

INTRODUCTION TO QUANTUM COMPUTING



INTRODUCTION TO QUANTUM COMPUTING



1900s

- Einstein
- Planck
- Bohr
- Feynman
- Schrödinger
- ...

INTRODUCTION TO QUANTUM COMPUTING



Theoretical
physics



→ Experimental physics

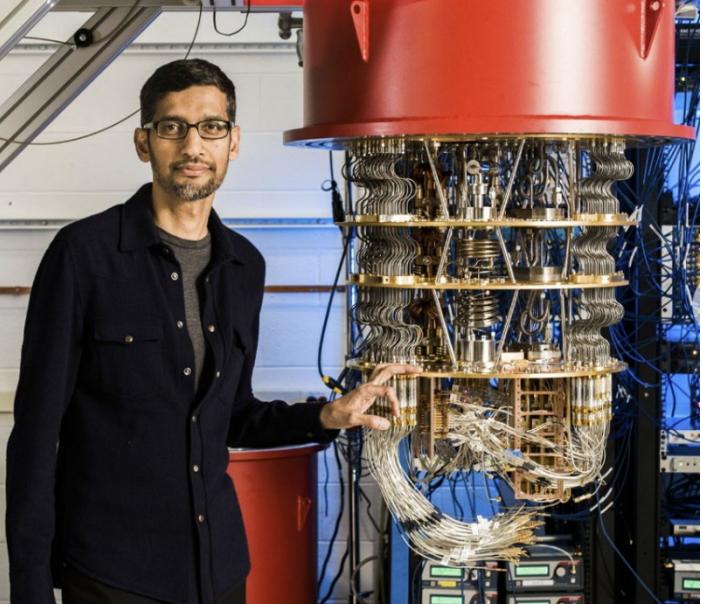


→ Engineering

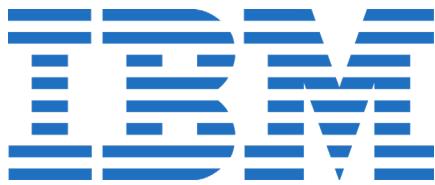
INTRODUCTION TO QUANTUM COMPUTING



- 1998 – Los Alamos National Laboratory – Massachusetts Institute of Technology
- First quantum computer
2 qubits
- Nuclear Magnetic Resonance



INTRODUCTION TO QUANTUM COMPUTING



Google

rigetti

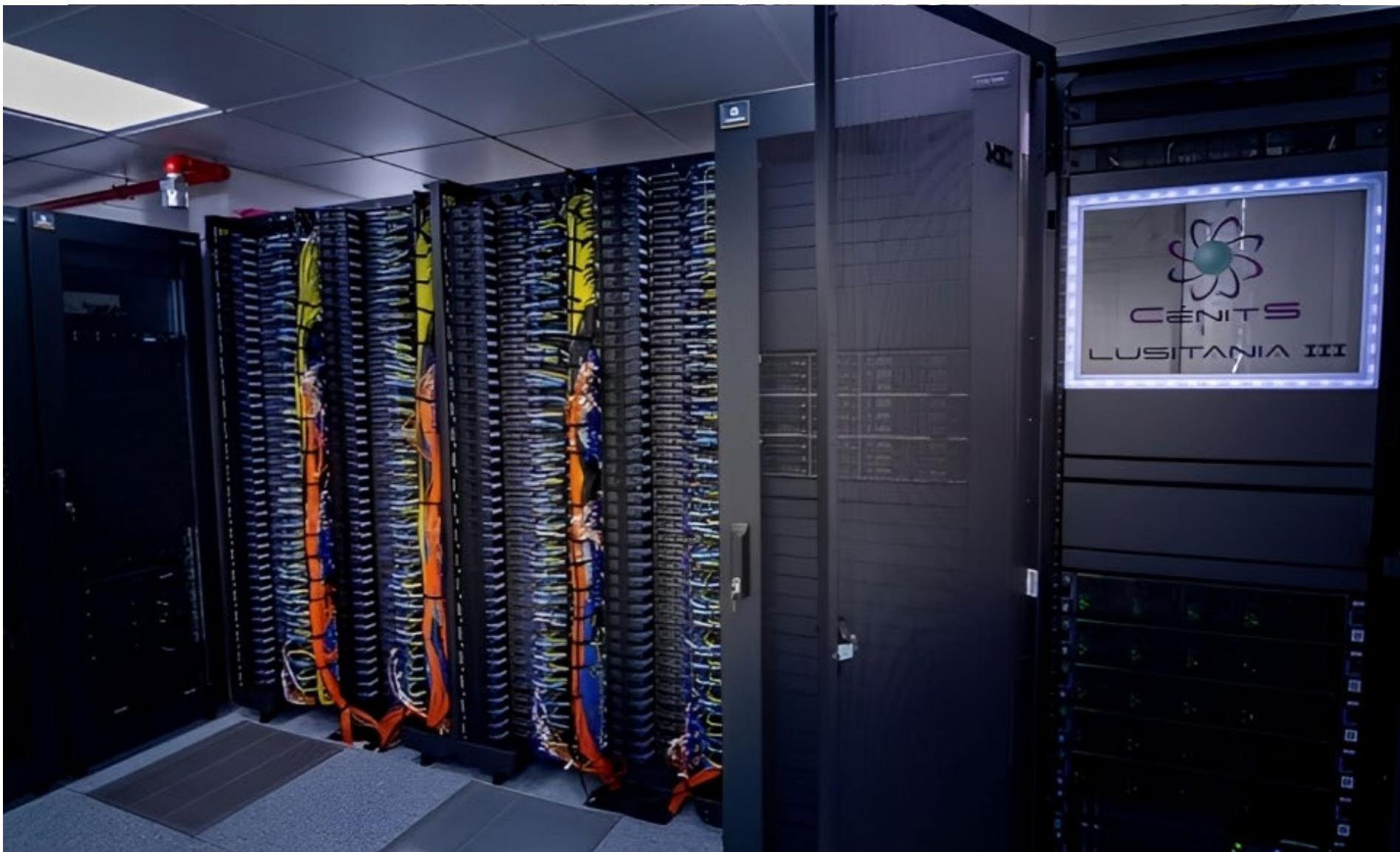


XANADU

IONQ

D-Wave
The Quantum Computing Company™

INTRODUCTION TO QUANTUM COMPUTING



Quantum Supremacy

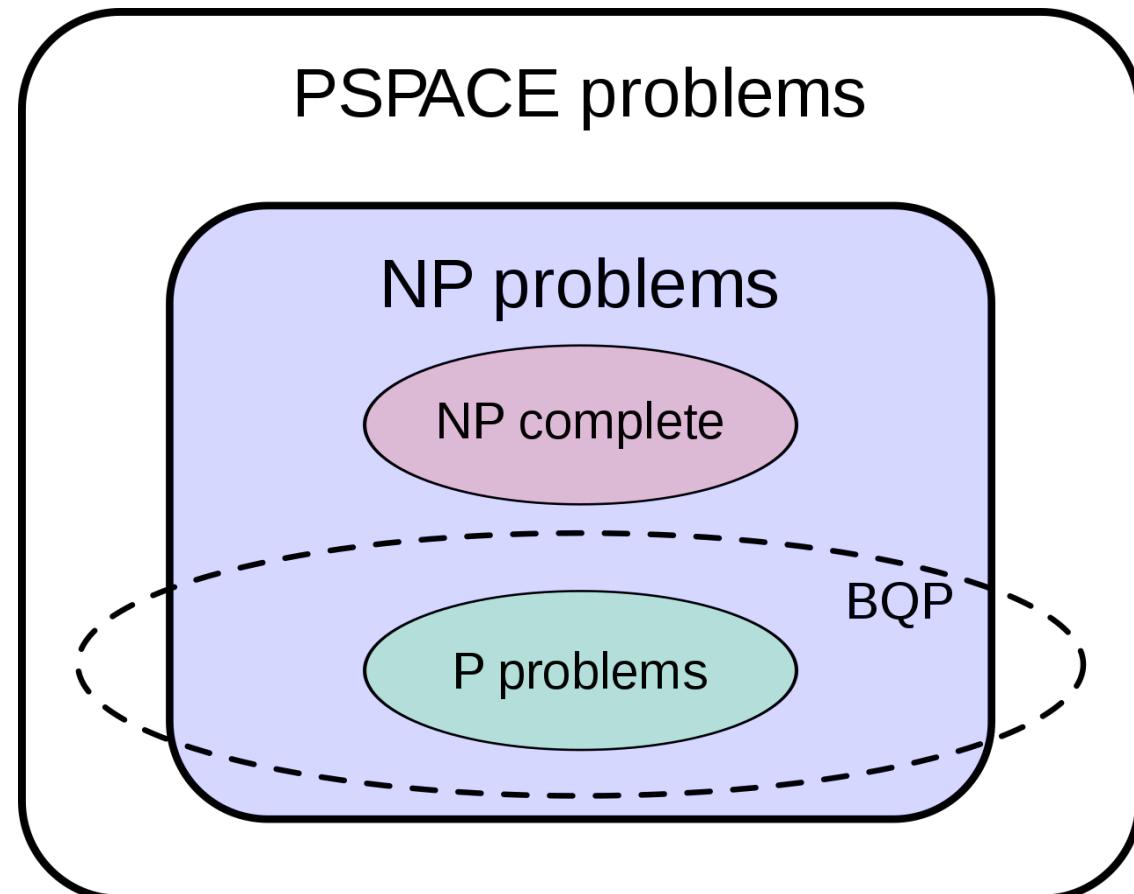
INTRODUCTION TO QUANTUM COMPUTING

The collage consists of three news snippets from different sources:

- Left Snippet:** A screenshot of The Times website showing a headline about Google's quantum computer. The headline reads "Google's quantum computer does a 10,000-year task in minutes". Below the headline is a photo of Sundar Pichai standing next to a large, complex quantum computing rig. The caption below the photo reads: "Sundar Pichai, Google's chief executive, with one of the company's quantum computers in the Santa Barbara lab. GOOGLE/GETTY IMAGES".
- Middle Snippet:** A screenshot of The Times website showing a headline about Chinese teams reaching quantum supremacy. The headline reads "Two Chinese teams claim to have reached primacy with quantum computers". Below the headline is a photo of a dense array of optical fibers and components, likely part of an optical quantum computer.
- Right Snippet:** A screenshot of Phys.org website showing a headline about Chinese teams reaching quantum supremacy. The headline is identical to the one in the middle snippet. Below the headline is a photo of a dense array of optical fibers and components, similar to the one in the middle snippet.

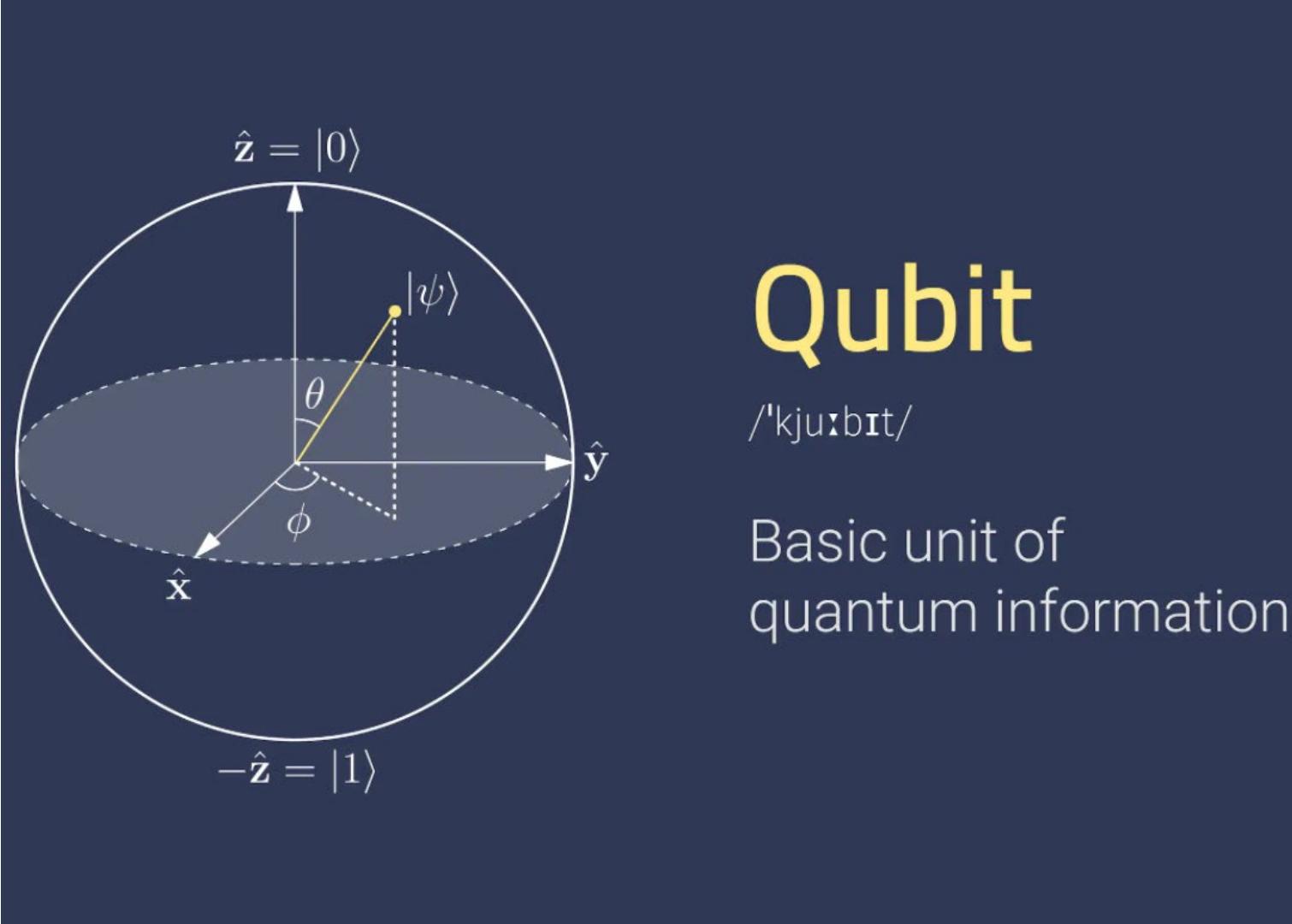
Computational complexity theory

Bounded-error Quantum Polynomial time (BQP) is the class of decision problems solvable by a quantum computer in polynomial time, with an error probability of at most 1/3 for all instances



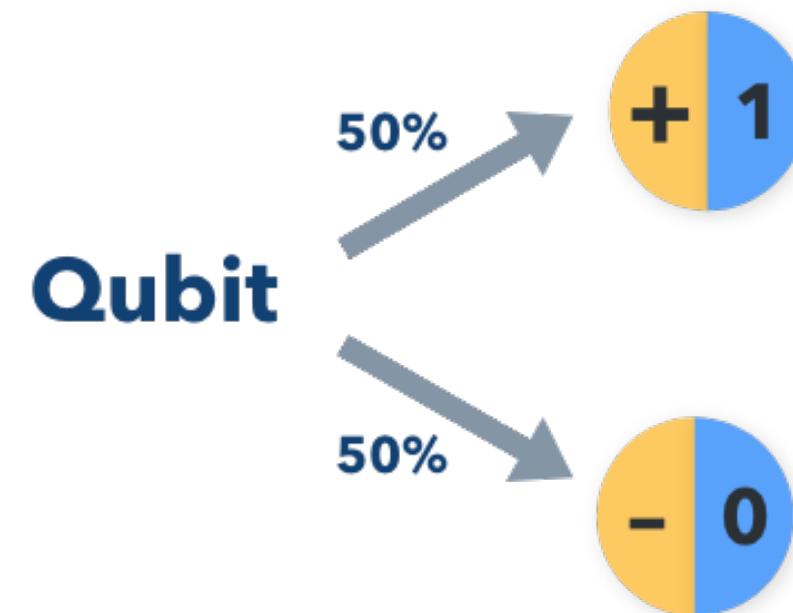
Qubit

INTRODUCTION TO QUANTUM COMPUTING



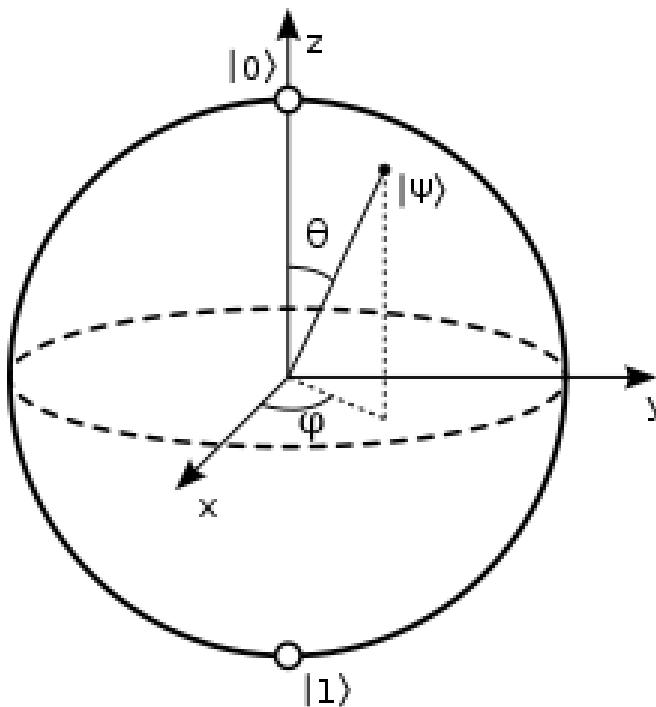
Qubit collapse

INTRODUCTION TO QUANTUM COMPUTING



Qubit representation

INTRODUCTION TO QUANTUM COMPUTING



$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \text{ and } |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, |01\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, |10\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \text{ and } |11\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

Qubit superposition

INTRODUCTION TO QUANTUM COMPUTING



SCHRÖDINGER
CAT



Public classical authenticated channel



Madrid's quantum communication network deploys its first phase

Researchers at the Center for Computational Simulation at the Polytechnic University of Madrid are coordinating the Madrid quantum communication network, a European benchmark. They are researching quantum algorithms that will increase the security of critical applications in fields such as telecommunications, health care, electricity supply and public services, among others.

22.03.2021



Quantum entanglement enables particles to affect each other instantaneously across any distance.



