Lab 1:
Your first circuit step-by-step with IBM Quantum

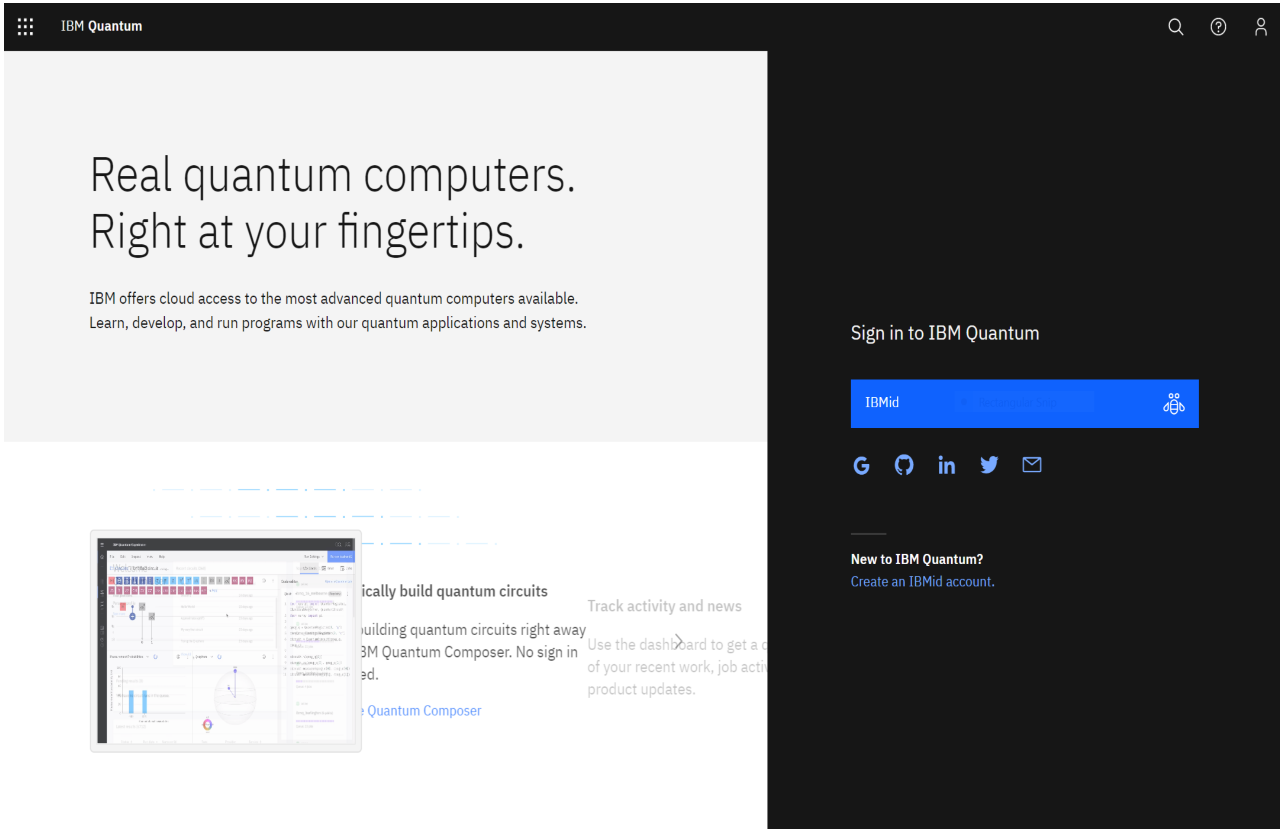
We are going to use throughout this course the environment offered