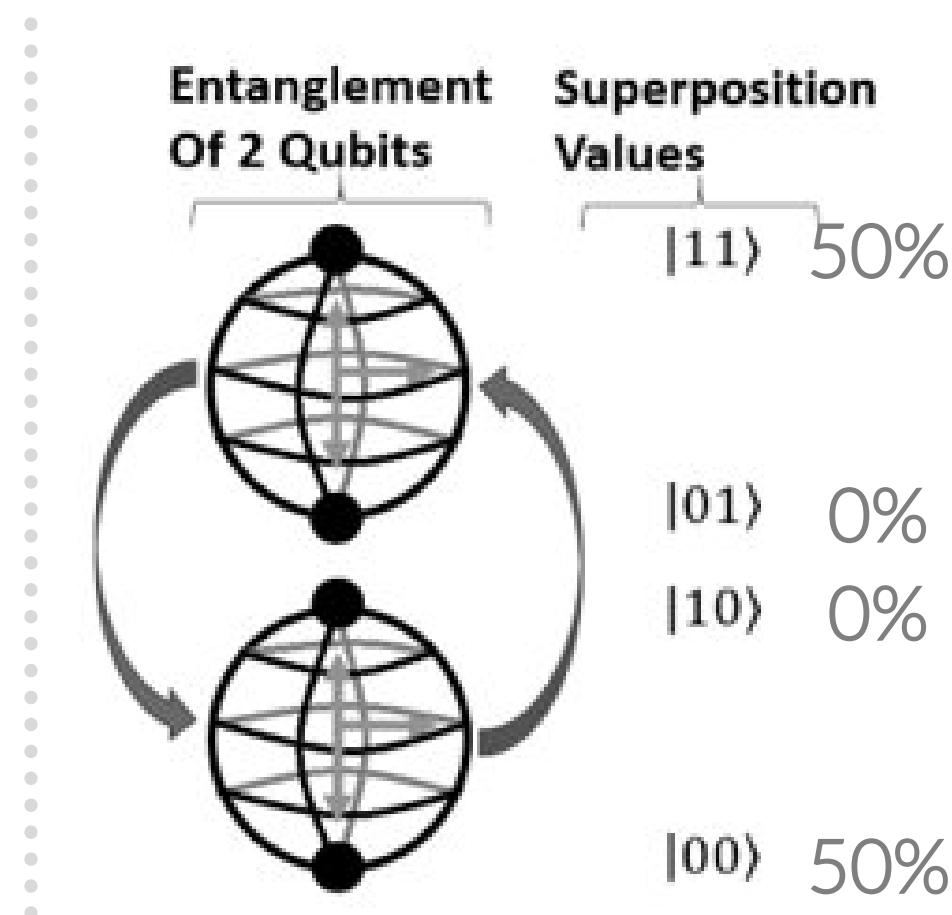
Entangled particles would remain “connected” even if they were on *opposite sides of the universe.*

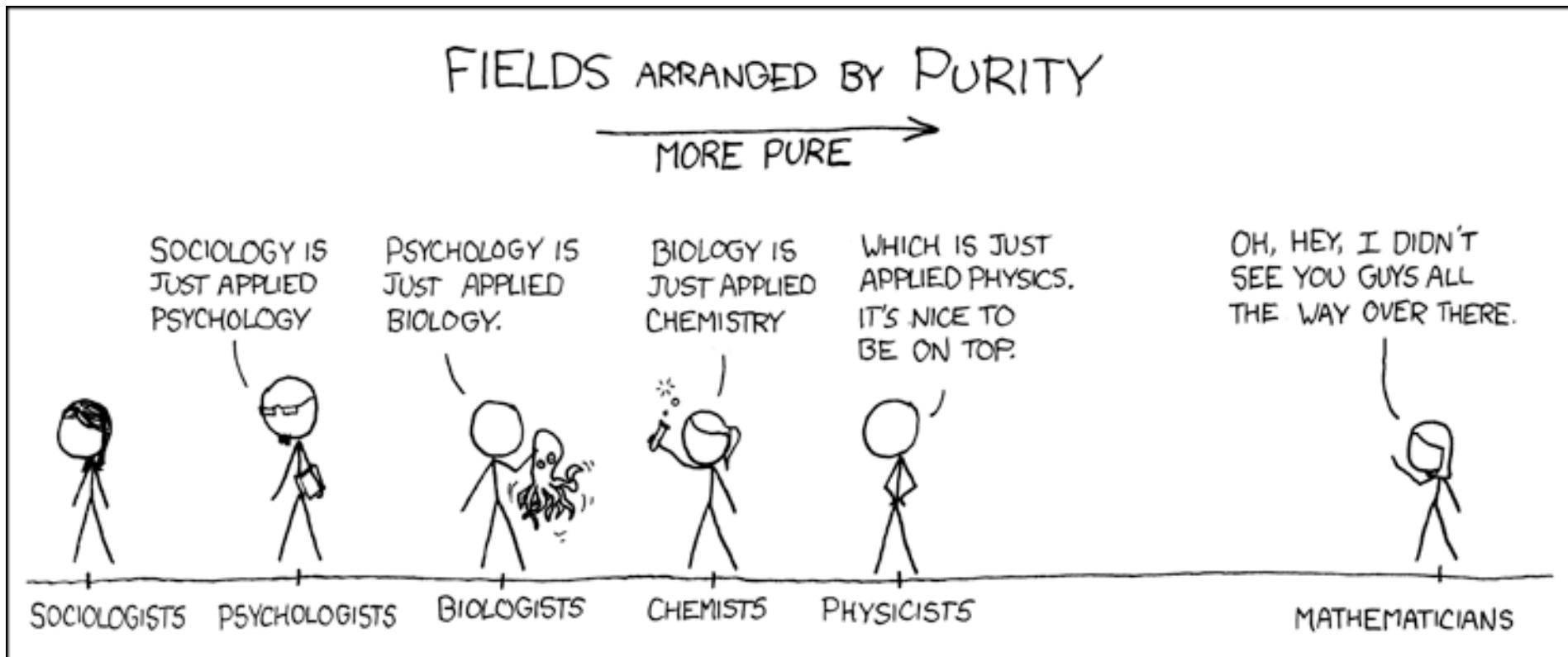
Source: Coma Niddy, PBS

Qubit entanglement

INTRODUCTION TO QUANTUM COMPUTING

- Quantum entanglement is the fundamental resource of Quantum Physics, which is necessary to generate, in order to achieve the Maximum Effect in Quantum Computing
- **It cannot occur in classical bits**
- The quantum state of (n) qubits involves the superposition of all possible configurations



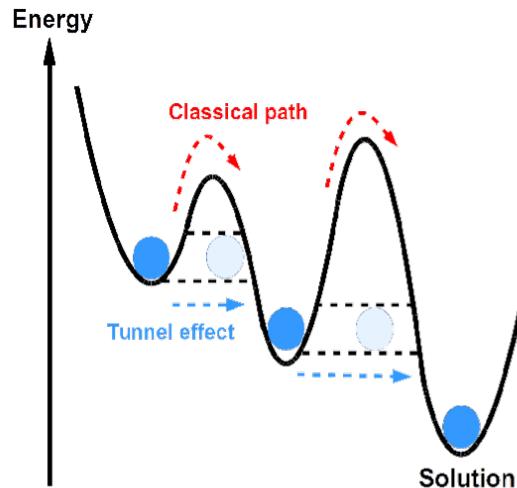




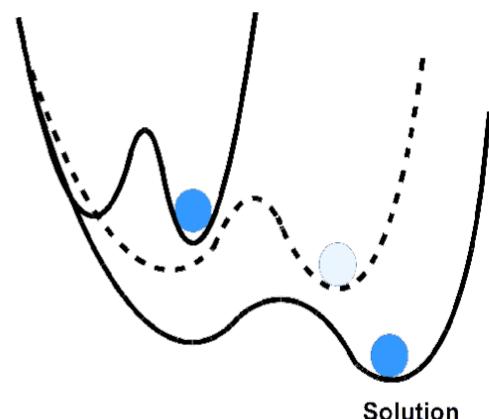
- [2 Algorithms based on the quantum Fourier transform](#)
 - [2.1 Deutsch–Jozsa algorithm](#)
 - [2.2 Bernstein–Vazirani algorithm](#)
 - [2.3 Simon's algorithm](#)
 - [2.4 Quantum phase estimation algorithm](#)
 - [2.5 Shor's algorithm](#)
 - [2.6 Hidden subgroup problem](#)
 - [2.7 Boson sampling problem](#)
 - [2.8 Estimating Gauss sums](#)
 - [2.9 Fourier fishing and Fourier checking](#)
- [3 Algorithms based on amplitude amplification](#)
 - [3.1 Grover's algorithm](#)
 - [3.2 Quantum counting](#)
- [4 Algorithms based on quantum walks](#)
 - [4.1 Element distinctness problem](#)
 - [4.2 Triangle-finding problem](#)
 - [4.3 Formula evaluation](#)
 - [4.4 Group commutativity](#)
- [5 BQP-complete problems](#)
 - [5.1 Computing knot invariants](#)
 - [5.2 Quantum simulation](#)
 - [5.3 Solving a linear systems of equations](#)
- [6 Hybrid quantum/classical algorithms](#)
 - [6.1 QAOA](#)
 - [6.2 Variational quantum eigensolver](#)

Types of programming

Quantum annealing (which also includes adiabatic quantum computation) is a quantum computing method used to find the optimal solution of problems involving many solutions



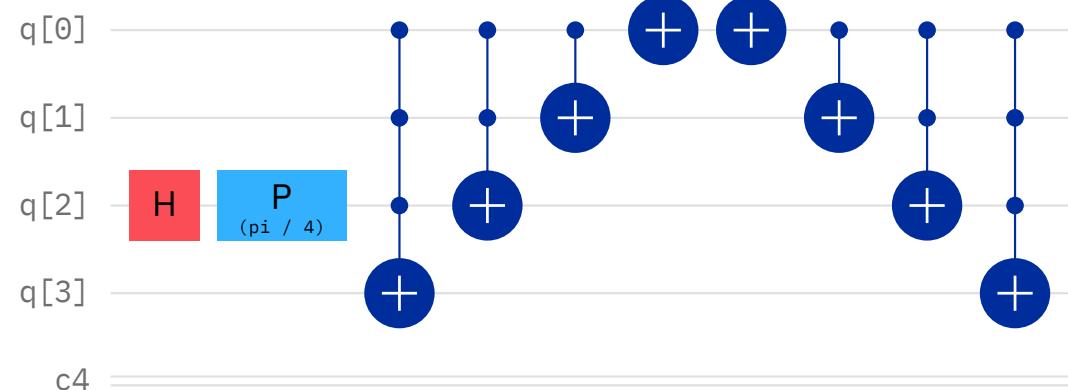
Quantum Tunnelling

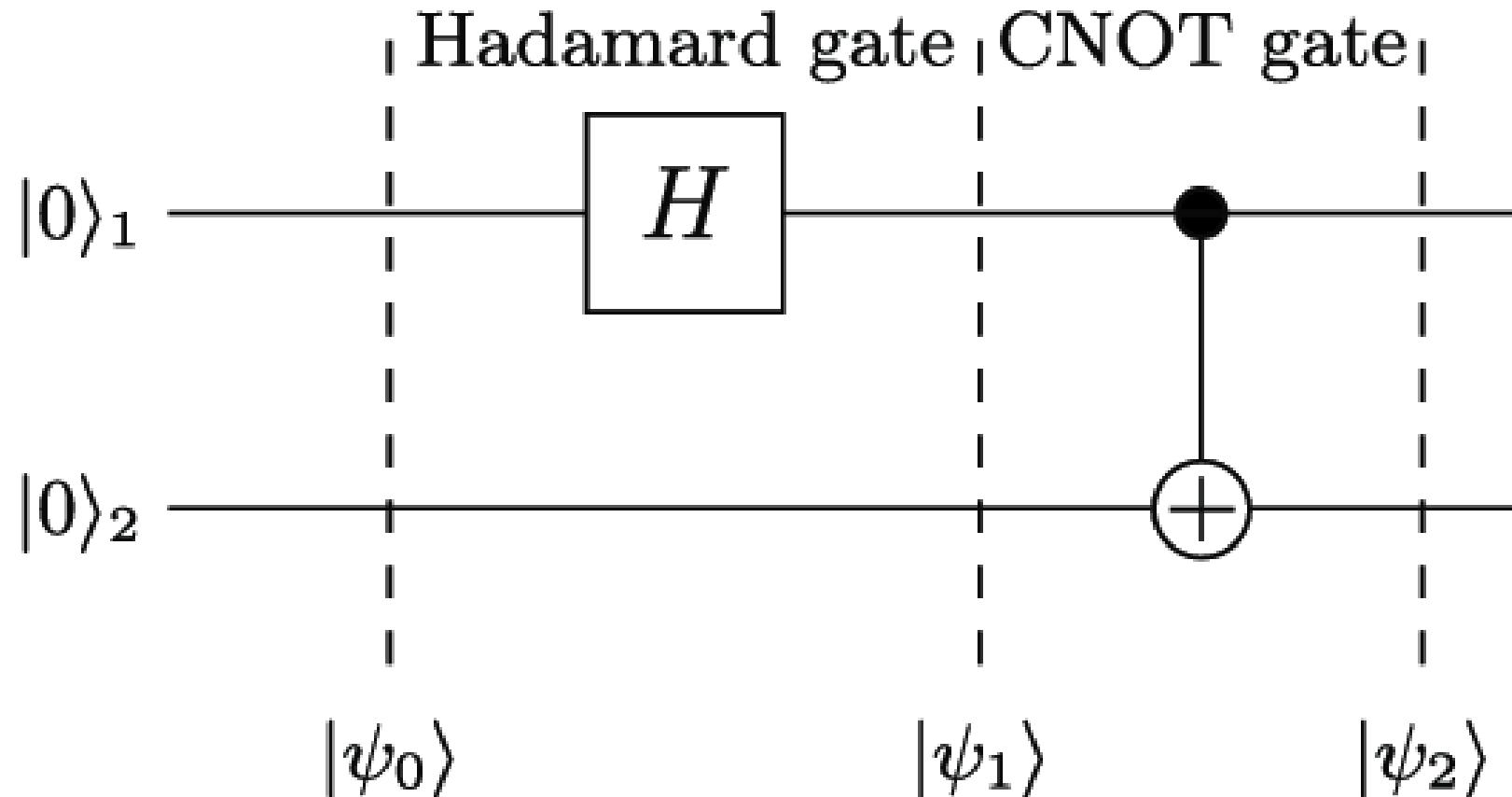


Adiabatic evolution

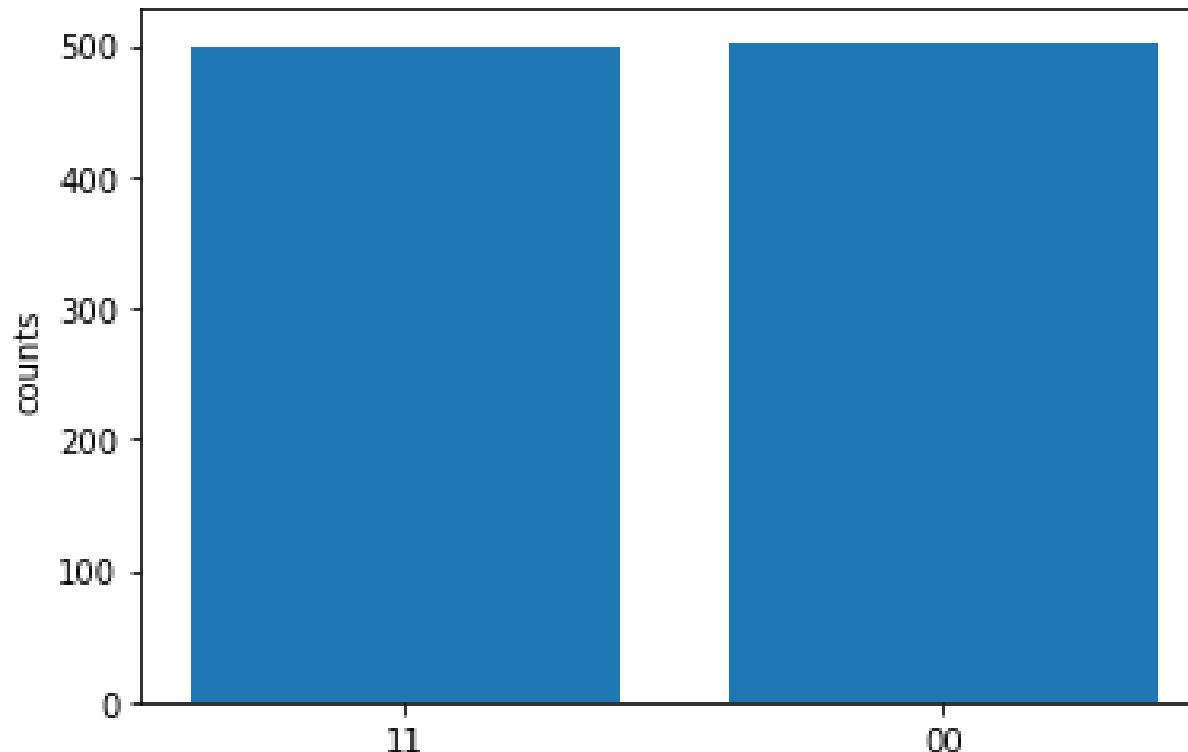
QUANTUM PROGRAMMING

Universal quantum gate model is based on creating quantum structures using stable qubits and solving today's problems with **quantum circuits**



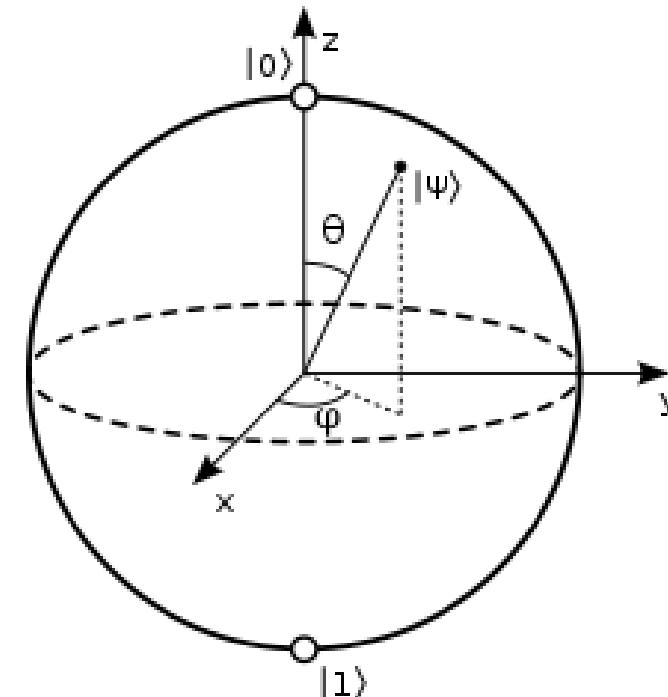
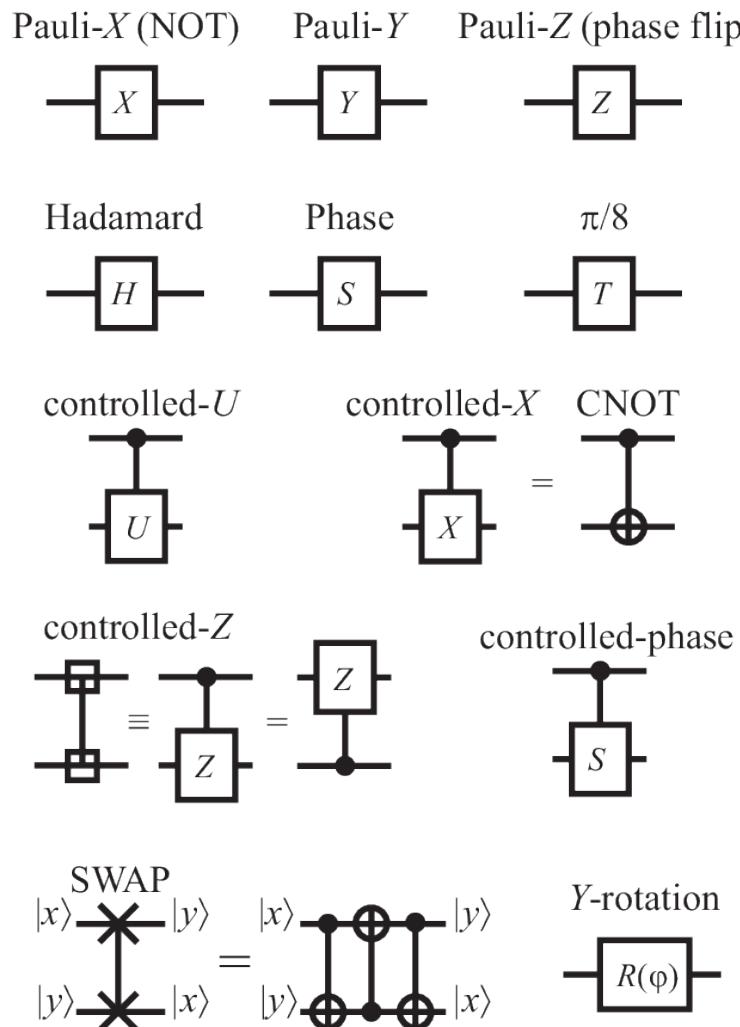


```
bell = Circuit().h(0).cnot(0, 1)
```



Quantum gates

QUANTUM PROGRAMMING



QUANTUM PROGRAMMING

- IBM Quantum Composer: <https://quantum-computing.ibm.com/composer>

The screenshot shows the IBM Quantum Composer interface. At the top, there's a navigation bar with 'File', 'Edit', 'Inspect', 'View', 'Share', 'Sign in' (disabled), and a search bar. Below the bar, it says 'Untitled circuit'. A toolbar below the circuit contains various quantum gate icons: H, \oplus , \oplus_1 , \oplus_2 , I, T, S, Z, T^\dagger , S^\dagger , P, RZ, $|0\rangle$, \otimes , if, $|\cdot\rangle$, \sqrt{X} , \sqrt{X}^\dagger , Y, RX, RY, U, RX, RZZ, and '+ Add'. The circuit itself has two qubits: q₀ and q₁. q₀ starts with an H gate. q₁ has a '+' symbol above it. There are two classical bits: c₀ and c₁. On the right side, there's a 'Visualizations' section with a 'Qiskit' dropdown set to 'Read only'. Below it is a code editor window titled 'Open in Quantum Lab' containing Python code for creating a quantum circuit:

```
1 from qiskit import
2 QuantumRegister,
3 ClassicalRegister,
4 QuantumCircuit
5 from numpy import pi
6
7 qreg_q = QuantumRegister(2,
8 'q')
9 creg_c = ClassicalRegister(2,
10 'c')
11 circuit = QuantumCircuit
12 (qreg_q, creg_c)
13
14 circuit.h(qreg_q[0])
```

At the bottom, there are two visualizations: a 'Probabilities' plot showing a 50% chance for both computational basis states (00 and 01) and a 'Q-sphere' plot showing the state vector on aBloch sphere.

QUANTUM PROGRAMMING

Quantum Languages

Imperative languages

- OpenQASM by IBM for Qiskit and the IBM Q Exp
- Q# A language developed by Microsoft
- QCL One of the first implemented quantum programming languages
- Quantum pseudocode The first formalized language for description of quantum algorithms
- QISI> Is a platform embedded in .Net language supporting quantum programming in a quantum extension of while-language
- Q language Is the second implemented imperative quantum programming language
- qGCL Was defined by P. Zuliani in his PhD thesis. It's based on Guarded Command Language created by Edsger Dijkstra
- QMASM Quantum Macro Assembler a low-level language specific to quantum annealers such as the D-Wave
- Python (late 80s), although not an originally quantum programming language, is the most widely used, including by D-Wave, Google, Rigetti, IBM, Microsoft

Functional languages

- QFC and QPL QFC uses a flow chart syntax, whereas QPL uses a textual syntax.
- QML Is a Haskell-like quantum programming language by Altenkirch and Grattage.
- LIQUiJ> Is a quantum simulation extension on the F# programming language.
- Quantum lambda calculi Are extensions of the classical lambda calculus introduced by Alonzo Church and Stephen Cole Kleene in the 1930s
- Quipper It is implemented as an embedded language, using Haskell as the host language

Quantum Service Providers

Cloud / QCaaS providers



Underlying technology providers



QCs
Optimisation

Honeywell
Trapped Ion
IQBit
Software

IONQ
Trapped Ion
TOSHIBA
Simulator (GPU)

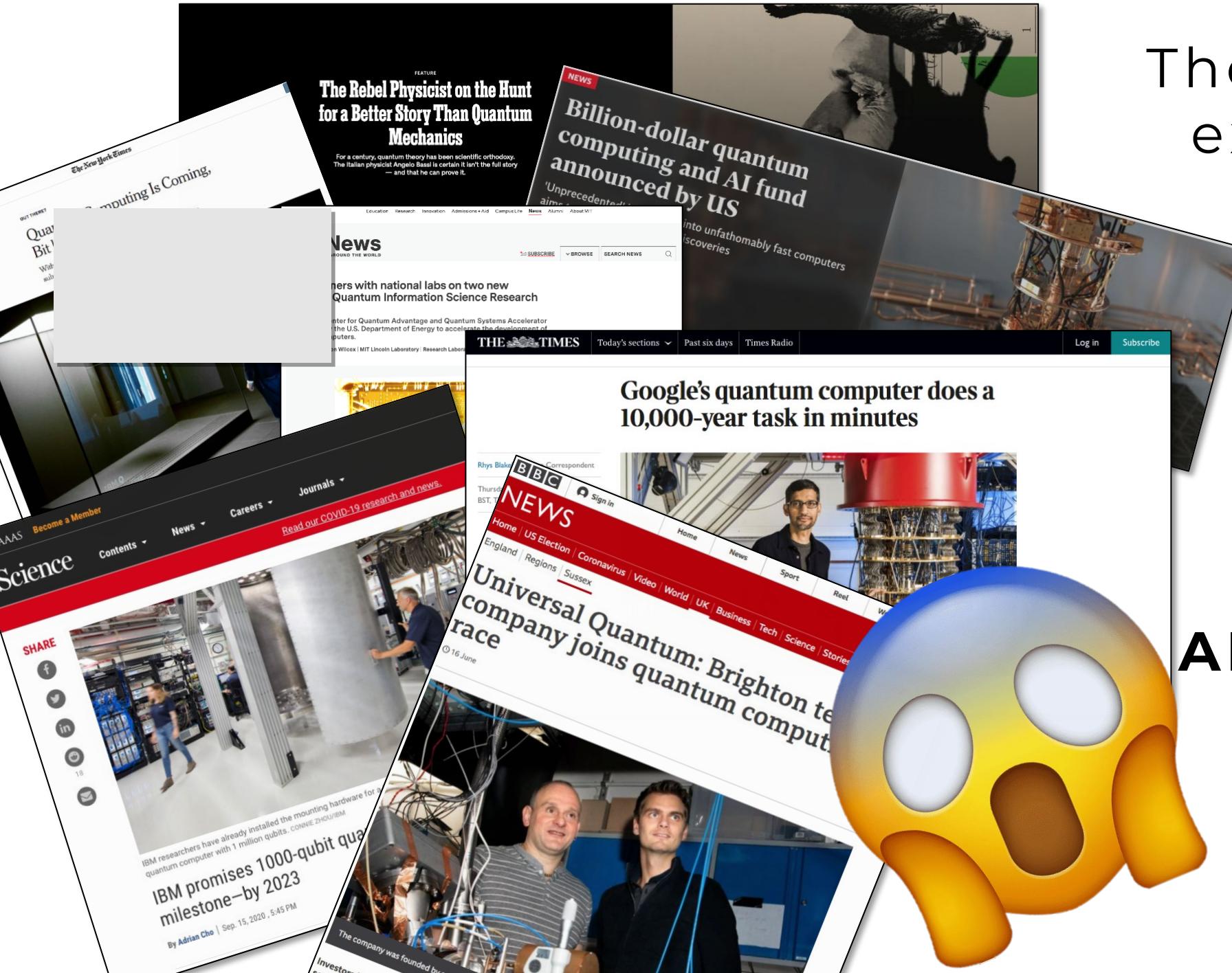


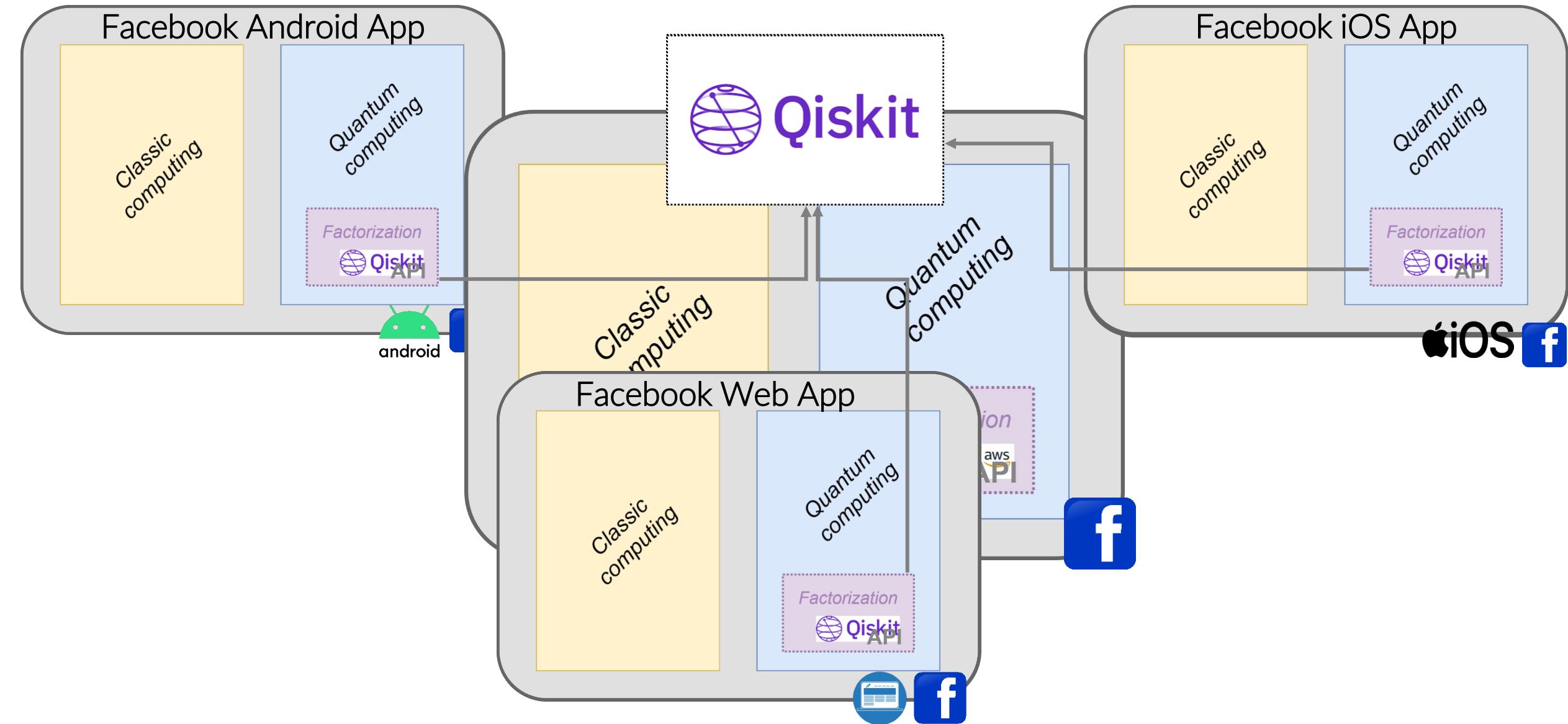
Let's go to the practical part!!



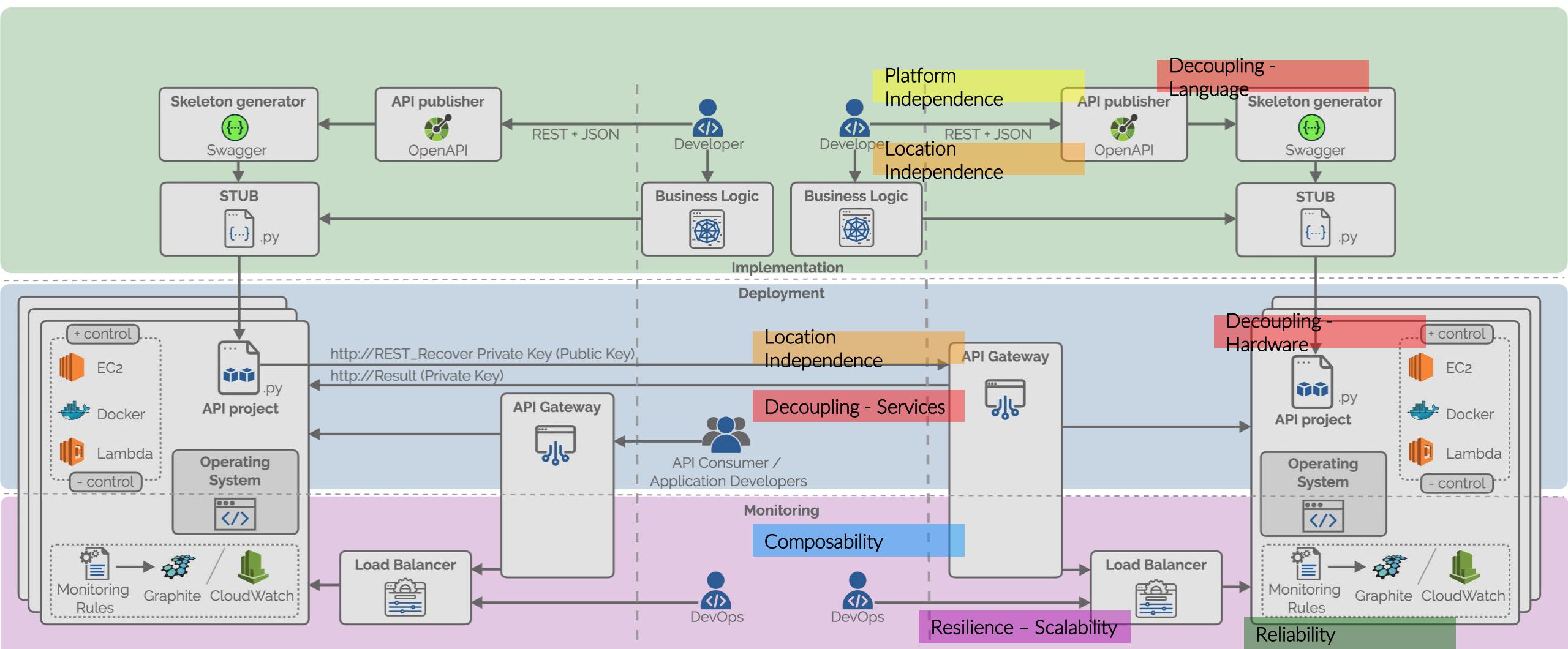
Material of the examples
and preparation of the
environment

There are a lot of expectatives on Quantum Computing ... BUT WE DON'T HAVE TOOLS TO BUILD REAL QUANTUM APPLICATIONS YET

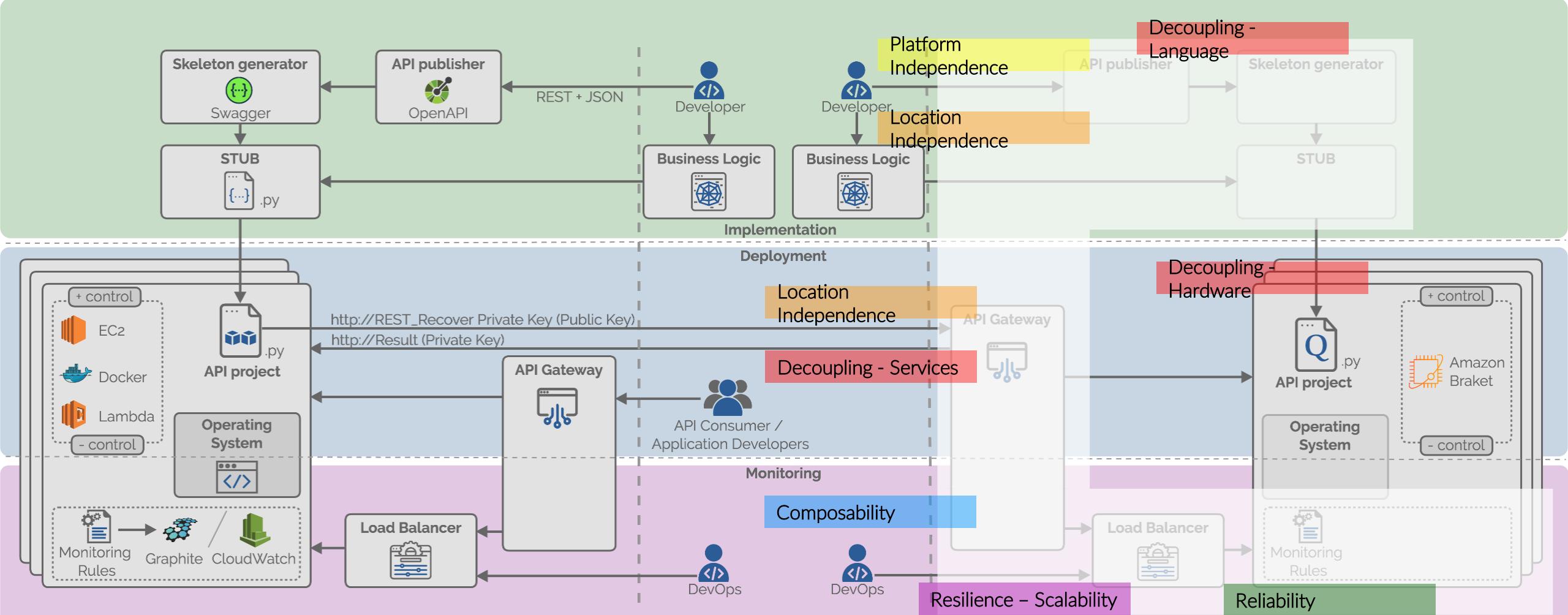




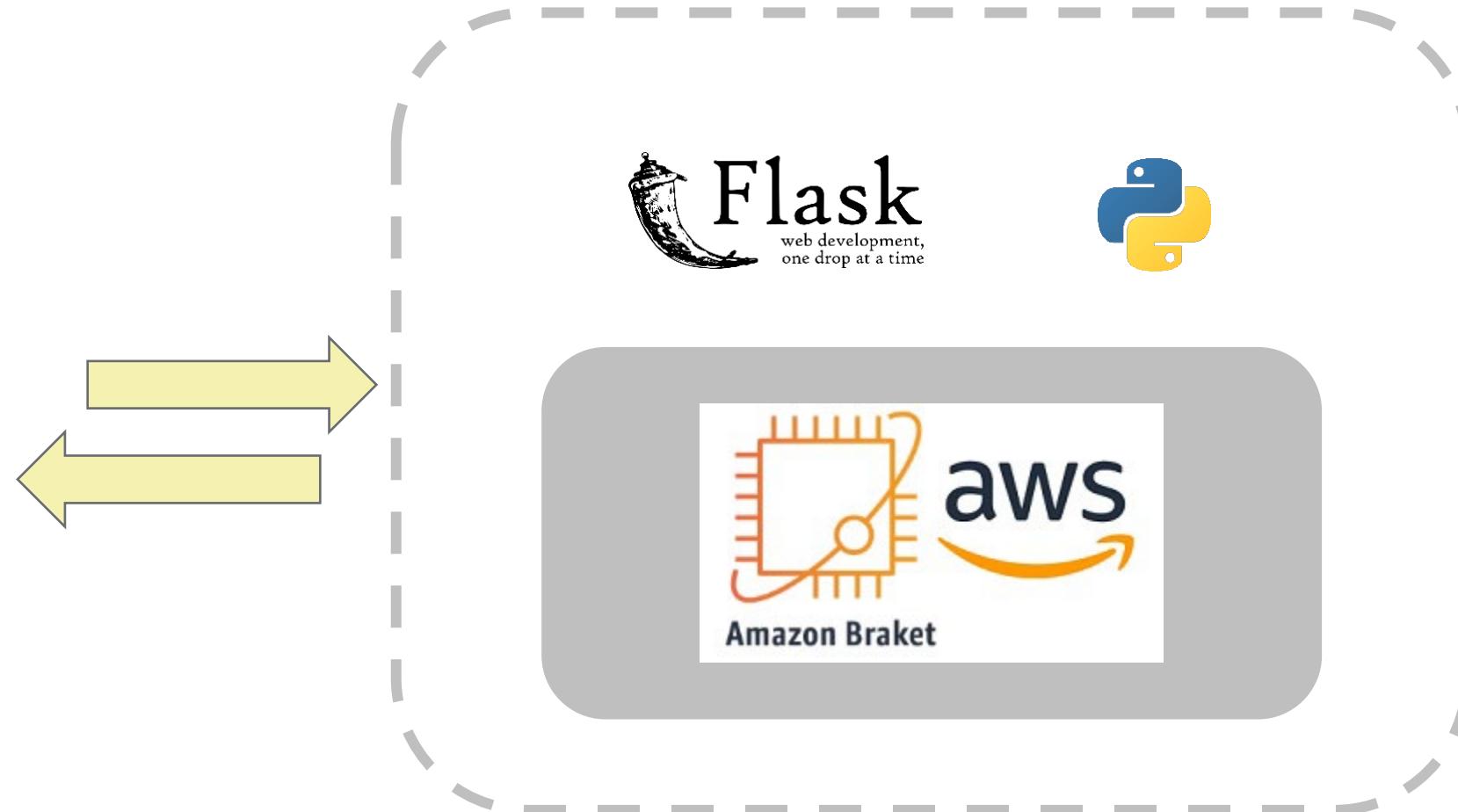
A good classical service implementation



Quantum service implementation



QUANTUM WEB SERVICES WITH AWS



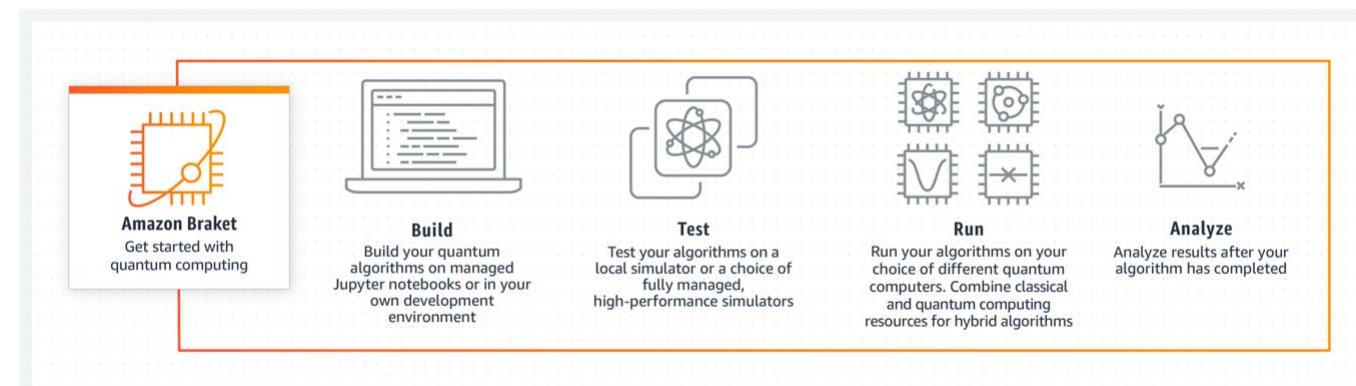
QUANTUM WEB SERVICES WITH AWS

Amazon Braket

- Amazon AWS Quantum Services Provider
- Accelerating quantum computing research
- Access to various quantum computers and simulators
- Cloud programming using Python
- Connection to local services through boto3



Amazon Braket



Flask

- Framework available in Python
 - Creation of web applications
 - Minimalist and "lines-of-code-saving" philosophy
 - Based on Werkzeug's WSGI specification
 - Jinja2 template engine
 - BSD license
-
- **Most famous alternative to Django** due to its reduced learning curve
 - Widely used for the **definition of web services** based on REST APIs



Other tools

Dwave Ocean

- Plugin for programming Dwave annealing machines

Pandas

- Library for data handling and data processing with Python

Matplotlib

- Library for the creation of graphs and visual representations of data with Python.



ADVANCED STEPS: Setting up your environment with Amazon Braket

1. AWS account creation

1. Open AWS Web
2. Choose Create an AWS account
3. Fill in the information
4. Choose personal account
5. Fill in personal information



Sign up for AWS

Root user email address

Used for account recovery and some administrative functions

AWS account name

Choose a name for your account. You can change this name in your account settings after you sign up.

Verify email address

OR

Sign in to an existing AWS account

ADVANCED STEPS: Setting up your environment with Amazon Braket

1. AWS account creation

6. Enter payment method details
7. Verify telephone number

Secure verification

 We will not charge for usage below AWS Free Tier limits. We temporarily hold \$1 USD/EUR as a pending transaction for 3-5 days to verify your identity.



Sign up for AWS

Billing Information

Credit or Debit card number



AWS accepts all major credit and debit cards. To learn more about payment options, review our [FAQ](#)

Expiration date

 Month Year

Cardholder's name

Billing address

Use my contact address

Avenida de la Universidad s/n
Cáceres Extremadura 10003
ES

Use a new address

Verify and Continue (step 3 of 5)

You might be redirected to your bank's website to authorize the verification charge.



ADVANCED STEPS: Setting up your environment with Amazon Braket

1. AWS account creation

8. Choose Basic plan

→ Login with "Root User"

Select a Support Plan

AWS offers a selection of support plans to meet your needs. Choose the support plan that best aligns with your AWS usage. [Learn more](#)

Support Plan	Cost	Features
Basic Plan	Free	<ul style="list-style-type: none">Included with all accounts24/7 self-service access to forums and resourcesBest practice checks to help improve security and performanceAccess to health status and notifications
Developer Plan	From \$29/month	<ul style="list-style-type: none">For early adoption, testing and developmentEmail access to AWS Support during business hours1 primary contact can open an unlimited number of support cases12-hour response time for nonproduction systems
Business Plan	From \$100/month	<ul style="list-style-type: none">For production workloads & business-critical dependencies24/7 chat, phone, and email access to AWS SupportUnlimited contacts can open an unlimited number of support cases1-hour response time for production systems

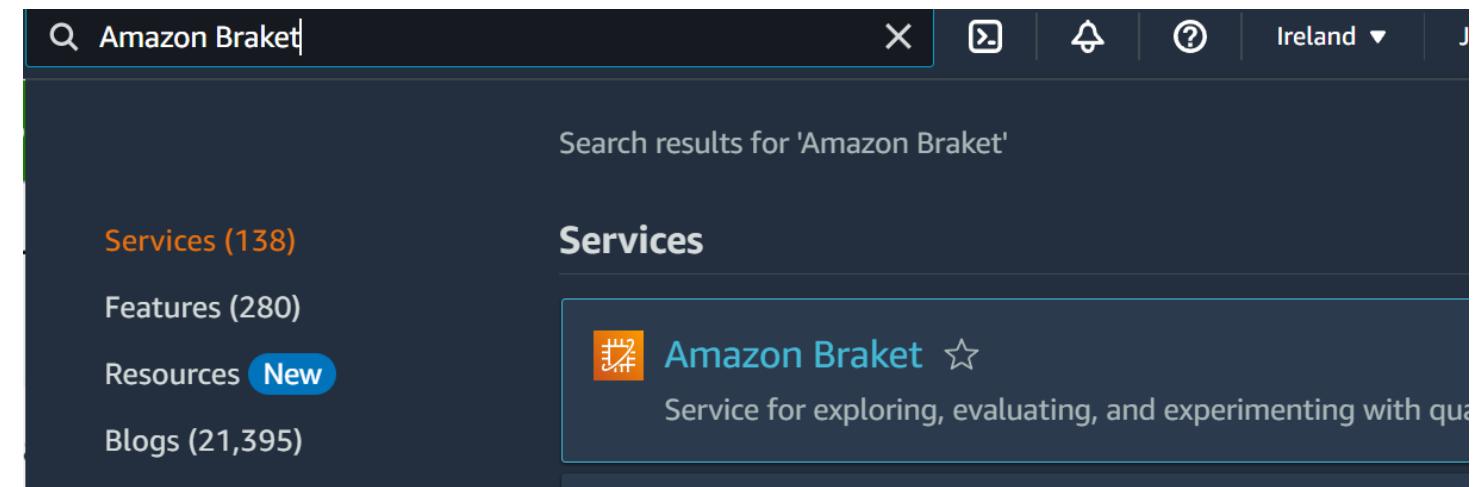
Need Enterprise level support?

Contact your account manager for additional information on running business and mission critical-workloads on AWS (starting at \$15,000/month). [Learn more](#)

ADVANCED STEPS: Setting up your environment with Amazon Braket

2. Using the AWS Braket service

1. Search Amazon Braket in the Amazon Console



ADVANCED STEPS: Setting up your environment with Amazon Braket

2. Using the AWS Braket service

2. Select a valid region

Region Unsupported

Amazon Braket is not available in Europe (Ireland). Please select another region.

Supported Regions
Europe (London)
US East (N. Virginia)
US West (N. California)
US West (Oregon)

ADVANCED STEPS: Setting up your environment with Amazon Braket

2. Using the AWS Braket service

3. Create notebook instance

The screenshot shows the Amazon Braket web interface. On the left, there is a sidebar with the following navigation options: Devices, **Notebooks** (which is currently selected and highlighted in orange), Tasks, and Announcements. The main content area is titled "Amazon Braket > Notebooks". At the top right of this area, there are three buttons: a white button with a black "C" icon, a white "Actions" button with a downward arrow, and an orange "Create notebook instance" button. Below these buttons is a search bar with the placeholder text "Search notebooks". To the right of the search bar are navigation icons: a left arrow, a page number "1", a right arrow, and a gear icon. The main table below has columns for Name, Instance, Creation time, Status, and URL. A message at the bottom of the table says "No Notebooks" and provides instructions: "Use Jupyter Notebooks to create quantum programs in an interactive coding environment." Below this message is a "Create notebook" button.

ADVANCED STEPS: Setting up your environment with Amazon Braket

2. Using the AWS Braket service

4. Select an instance of type *ml.t3.medium* and leave the rest of the options as default

The screenshot shows two stacked configuration panels for creating a new notebook instance.

Notebook instance settings:

- Notebook instance name:** amazon-braket-example
- Notebook instance type:** ml.t3.medium
- Additional settings:** A collapsed section.

Permissions and encryption:

- IAM role:** Create a new role
- Root access — optional:**
 - Enable - Give users root access to the notebook
 - Disable - Don't give users root access to the notebook
- Encryption key — optional:** No custom encryption key

A callout box highlights the IAM role creation step, stating: "Passing an IAM role gives Amazon SageMaker permission to perform actions in other AWS services on your behalf. Creating a role here will grant permissions described by the [AmazonBraketFullAccess](#) IAM policy to the role you create."

ADVANCED STEPS: Setting up your environment with Amazon Braket

2. Using the AWS Braket service

5. Check that the notebook has been created

Notebooks (1)					C	Actions ▾	Create notebook instance
					◀	1	▶
					⟳	⋮	⌚
Name	Instance	Creation time	Status	URL			
amazon-braket-example	ml.t3.medium	Nov 03, 2021 13:17 (UTC)	InService	amazon-braket-example-7cbc.notebook.us-east-1.sagemaker.aws			

ADVANCED STEPS: Setting up your environment with Amazon Braket

2. Using the AWS Braket service

6. Execute notebook examples

jupyter

Open JupyterLab | Quit | Logout

Files Running Clusters Conda

Select items to perform actions on them.

Upload New

	Name	Last Modified	File size
<input type="checkbox"/>	Braket algorithms / textbook	hace unos segundos	
<input type="checkbox"/>	Bells_Inequality.ipynb	hace 6 días	7.06 kB
<input type="checkbox"/>	Bernstein_Vazirani_Algorithm.ipynb	hace 6 días	48.7 kB
<input type="checkbox"/>	CHSH_Inequality.ipynb	hace 6 días	60.9 kB
<input type="checkbox"/>	Deutsch_Jozsa_Algorithm.ipynb	hace 6 días	25.1 kB
<input type="checkbox"/>	Grovers_Search.ipynb	hace 6 días	26.3 kB
<input type="checkbox"/>	Quantum_Approximate_Optimization_Algorithm.ipynb	hace 6 días	43 kB
<input type="checkbox"/>	Quantum_Circuit_Born_Machine.ipynb	hace 6 días	99.4 kB
<input type="checkbox"/>	Quantum_Fourier_Transform.ipynb	hace 6 días	66.4 kB
<input type="checkbox"/>	Quantum_Phase_Estimation_Algorithm.ipynb	hace 6 días	47.7 kB
<input type="checkbox"/>	Quantum_Walk.ipynb	hace 6 días	122 kB
<input type="checkbox"/>	Shors_Algorithm.ipynb	hace 6 días	6.11 kB
<input type="checkbox"/>	Simons_Algorithm.ipynb	hace 6 días	6.6 kB
<input type="checkbox"/>	Template.ipynb	hace 6 días	3.24 kB
<input type="checkbox"/>	notebook_plotting.py	hace 6 días	1.77 kB
<input type="checkbox"/>	README.md	hace 6 días	296 B
<input type="checkbox"/>	requirements.txt	hace 6 días	23 B

jupyter Shors_Algorithm (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Logout Not Trusted conda_braket O

Shor's Algorithm

This example provides an implementation of the Shor's algorithm using the Amazon Braket SDK. Shor's algorithm is used to find prime factors of an integer. On a quantum computer, Shor's algorithm runs in polynomial time and is almost exponentially faster than the most efficient known classical factoring algorithm. The efficiency of Shor's algorithm is due to the efficiency of the Quantum Fourier transform, Quantum Phase estimation and modular exponentiation by repeated squarings. In this notebook, you implement the Shor's algorithm in code using the Amazon Braket SDK and run a simple example of factoring 15.

References

[1] Wikipedia: https://en.wikipedia.org/wiki/Shor%27s_algorithm
[2] Nielsen, Michael A., Chuang, Isaac L. (2010). Quantum Computation and Quantum Information (2nd ed.). Cambridge: Cambridge University Press.

```
In [1]: from braket.devices import LocalSimulator
from braket.aws import AwsDevice
from braket.experimental.algorithms.shors.shors import (
    shors_algorithm,
    run_shors_algorithm,
    get_factors_from_results
)
```

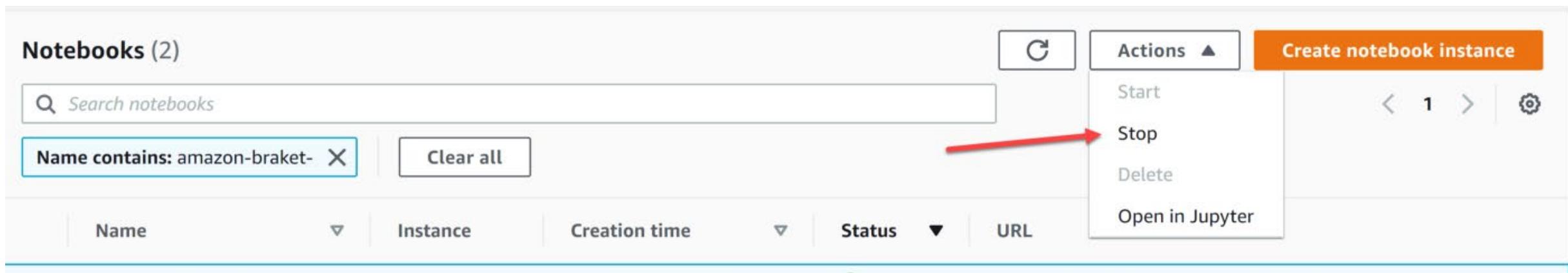
Prepare inputs for Shor's Algorithm

```
In [2]: N = 15 # Integer to factor (currently 15, 21, 35 work)
a = 7 # Any integer that satisfies 1 < a < N and gcd(a, N) = 1.
```

ADVANCED STEPS: Setting up your environment with Amazon Braket

2. Using the AWS Braket service

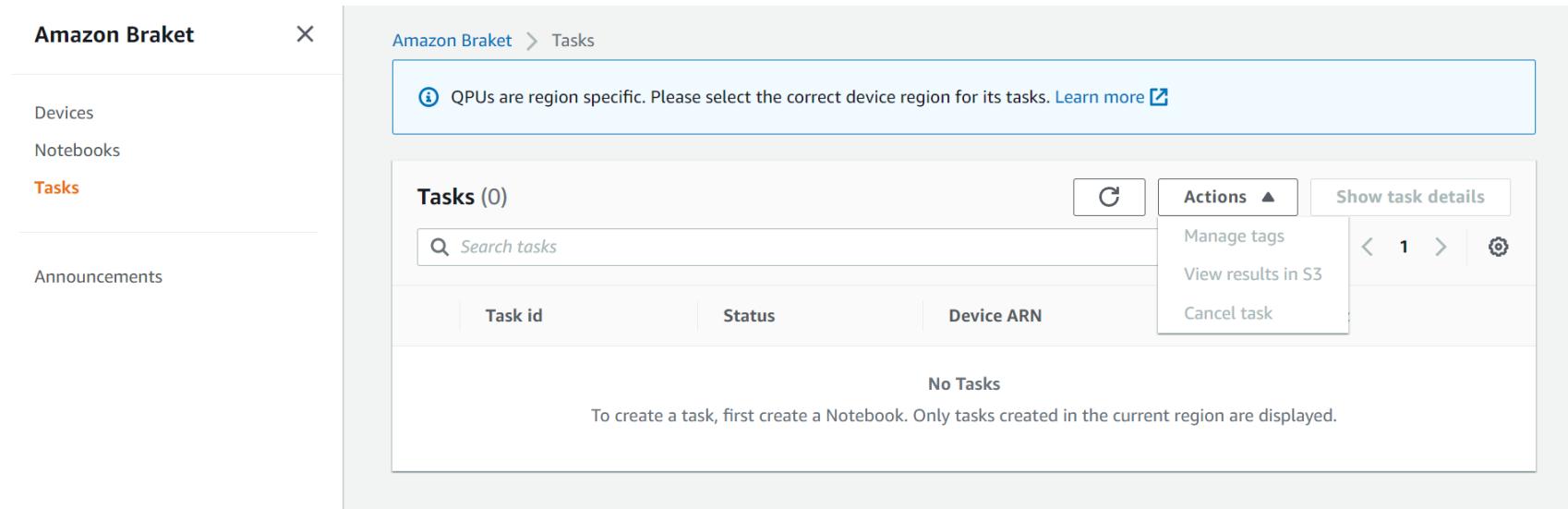
7. Stop notebook instance



ADVANCED STEPS: Setting up your environment with Amazon Braket

2. Using the AWS Braket service

8. Stop pending tasks



The screenshot shows the Amazon Braket interface with the 'Tasks' tab selected. A message at the top states: "QPUs are region specific. Please select the correct device region for its tasks. [Learn more](#)". Below this, a table titled "Tasks (0)" is displayed with columns for "Task id", "Status", and "Device ARN". The table is empty, showing "No Tasks". To the right of the table is an "Actions" dropdown menu with options: "Manage tags", "View results in S3", and "Cancel task".

Local installation

1. Python 3 intallation

Windows

1. Go to the Windows Store
2. Search for Python 3.9
3. Click on get
4. Open a CMD
5. “python --version” o “python3 --version”

Ubuntu

1. sudo apt-get update
2. sudo apt-get install python3.9
3. “python --version” o “python3 --version”

MacOS

1. /bin/bash -c "\$(curl -fsSL <https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh>)"
- 2.export PATH="/usr/local/opt/python/libexec/bin:\$PATH"
- 3.brew install Python
4. “python --version” o “python3 --version”

Local installation

2. Installing the Amazon Braket SDK, Flask and other tools

```
cd amazon-braket-algorithm-library
```

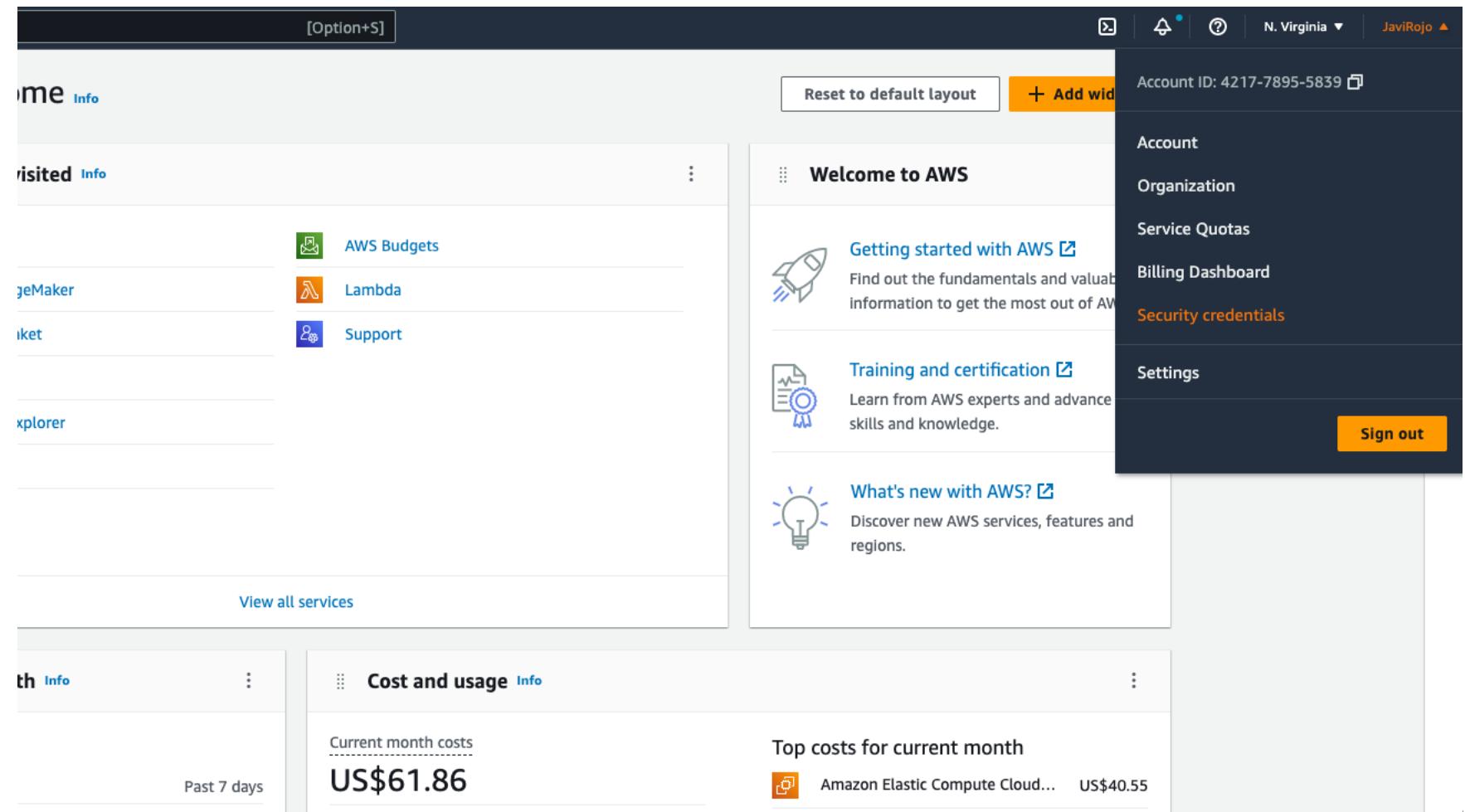
```
pip install .
```



Local installation

3. Connecting our quantum services to the AWS account

1. Get aws access key



Local installation

3. Connecting our quantum services to the AWS account

1. Get aws access key

The screenshot shows two related AWS IAM pages:

Multifactor authentication (MFA) (0)

Use MFA to increase the security of your AWS environment. Signing in with MFA requires an authentication code from an MFA device. Each user can have a maximum of 8 MFA devices assigned. [Learn more](#)

Access keys (0)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

MFA Devices Table Headers:

Device type	Identifier	Created on
-------------	------------	------------

No MFA devices. Assign an MFA device to improve the security of your AWS environment

Assign MFA device

Access Keys Table Headers:

Access key ID	Created on	Access key last used	Region last used	Service last used	Status
---------------	------------	----------------------	------------------	-------------------	--------

No access keys

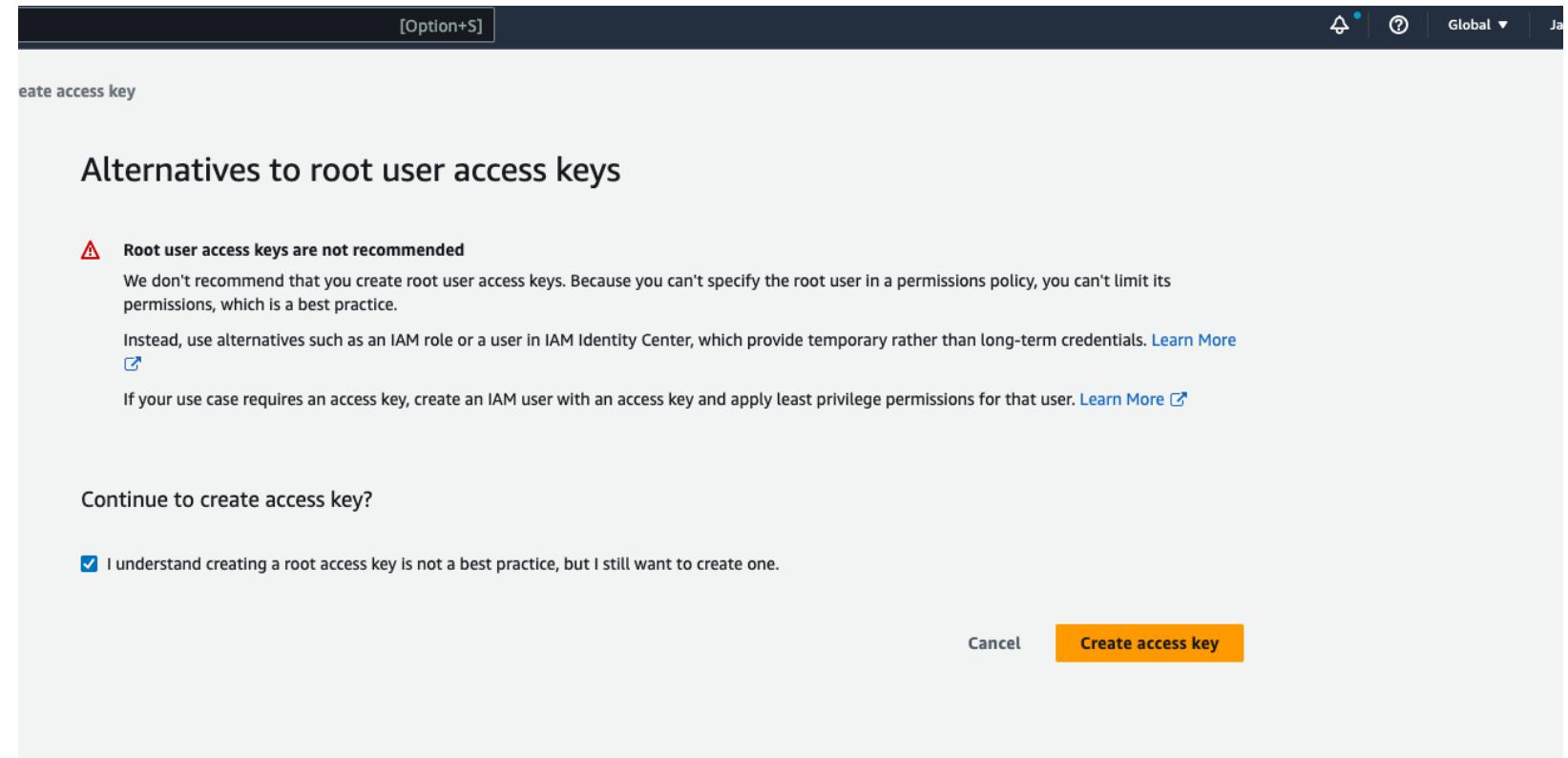
As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials. [Learn more](#)

Create access key

Local installation

3. Connecting our quantum services to the AWS account

1. Get aws access key



Local installation

3. Connecting our quantum services to the AWS account

1. Get aws access key

The screenshot shows the AWS IAM Access Keys page. At the top, there is a green banner with the text: "Your access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time." Below the banner, there is a link to "Create access key". The main section is titled "Retrieve access key" and contains a table with two columns: "Access key" and "Secret access key". The "Access key" column contains the value "AKIAWEM74DY7WMFUG3N5" and the "Secret access key" column contains a redacted value followed by a "Show" link. A large red oval highlights this row. Below the table, there is a section titled "Access key best practices" with the following bullet points:

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

At the bottom of the page, there is a link to "Best practices for managing AWS access keys". At the very bottom, there are two buttons: "Download .csv file" and "Done".

Local installation

3. Connecting our quantum services to the AWS account

2. Entering passwords in our computer

2.1. Create folder ".aws" in root of our user

2.2. Create "credentials" and "config" files without extension in .aws folder

Local installation

3. Connecting our quantum services to the AWS account

File “credentials” (REPLACE access key id & secret access key)

```
[default]
```

```
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
```

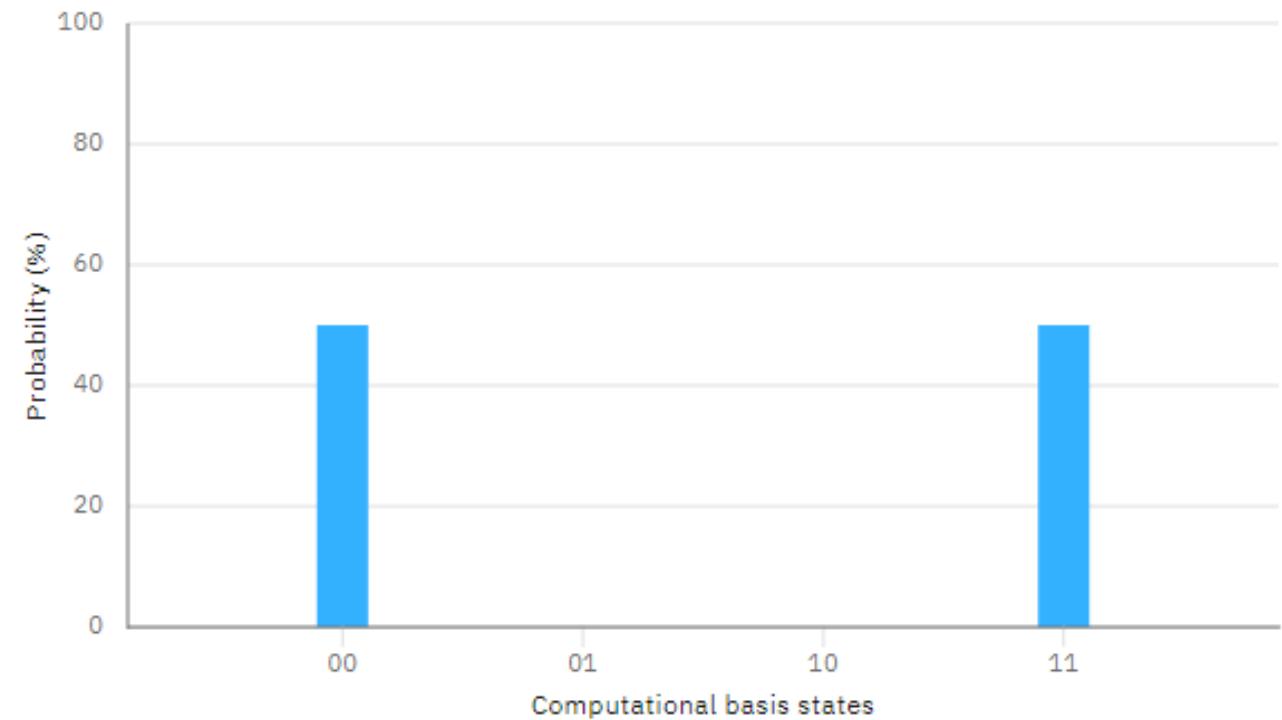
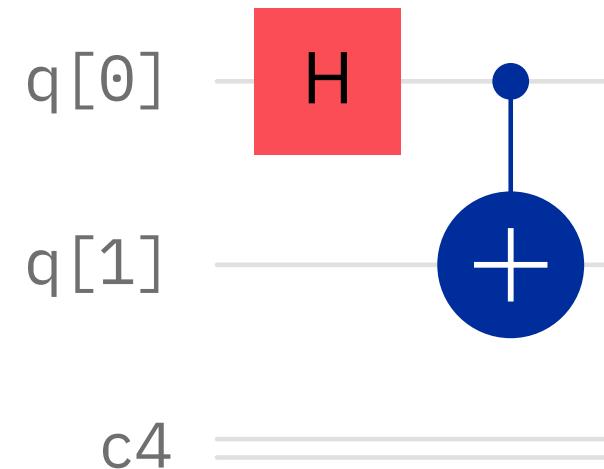
```
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

Local installation

3. Conectar nuestros servicios cuánticos con la cuenta de AWS

File “config”

```
[default]  
region=us-west-2  
output=json
```



```
@app.route('/execute', methods=["get"])
def execute_quantum_task():

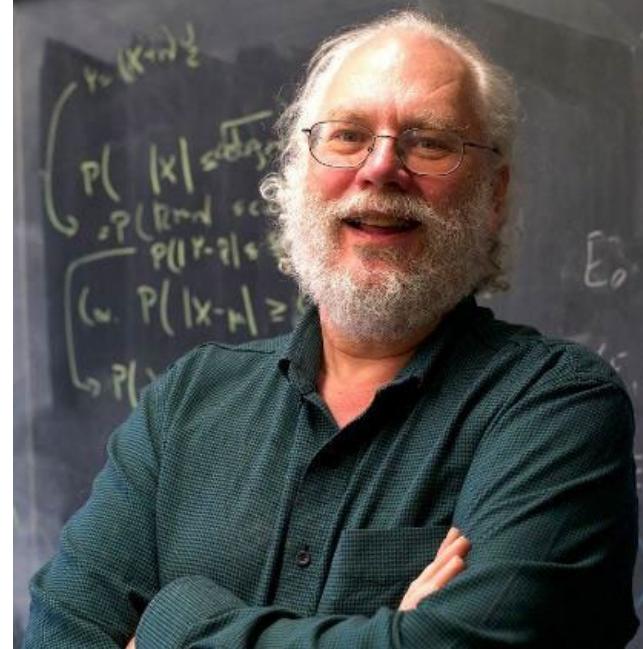
    # build a Bell state with two qubits.
    # Here 'cnot(control=0, target=1)' can be simplified as 'cnot(0,1)'
    bell = Circuit().h(0).cnot(control=0, target=1)

    # set up device
    device = LocalSimulator()

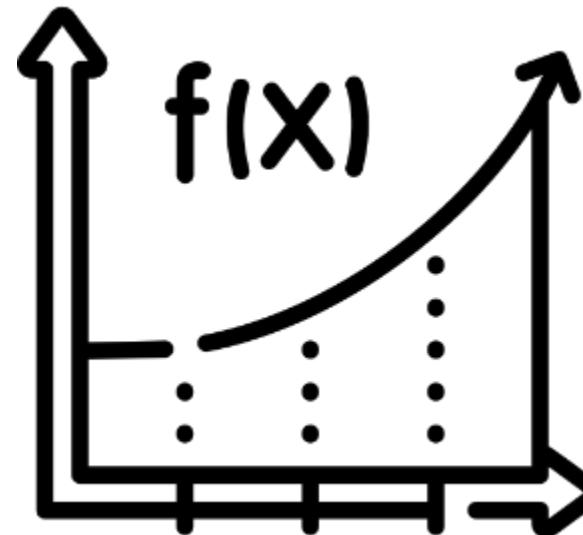
    # run circuit
    result = device.run(bell, shots=100).result()
    # get measurement shots
    counts = result.measurement_counts

    # print counts
    print(counts)
    # plot using Counter
    plt.clf()
    plt.bar(counts.keys(), counts.values())
    plt.xlabel('bitstrings')
    plt.ylabel('counts')
    plt.savefig("result.png")
    return send_file("result.png", mimetype='image/png')
```

- Shor's algorithm is a quantum computer algorithm for finding the prime factors of an integer
- It was developed in **1994** by the American mathematician **Peter Shor**



- Quantum algorithm for factoring a number N in $O((\log N)^3)$ time and $O(\log N)$ space



- Is a method that allows finding prime factors of a number in an efficient way

- It can be carried out **classically** or using **quantum circuits**.
- This algorithm demonstrates that the integer factorization problem can be efficiently solved on a quantum computer.

- This algorithm is interesting because many public key cryptographies, such as **RSA**, are based on the product of two prime numbers.



- This kind of cryptography could become deprecated if the algorithm could be efficiently implemented on a quantum computer.

- Shor's algorithm was practically demonstrated in 2001 by IBM, which decomposed 15 into its factors 3 and 5, using a quantum computer with 7 qubits.

REUTERS SCIENCE 12.19.2001 02:15 PM

Big Blue Takes Quantum Step

IBM researchers make another advance in quantum computing, demonstrating "Shor's Algorithm," which can break large encryption codes.

NEW YORK -- IBM researchers said on Wednesday they have demonstrated a calculation that could be used to break complicated codes, marking a small step in the advance of quantum computing, a technology based on quantum mechanics.

IBM scientists will publish details in the scientific journal *Nature* on Thursday of the demonstration of "Shor's Algorithm," a method of factoring numbers that was developed in 1994 by AT&T scientist Peter Shor.

It was that algorithm, and the promise it holds for its ability to break large encryption codes, that spurred interest in quantum computing in the 1990s.

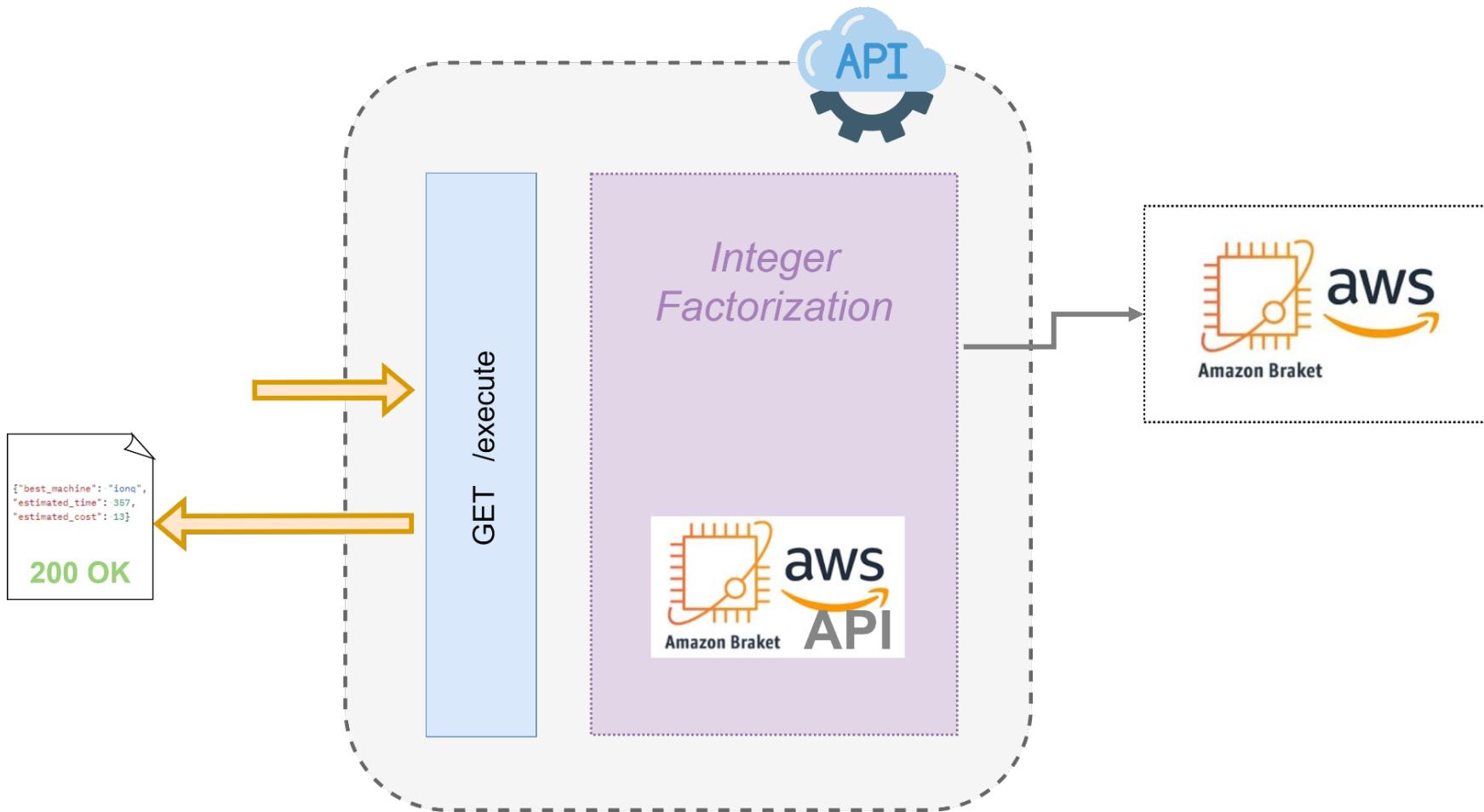
Quantum computing is one of several paths that researchers are taking as they strive to make smaller and smaller microchips. Under Moore's Law, which was set

WATCH



What the What Is Quantum Computing? We've Got You Covered

SHOR SERVICE



Valencia, D., García-Alonso, J., Rojo, J., Moguel, E., Berrocal, J., & Murillo, J. M. "Hybrid Clasical-Quantum Software Services Systems: Exploration of the Rough Edges." QUATIC'21. 2021.

SHOR SERVICE

Two different implementations for each simulator:

```
def shor(N, a):
# N = Integer to factor (currently 15, 21, 35 work)
# a = Any integer that satisfies 1 < a < N and gcd(a, N) = 1.

    shors_circuit = shors_algorithm(N, a)

    local_simulator = LocalSimulator()

    output = run_shors_algorithm(shors_circuit, local_simulator)

    guessed_factors = get_factors_from_results(output, N, a)

    return (guessed_factors)

@app.route('/shor_simulator', methods=['GET'])
def shor_simulator():
    N = int(request.args.get('N'))
    a = int(request.args.get('a'))
    response = shor(N, a)

    return jsonify(str(response))
```

```
@app.route('/shor_simulator_sv1', methods=['GET'])
def shor_simulator_sv1():
    N = int(request.args.get('N'))
    a = int(request.args.get('a'))
    shors_circuit = shors_algorithm(N, a)
    managed_sim = AwsDevice("arn:aws:braket::::device/quantum-simulator/amazon/sv1")
    output = run_shors_algorithm(shors_circuit, managed_sim)
    guessed_factors = get_factors_from_results(output, N, a)

    return jsonify(str(guessed_factors))
```

API Deployment

1. Run the follow commands in your system terminal:

- cd Shor
- python shor_service.py

or

python3 shor_service.py

```
Last login: Thu Feb 16 10:12:58 on ttys000
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) MacBook-Pro-de-Javier:amazon-braket-algorithm-library javier$ cd Shor
(base) MacBook-Pro-de-Javier:Shor javier$ python shor_service.py
 * Serving Flask app 'shor_service'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
 * Running on http://127.0.0.1:8080
Press CTRL+C to quit
```

API Deployment

2. Download Postman:

<https://www.postman.com/downloads/>



3. Import collection file (Shor service.postman_collection.json) from Shor folder:

A screenshot of the Postman application interface. On the left, there's a sidebar with 'API Network' and 'Explore' tabs, and a main area with 'Import Elements' and a 'New' button. In the center, a modal window titled 'Import Elements' is open. It has tabs for 'Import' (selected), 'File', 'Folder', 'Link', 'Raw text', 'Code repository', 'API Gateway', and 'New'. Below these tabs is a 'Choose Files' button. To the right of the modal, there's a preview area showing 'Format: Postman Collection v2.1'. On the far right, there's another 'Import Elements' window titled 'Import'. It has a search bar, a 'Collections' section with checkboxes for 'Collection Name' and 'Shor service', and an 'Import' button at the bottom.



API Deployment

4. Test the shor service in postman:

- Shor local simulator: http://localhost:8080/shor_simulator?N=15&a=7
- Shor sv1 simulator: http://localhost:8080/shor_simulator_sv1?N=15&a=7

Shor service / Shor sv1 simulator AWS

GET http://localhost:8080/shor_simulator_sv1?N=15&a=7 Send

Params • Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> N	15	Shor service / Shor local simulator		
<input checked="" type="checkbox"/> a	7	GET http://localhost:8080/shor_simulator?N=15&a=7		
Key	Value	Description		

Shor service / Shor local simulator

GET http://localhost:8080/shor_simulator?N=15&a=7 Send

Params • Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> N	15			
<input checked="" type="checkbox"/> a	7			
Key	Value	Description		

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 "{'guessed_factors': 3.5}"
```

Status: 200 OK Time: 439 ms Size: 196 B Save Response

66/34

Quantum Web Services: development and deployment

Quantum OpenAPI

Javier Romero-Álvarez
Jaime Alvarado-Valiente
Enrique Moguel
José García-Alonso



JUNTA DE EXTREMADURA
Consejería de Economía, Ciencia y Agenda Digital

SPILab
by Quercus SEG

Software Engineering Group
QUERCUS
UNIVERSIDAD DE EXTREMADURA

Q
EX
UNIVERSIDAD DE EXTREMADURA

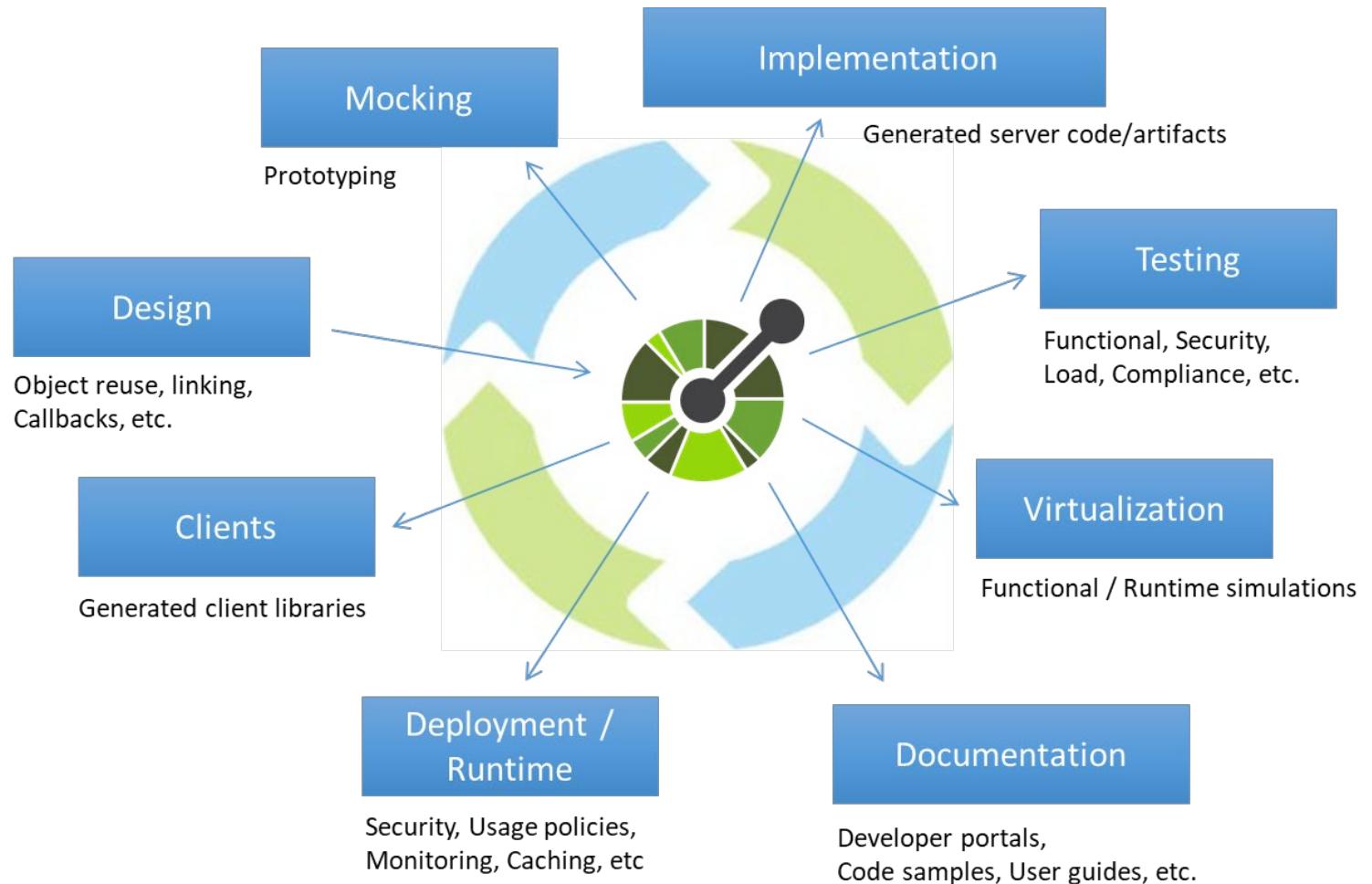


OUTLINE

1. Introduction
 - a. OpenAPI
 - b. GitHub Actions
2. Quantum service generation
3. Quantum service deployment
4. Workflow for the continuous deployment

INTRODUCTION

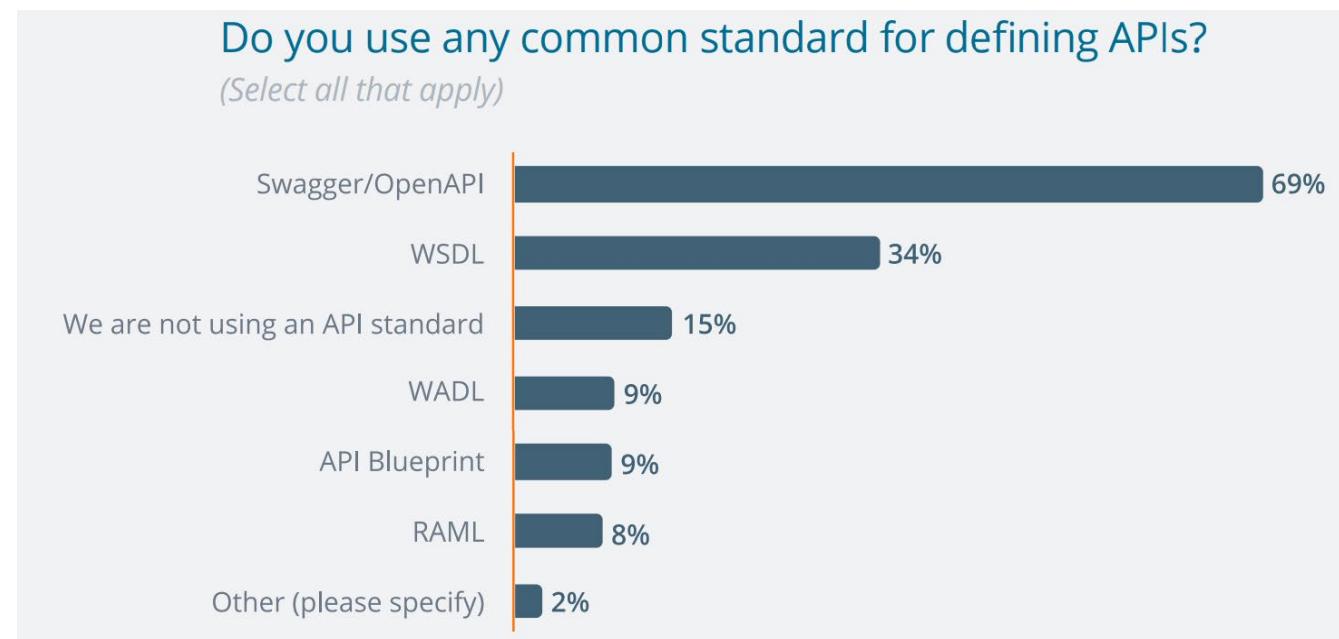
- The OpenAPI Initiative (OAI) was created by a consortium to standardise the way REST APIs are described.
- OpenAPI is a specification for describing, consuming and visualising RESTful web services.



INTRODUCTION

Why OpenAPI?

OpenAPI is used and supported by large companies and, over the years, has established itself as the standard to use for building APIs for apps



<https://smartbear.com/blog/a-look-at-the-key-takeaways-from-state-of-api-2020/>

INTRODUCTION

Open API Tools

➤ Editor

<http://editor.swagger.io>

➤ API explorer

<http://petstore.swagger.io>

➤ Validator

<https://online.swagger.io/validator>

➤ Opensource generator:

- skeletons para backends
- proxies para clientes o front-end

<https://openapi-generator.tech/>



Open API Tools

INTRODUCTION

- Editor: <http://editor.swagger.io>

The screenshot shows the Swagger Editor interface. On the left, the OpenAPI specification is displayed in a code editor:

```
openapi: 3.0.2
info:
  title: Quantum Services
  description: This API enables the generation of quantum services.
  version: '1.0'
  termsOfService: 'https://quantum.spilab.es/terms'
  contact:
    name: Spilab
    url: 'https://spilab.es'
    email: info@spilab.es
  tags:
    - name: quantum
      description: Everything about your Quantum Service
    - name: quantum_code
      description: Everything about your quantum service code and circuits
paths:
  /circuit/randomAWS:
    get:
      tags:
        - quantum_code
        summary: Get the circuit implementation of Grover Algorithm
        description: ''
        operationId: grover_circuitAWS
        parameters:
          - name: machine
            in: query
            description: Name of the machine where to execute
            required: true
            style: form
            explode: false
            deprecated: true
            schema:
              type: string
          - name: shots
            in: query
            description: Number of shots
            required: true
            style: form
            explode: false
            deprecated: true
            schema:
              type: number
        responses:
          '200':
            description: successful operation
          '405':
            description: Invalid execution
          x-quantumCode: 'https://algassert.com/quirk#circuit=[%22cols%22:[[%22H%22,%22H%22,%22H%22,%22H%22,%22H%22,%22Measure%22,%22Measure%22,%22Measure%22,%22Measure%22,%22Measure%22]]]'

  /circuit/randomIBM:
    get:
      tags:
        - quantum_code
        summary: Get the circuit implementation of Grover Algorithm
        description: ''
        operationId: grover_circuitIBM
```

On the right, the generated API documentation is shown:

Quantum Services 1.0 OAS3

This API enables the generation of quantum services.

Terms of service
Spilab - Website
Send email to Spilab
Find out more about Swagger

quantum

Everything about your Quantum Service

quantum_code

Everything about your quantum service code and circuits

GET /circuit/randomAWS Get the circuit implementation of Grover Algorithm

GET /circuit/randomIBM Get the circuit implementation of Grover Algorithm

Schemas

quantum category >

quantum code >

- API explorer: <http://petstore.swagger.io>

The screenshot shows a web browser window displaying the Swagger UI for the Petstore API. The URL in the address bar is <https://petstore.swagger.io/v2/swagger.json>. The page title is "Swagger Petstore 1.0.6". The main content area is titled "pet" and lists various API endpoints:

- POST /pet/{petId}/uploadImage** uploads an image
- POST /pet** Add a new pet to the store
- PUT /pet** Update an existing pet
- GET /pet/findByStatus** Finds Pets by status
- GET /pet/findByTags** Finds Pets by tags
- GET /pet/{petId}** Find pet by ID
- POST /pet/{petId}** Updates a pet in the store with form data
- DELETE /pet/{petId}** Deletes a pet

At the top of the page, there is a "Schemes" dropdown set to "HTTPS" and an "Authorize" button. The "Explore" button is located at the top right of the header bar.

- Validator: <https://online.swagger.io/validator>

The screenshot shows the Swagger UI Validator interface. At the top, it displays the URL <https://validator.swagger.io/validator/openapi.json>. Below the header, there's a section titled "Swagger Validator Badge" with a "2.1.2 OAS3" badge. It says "Parses and validates a Swagger/OpenAPI 2.0 or an OpenAPI 3.x definition". A dropdown menu under "Servers" is set to "/validator/". The main area is titled "Validator" and lists several API endpoints:

- GET /** Validates a Swagger/OpenAPI 2.0 or an OpenAPI 3.x definition returning a valid/invalid badge
- POST /** Validates a Swagger/OpenAPI 2.0 or an OpenAPI 3.x definition returning a valid/invalid badge
- GET /debug** Validates a Swagger/OpenAPI 2.0 or an OpenAPI 3.x definition returning a validation response
- POST /debug** Validates a Swagger/OpenAPI 2.0 or an OpenAPI 3.x definition returning a validation response
- GET /parse** Resolves / Dereferences a Swagger/OpenAPI 2.0 or an OpenAPI 3.x definition returning the resolved file
- POST /parse** Resolves / Dereferences Swagger/OpenAPI 2.0 or an OpenAPI 3.x definition returning the resolved file

Below the Validator section is a "Schemas" section which is currently collapsed.

Open API Tools

INTRODUCTION

- Opensource generator: <https://openapi-generator.tech/>

The screenshot shows the homepage of the OpenAPI Generator website. The top navigation bar includes links for "Getting Started", "Generators", "Roadmap", "FAQ", "Team", "Blog", and "API". Below the header, a large green banner features the text "OpenAPI Generator" and "Generate clients, servers, and documentation from OpenAPI 2.0/3.x documents". It also contains buttons for "Try It Out", "Install", and "Generators", "Customization", "Integrations". A dark sidebar on the left lists "Sponsors" and "Bronze Sponsors". The main content area highlights the tool's ease of use and server integration.

Sponsors

If you find OpenAPI Generator useful, please consider asking your company to [become a sponsor](#).

You can also individually sponsor the project by [becoming a backer](#).

Thank you to our bronze sponsors!

Easy to Use

With 50+ client generators, you can easily generate code to interact with any server which exposes an OpenAPI document.

Servers

Getting started with server development can be tough, especially if you're evaluating technologies. We can reduce the burden when you bring your own OpenAPI document.

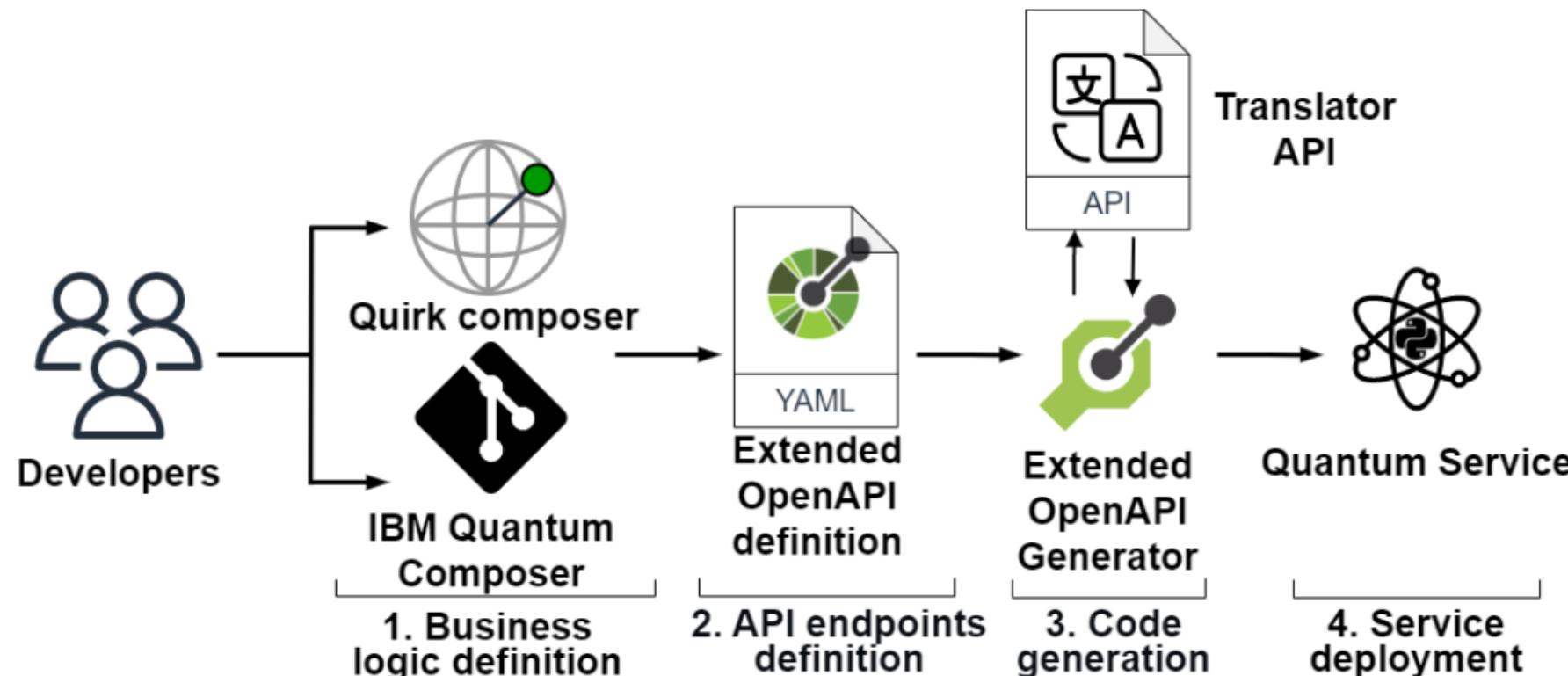
QUANTUM SERVICES GENERATION

To implement a classic service using OpenAPI, a developer needs to combine two main aspects:

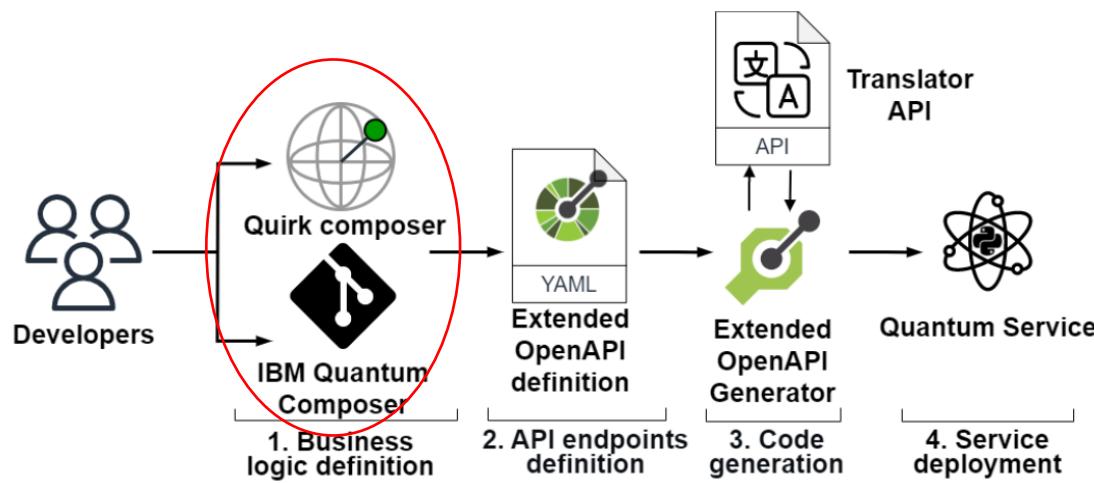
- **The business logic** of the service, which implements the functionality provided by the service.
- **The endpoint** of the service, which determines how an external client can invoke the service.

QUANTUM SERVICES GENERATION

Defining a model for the servitisation of quantum algorithms by extending the OpenAPI specification



QUANTUM SERVICES GENERATION

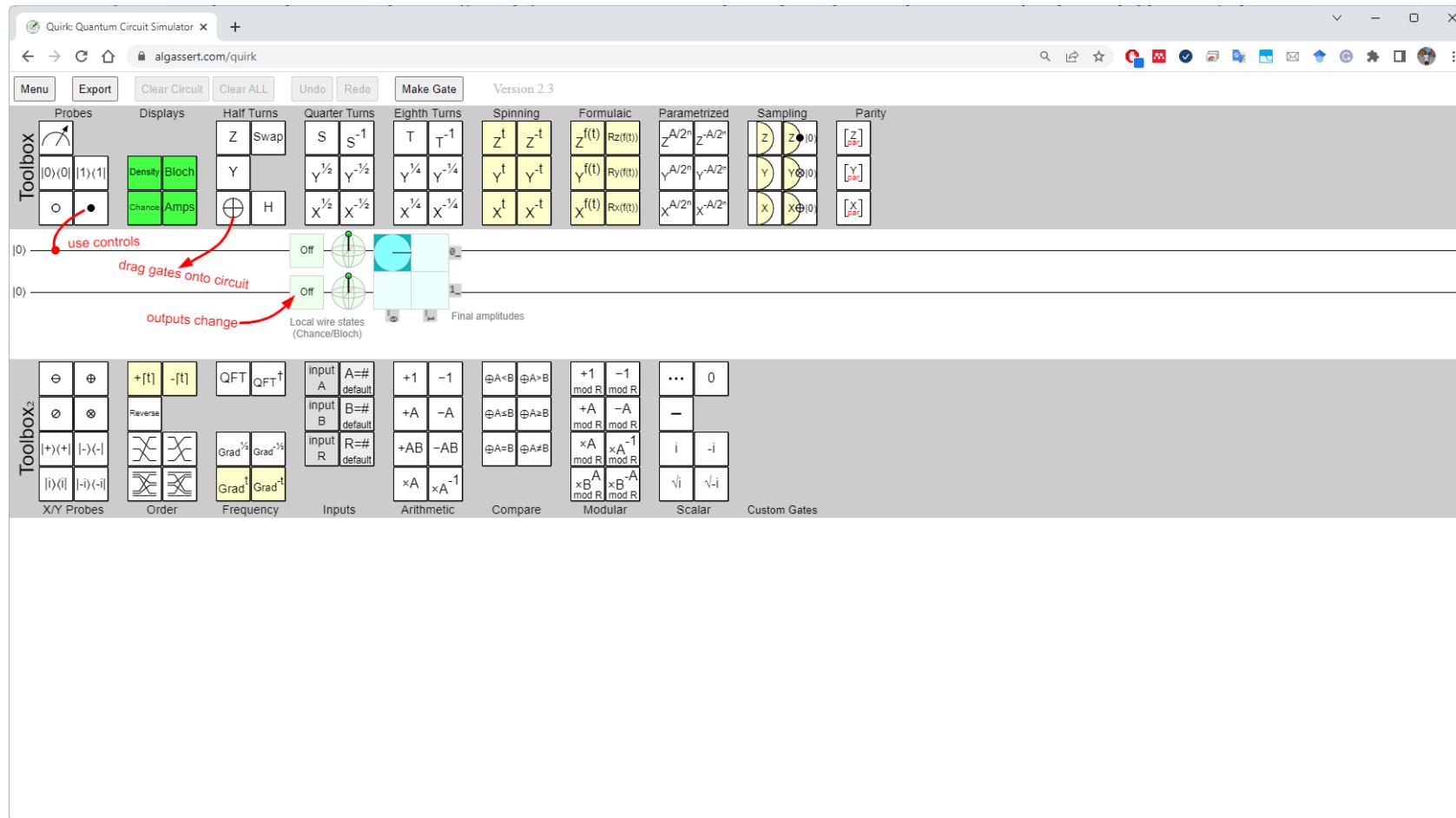


The first step is to define the business logic of the service as a quantum circuit using **Open Quirk**, indicating the Open Quirk link of the created circuit. Or directly indicating the link where is the source code in Qiskit

QUANTUM SERVICES GENERATION

OpenAPI for quantum algorithm servitisation

1. Definition of the circuit corresponding to the Quantum Algorithm with Quirk



QUANTUM SERVICES GENERATION

OpenAPI for quantum algorithm servitisation

1. Definition of the circuit corresponding to the Quantum Algorithm with IBM Quantum Composer

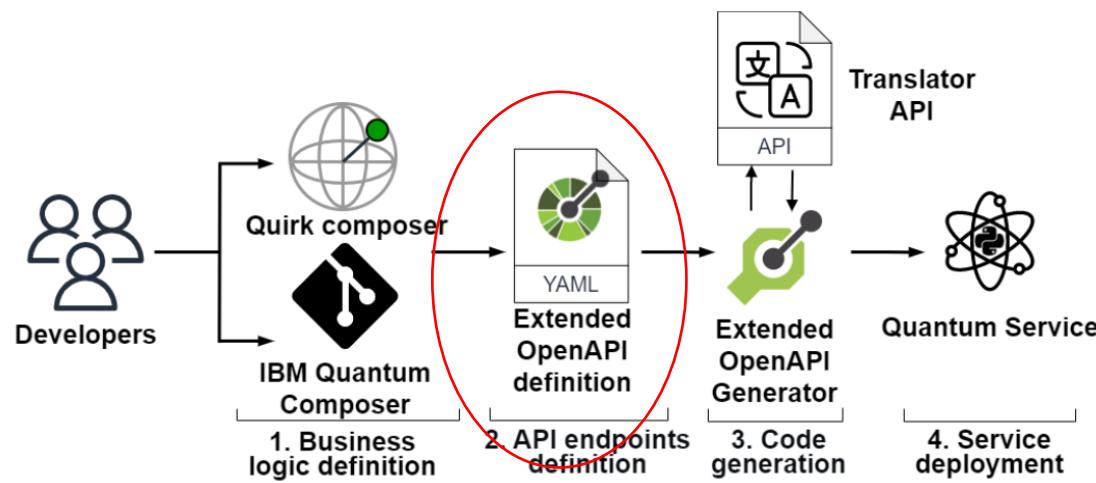
The screenshot shows the IBM Quantum Composer interface with the following components:

- Operations Panel:** On the left, it displays various quantum gate icons: H, +, S, Z, T, T[†], S[†], P, RZ, R[†], I, √X, √X[†], Y, RX, RY, RXX, RZZ, U, RCCX, RC3X.
- Circuit Editor:** A 5-qubit circuit (q[0] to q[4]) and 1 classical register (c[0]). The circuit consists of:
 - q[0]: H gate
 - q[1]: + gate
- OpenQASM 2.0 Editor:** Shows the generated OpenQASM code:

```
OPENQASM 2.0;
include "qelib1.inc";
qreg q[4];
creg c[4];
h q[0];
cx q[0], q[1];
```

A red circle highlights this area.
- Probabilities Plot:** A bar chart showing the probability (%) of each computational basis state. The x-axis lists states from |0000> to |1111>. The y-axis ranges from 0 to 100. The plot shows two bars at approximately 50% for |0000> and |0111>.
- Q-sphere:** A 3D visualization of the Bloch sphere showing the state vector |0000> and |0111>. The sphere has axes labeled Phase (0 to π/2) and Phase angle (0 to 3π/2). A legend indicates that the blue dot represents the State and the orange line represents the Phase angle.

QUANTUM SERVICES GENERATION



In the second step, the quantum API is defined, for which an API contract must be established with OpenAPI.

QUANTUM SERVICES GENERATION

Definition of the API contract, in YAML format, using the Swagger Editor

The screenshot shows the Swagger Editor interface with the following details:

Swagger Editor (Supported by SMARTBEAR)

openapi: 3.0.2

```
1 openapi: 3.0.2
2 info:
3   title: Quantum Services
4   description: This API enables the generation of quantum services.
5   version: '1.0'
6   termsOfService: 'https://quantum.spilab.es/terms'
7   contact:
8     name: Spilab
9     url: 'https://spilab.es'
10    email: info@spilab.es
11  tags:
12    - name: quantum
13      description: Everything about your Quantum Service
14    - name: quantum_code
15      description: Everything about your quantum service code and
16        circuits
17  paths:
18    /circuit/grover:
19      get:
20        tags:
21          - quantum_code
22          summary: Get the circuit implementation of Grover Algorithm
23          description: ''
24          operationId: grover_circuit
25          responses:
26            '200':
27              description: successful operation
28            '405':
29              description: Invalid execution
30            x-quantumCode: 'https://algassert.com/quirk#circuit={"cols"
31              :[[["H","H"]],[["X","X"]],[["*","Z"]],[["X","X"]],[["H","H"]],[["Z","Z"]
32              ,["*","Z"]],[["H","H"]],[["Measure","Measure"]]]}'
```

Quantum Services 1.0 OAS3

This API enables the generation of quantum services.

Terms of service
Spilab - Website
Send email to Spilab
Find out more about Swagger

quantum Everything about your Quantum Service

GET /**findByCategory** Finds quantum service by category

quantum_code Everything about your quantum service code and circuits

GET /**circuit/grover** Get the circuit implementation of Grover Algorithm

QUANTUM SERVICES GENERATION

Two custom variables, one to indicate the link to the quantum circuit, and the other to indicate the provider where we want to run it.

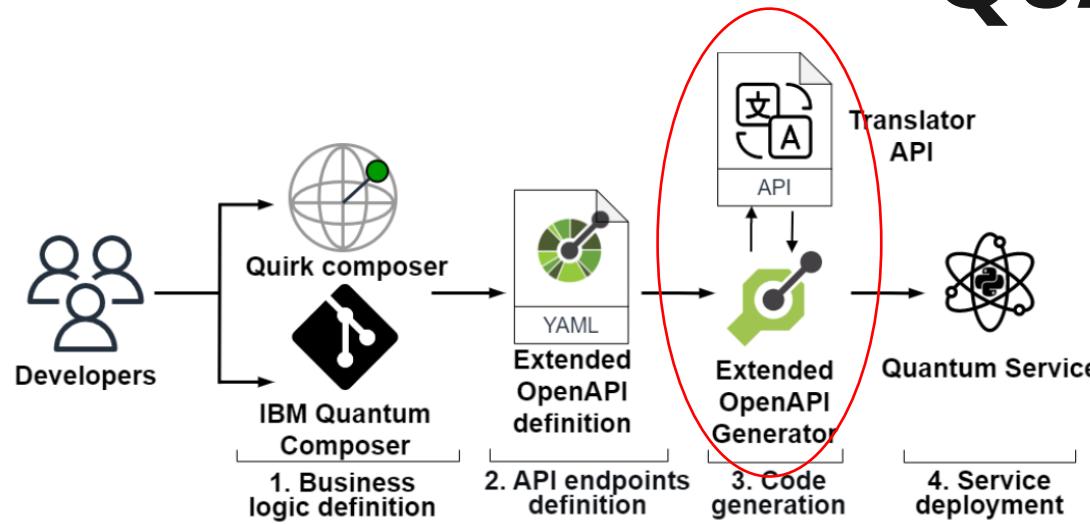
```
 30      description: number of shots
 31
 32      required: true
 33      style: form
 34      explode: false
 35      deprecated: true
 36      schema:
 37          type: number
 38
 39      responses:
 40          '200':
 41              description: successful operation
 42
 43          '405':
 44              description: Invalid execution
 45              x-quantumCode: 'QISKit-URL'
 46              x-quantumProvider: 'aws'
 47
 48      /circuit/ShorIBMqiskit:
 49
 50          get:
 51              tags:
```

QUANTUM SERVICES GENERATION

Two custom variables, one to indicate the link to the quantum circuit, and the other to indicate the provider where we want to run it.

```
paths:
  /circuit/randomAWS:
    get:
      tags:
        - quantum_code
      summary: Get the circuit implementation of Grover Algorithm
      description: ''
      operationId: grover_circuitAWS
      parameters:
        - name: machine
        - name: shots
      responses:
        '200':
          description: successful operation
        '405':
          description: Invalid execution
      x-quantumCode: 'QISKit-URL'
      x-quantumProvider: 'aws'
```

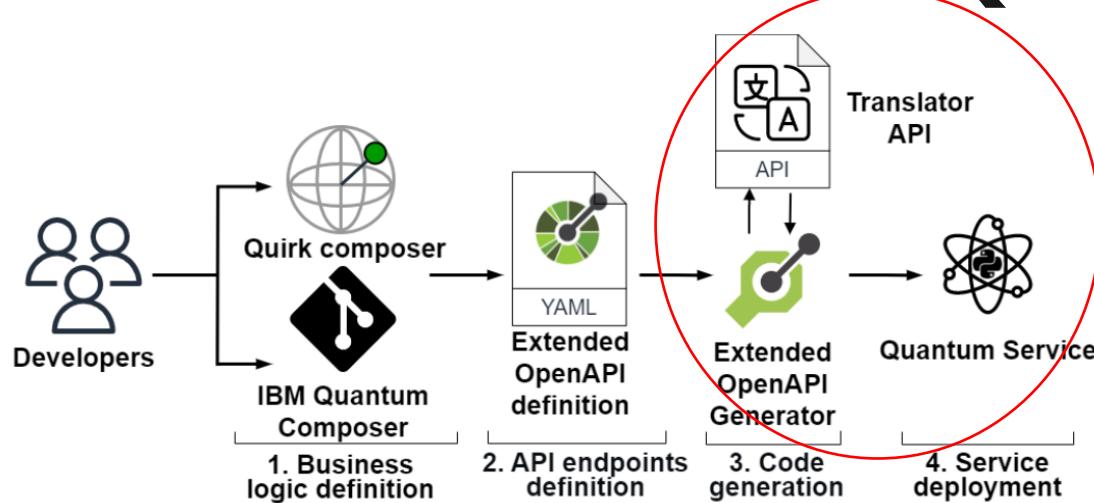
QUANTUM SERVICES GENERATION



In the third step, the extension of the OpenAPI Code Generator is used to generate the source code of the quantum services.

It uses our Translator API to compose the code from the provided quantum circuit for the selected provider.

QUANTUM SERVICES GENERATION



PATH PARAMETERS

*framework string python-quantum
framework

BODY DATA (required)

parameters

EXAMPLE MODEL

application/json

```
{
  "openAPIUrl": "https://raw.githubusercontent.com/QuantumOpenAPI/main/openapi_quantum.yaml",
}
```

from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit, execute, IBMQ] Qiskit libraries for quantum computing

```
def random_circuit(api_token): # noqa: E501
    """Get the circuit implementation for random numbers
    # noqa: E501
    :param api_token: API Token
    :type api_token: str
```

Classical wrapping endpoint service

```
:rtype: None
"""
IBMQ.enable_account(api_token)
provider = IBMQ.get_provider(hub='ibm-q')

q = QuantumRegister(16,'q')
c = ClassicalRegister(16,'c')
circuit = QuantumCircuit(q,c)
circuit.h(q) # Applies hadamard gate to all qubits
circuit.measure(q,c) # Measures all qubits

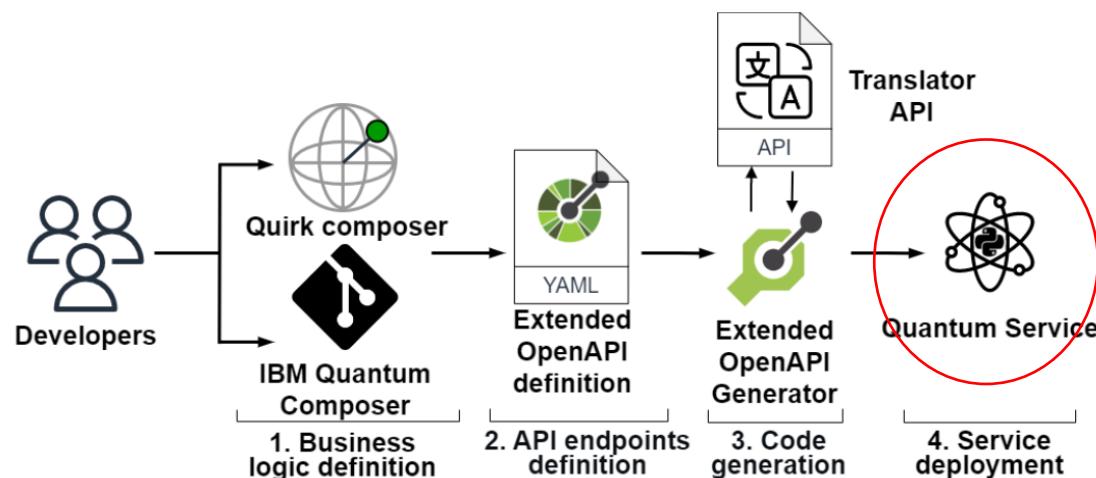
backend = provider.get_backend('ibmq_qasm_simulator')
job = execute(circuit, backend, shots=1)

result = job.result()
counts = result.get_counts(circuit)

return counts
```

Random Numbers quantum circuit

QUANTUM SERVICES GENERATION



These generated quantum services can be deployed and invoked (fourth step) via RESTful calls to the endpoints encapsulating the quantum algorithm code.

Quantum Services 1.0 OAS

/openapi.json

This API enables the generation of quantum services.

Terms of service
Spilab - Website
Send email to Spilab
Find out more about Swagger

Servers

quantum Everything about your Quantum Service

GET /findByCategory Finds quantum service by category

quantum_code Everything about your quantum service code and circuits

GET /circuit/grover Get the circuit implementation of Grover Algorithm

SCHEME :// HOST [":" PORT] [PATH ["?" QUERY]]
GET http://qapps.unex.es:8081/circuit/grover length: 40 byte(s)

Send ▾

QUERY PARAMETERS

HEADERS ^② Form ▾ BODY ^②
+ Add header Add authorization XHR does not allow payloads for GET request.

Response Cache Detected - Elapsed Time: 75ms

200 OK

HEADERS ^② pretty ▾ BODY ^② pretty ▾

Server: Werkzeug/2.1.2 Python/3.7.6
Date: Thu, 30 Jun 2022 10:42:47 GMT +3m

00: 100

Hands-on!



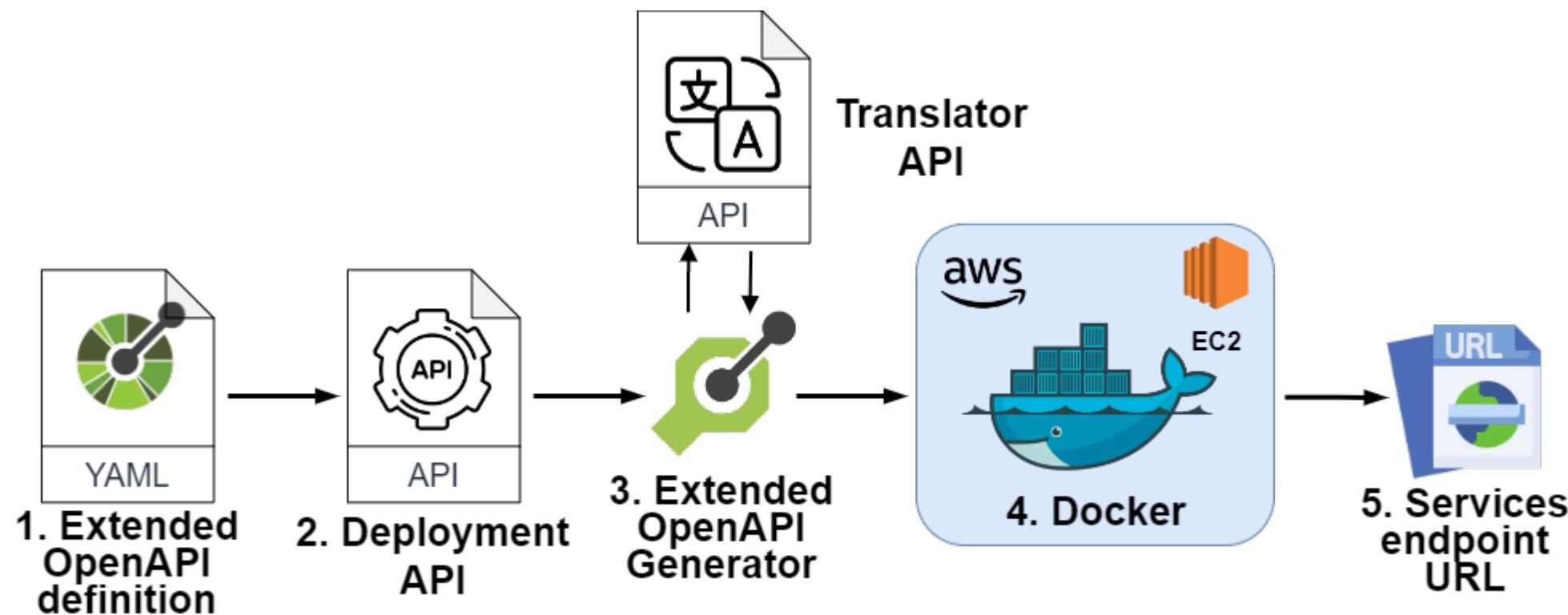
bit.ly/icweQuantumTutorial

QUANTUM SERVICE DEPLOYMENT

Once we have generated quantum services, which can be deployed manually, **why not do something to automate it?**

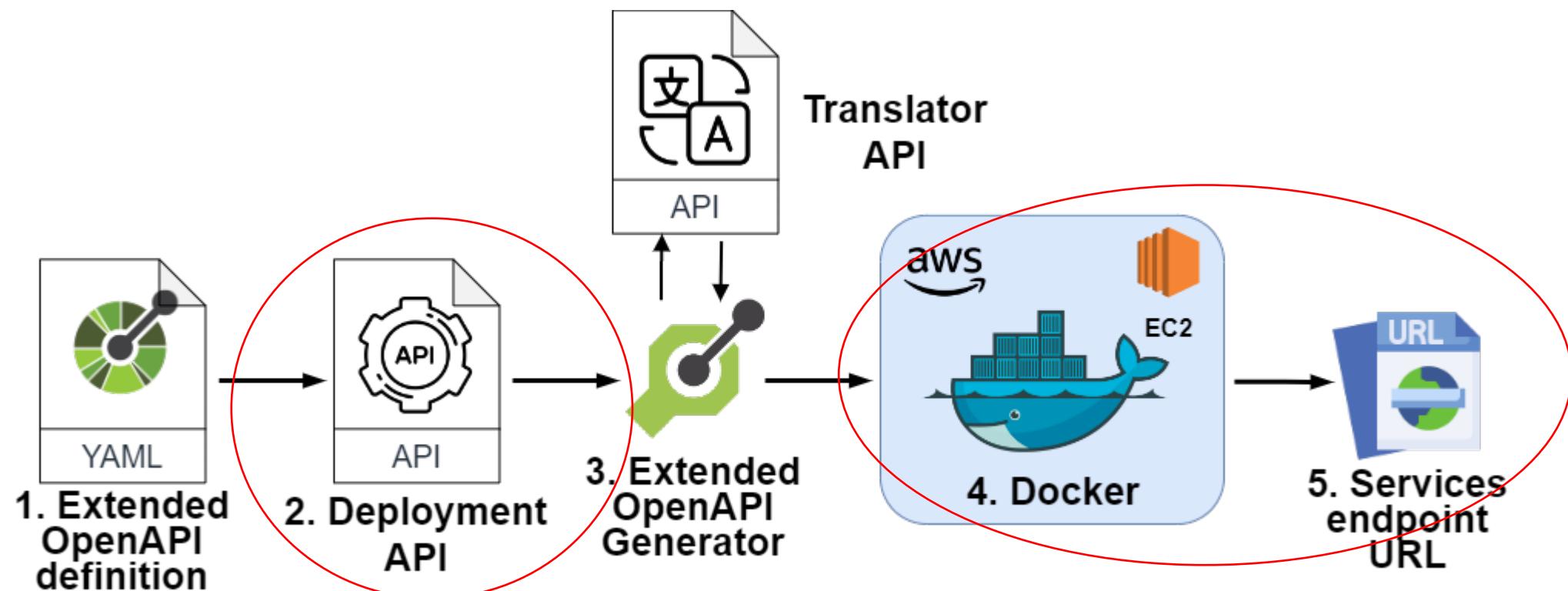
QUANTUM SERVICE DEPLOYMENT

Once we have generated quantum services, which can be deployed manually, why not do something to automate it?

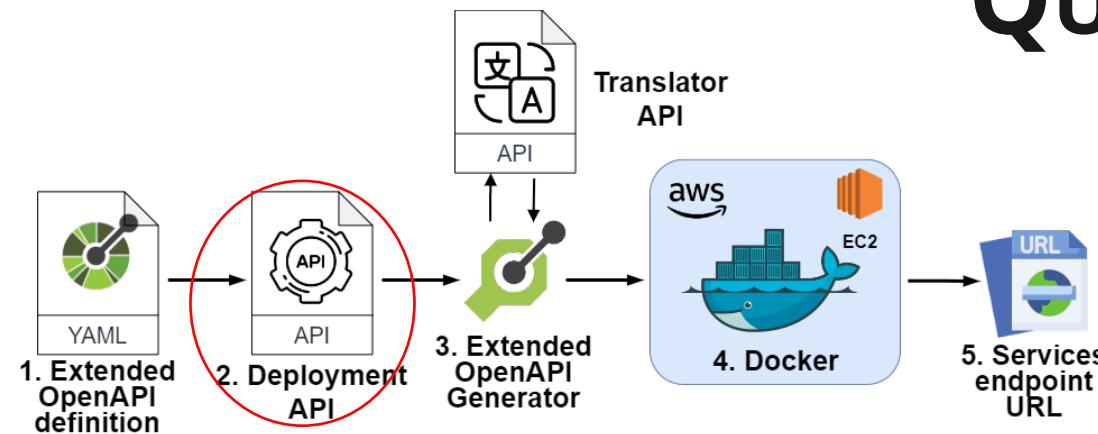


QUANTUM SERVICE DEPLOYMENT

Once we have generated quantum services, which can be deployed manually, why not do something to automate it?



QUANTUM SERVICE DEPLOYMENT



The Deployment API receives the YAML file based on the modification of the previous OpenAPI specification and the necessary parameters for the configuration of the providers (e.g. AWS keys).

QUANTUM SERVICE DEPLOYMENT

Docker Hub interface showing the repository `jromero236/quantumservices`.

Header navigation: Explore, Repositories, Organizations, Help, Upgrade, jromero236.

Breadcrumbs: jromero236 > Repositories > quantumservices > General.

Message: Using 0 of 1 private repositories. [Get more](#)

Repository tabs: General, Tags, Builds, Collaborators, Webhooks, Settings. **General** is selected.

General tab content:

- Add a short description for this repository** (info icon) [Update](#)
The short description is used to index your content on Docker Hub and in search engines. It's visible to users in search results.
- jromero236 / quantumservices**
- Description**
This repository does not have a description [Edit](#)
- Last pushed: 5 months ago**
- Docker commands**
To push a new tag to this repository,
`docker push jromero236/quantumservices:tagname`
- [Public View](#)

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	3 months ago	5 months ago

[See all](#) [Go to Advanced Image Management](#)

Automated Builds

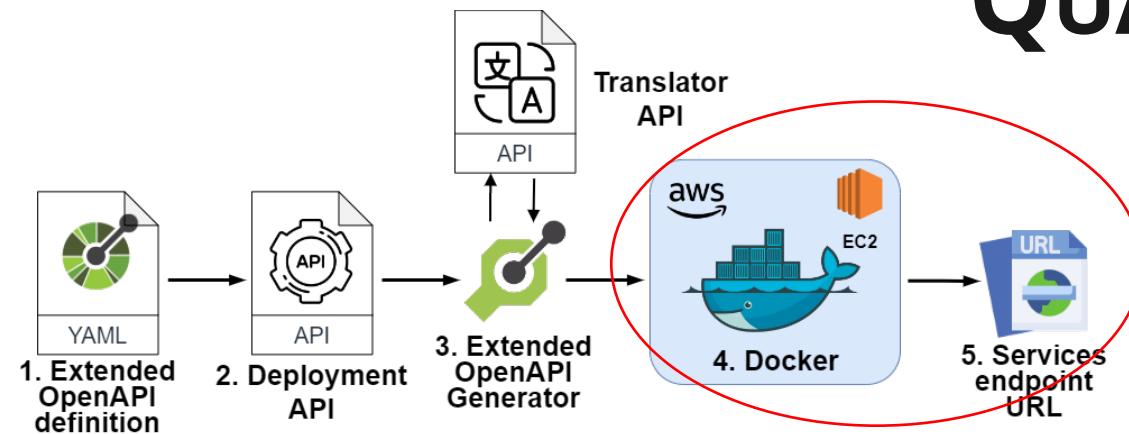
Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. [Read more about automated builds](#).

[Upgrade](#)



QUANTUM SERVICE DEPLOYMENT



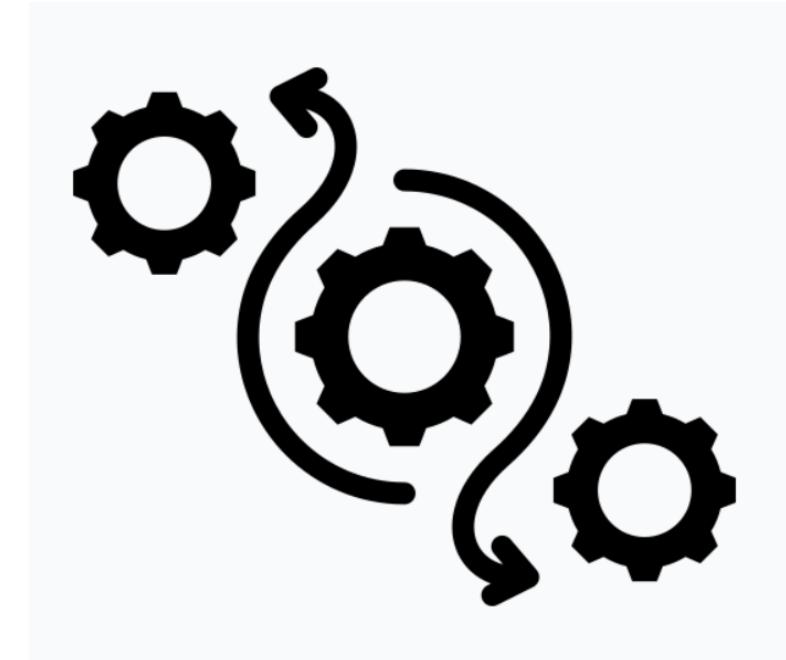
A screenshot of a REST API response interface. The top navigation bar includes "Body", "Cookies", "Headers (5)", "Test Results", "Status: 200 OK", "Time: 8.62 s", "Size: 304 B", and "Save Response". Below the navigation is a toolbar with "Pretty", "Raw", "Preview", "Visualize", "HTML", and a copy icon. The main content area shows the following HTML code:

```
1 <html>
2 <h1 href='http://quantumservicesdeployment.spilab.es:8088/ui'>http://quantumservicesdeployment.spilab.es:8088/ui</h1>
3 </html>
```

Hands-on!

WORKFLOW FOR THE CONTINUOUS DEPLOYMENT

Why not bring generation and deployment together and automate it?



WORKFLOW FOR THE CONTINUOUS DEPLOYMENT

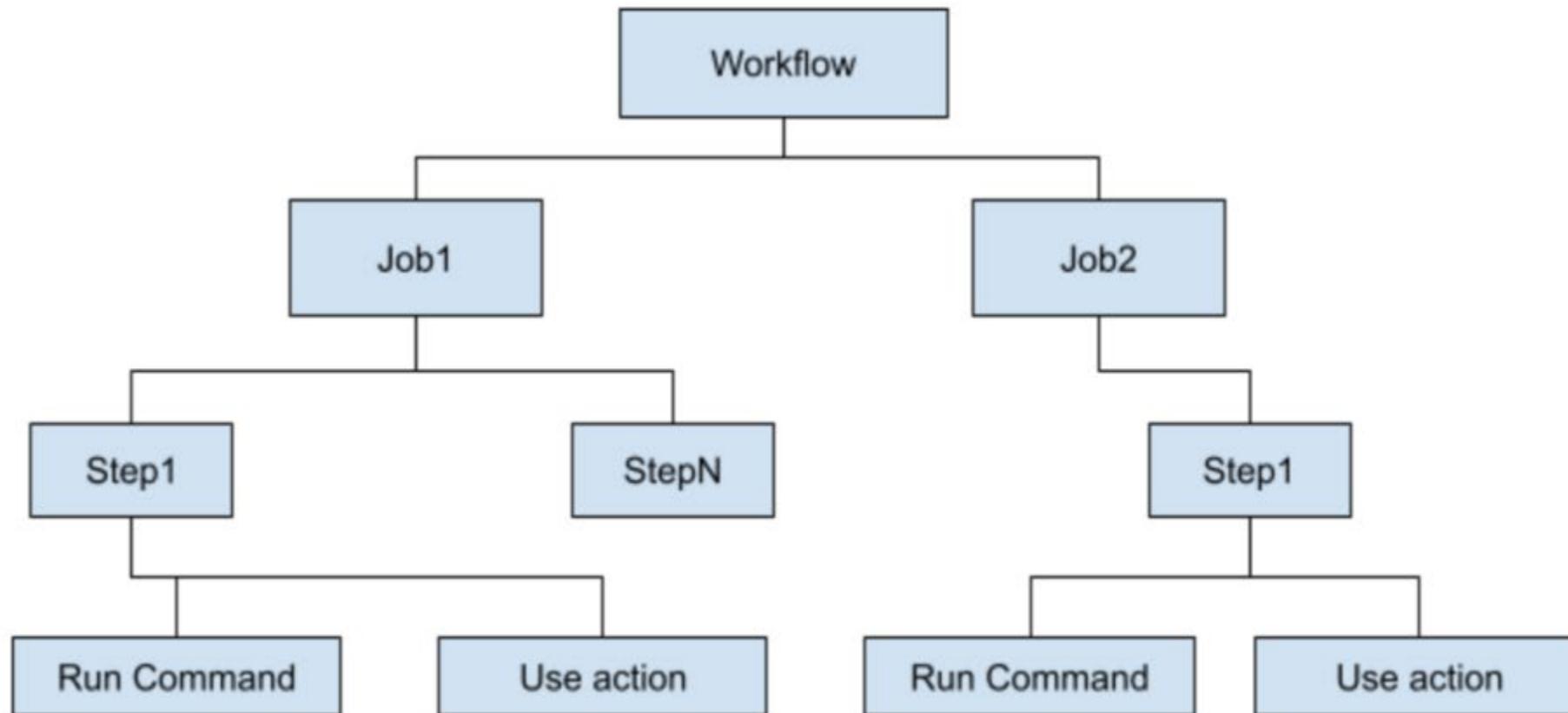


GitHub Actions

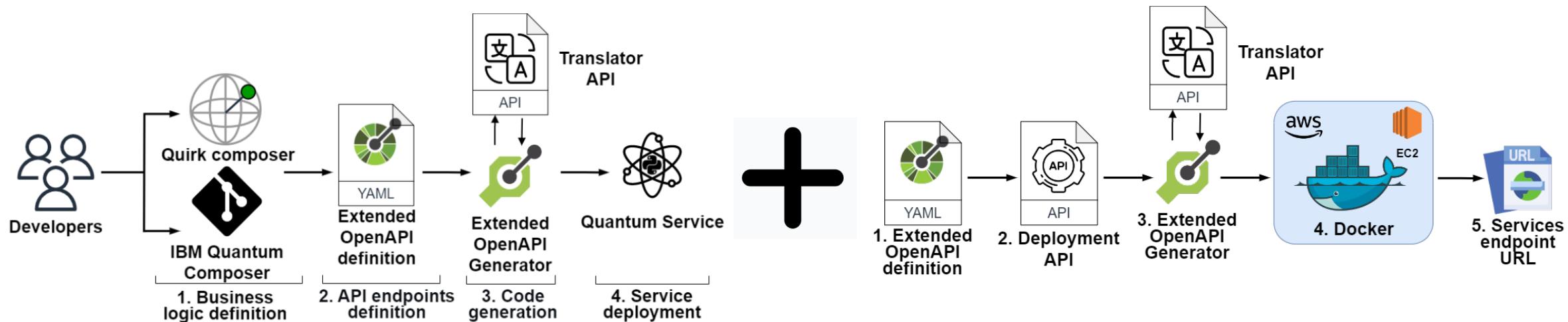
- 1. Automation:** GitHub Actions allows you to automate workflows in software development projects.
- 2. YAML-based Configuration:** Workflows are defined using YAML configuration files stored in the repository.
- 3. Event-Driven:** Workflows can be triggered by events like code pushes, pull requests, or scheduled intervals.
- 4. Integrations:** GitHub Actions seamlessly integrates with other GitHub features, enabling collaboration and streamlining the development process.

WORKFLOW FOR THE CONTINUOUS DEPLOYMENT

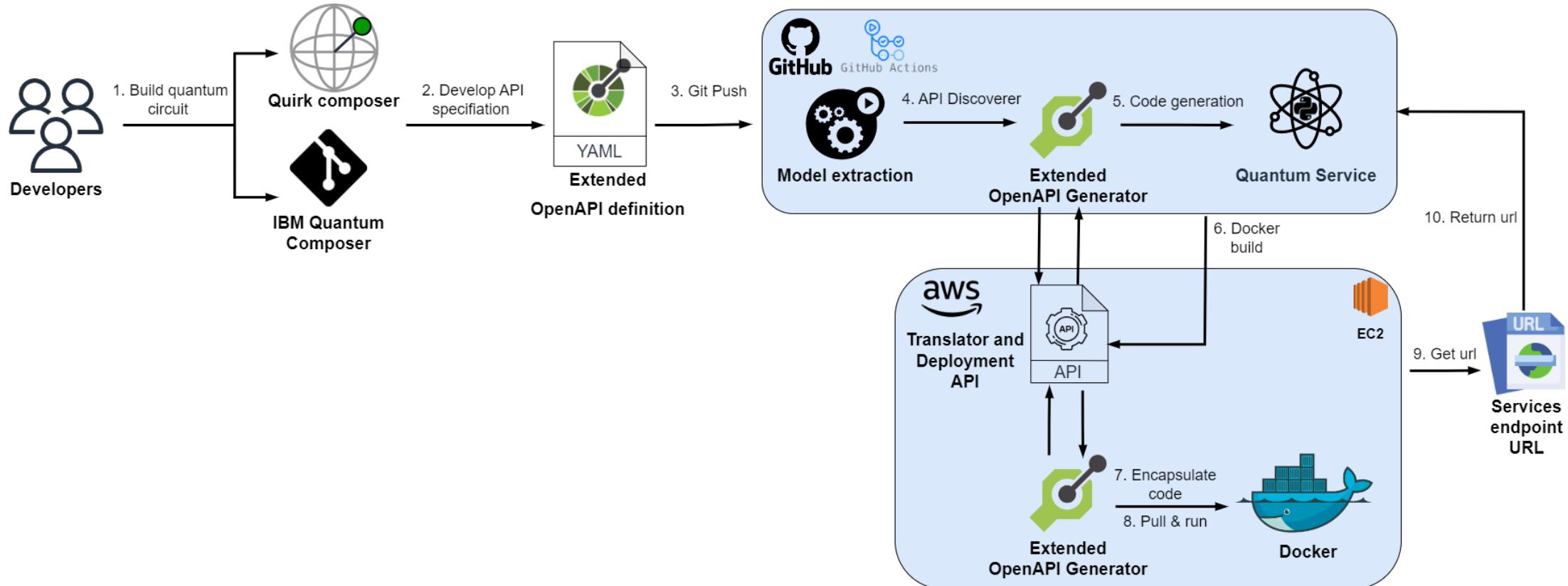
GitHub Actions



WORKFLOW FOR THE CONTINUOUS DEPLOYMENT



WORKFLOW FOR THE CONTINUOUS DEPLOYMENT



WORKFLOW FOR THE CONTINUOUS DEPLOYMENT

The screenshot shows a GitHub Actions workflow run for a repository named "Update openapi.yaml - javierrome". The workflow has a single job named "validate (3.8)". The job status is "succeeded last week in 1m 26s". The job details are as follows:

- Set up job**: Succeeded (3s)
- Build cpina/github-action-push-to-another-repository@main**: Succeeded (6s)
- Run actions/checkout@v3**: Succeeded (3s)
- Build python API with YAML**: Succeeded (0s)
- Set up Python 3.8**: Succeeded (0s)
- Install dependencies**: Succeeded (55s)
- Build with Maven**: Succeeded (6s)
- Prepare Sonar files**: Succeeded (1s)
- Pushes to another repository**: Succeeded (3s)
- Check python project**: Succeeded (0s)

Under the "Deploy Stage" section, the command output is displayed:

```
1 ► Run curl -X POST -F 'name=python' -F 'yaml=https://raw.githubusercontent.com/javierrome236/quantumDeployment/main/openapi.yaml' http://quantumservicesdeployment.spilab.es:8081/docker
13   % Total    % Received % Xferd  Average Speed   Time     Time   Current
14          Dload  Upload Total Spent   Left Speed
15
16  0    0    0    0    0    0      0  --::-- --::-- --::--  0
17 100  321  0    0  100  321  0   263  0:00:01  0:00:01  263
18 100  321  0    0  100  321  0   144  0:00:02  0:00:02  144
19 100  321  0    0  100  321  0   99  0:00:03  0:00:03  99
20 100  321  0    0  100  321  0   76  0:00:04  0:00:04  76
21 100  451  100 130  100  321  26  64  0:00:05  0:00:04  90
22 100  451  100 130  100  321  26  64  0:00:05  0:00:04  34
23 <html><h1 href='http://quantumservicesdeployment.spilab.es:8085/ui'>http://quantumservicesdeployment.spilab.es:8085/ui</h1></html>
```

- Post Set up Python 3.8**: Succeeded (0s)
- Post Build python API with YAML**: Succeeded (0s)
- Post Run actions/checkout@v3**: Succeeded (0s)
- Complete job**: Succeeded (0s)

Hands-on!

Quantum Survey



**PLEASE FILL OUT THE
FOLLOWING SURVEY
IT WILL BE OF GREAT HELP TO US!!**

<https://forms.office.com/e/sQgqTi1LQd>

THANKS FOR ATTENTION!

23rd International
Conference on Web
Engineering - ICWE
Alicante (Spain)

Javier Romero-Álvarez
✉ jromero@unex.es

Jaime Alvarado-Valiente
✉ jaimeav@unex.es



SPILab
by QuercusSEG



UNIÓN EUROPEA
FONDO EUROPEO DE
DESARROLLO REGIONAL
"Una manera de hacer Europa"
JUNTA DE EXTREMADURA
Consejería de Economía, Ciencia y Agenda Digital

SPILab
by Quercus SEG

Software Engineering Group
QUERCUS
UNIVERSIDAD DE EXTREMADURA

Q **EX**
UNIVERSIDAD DE EXTREMADURA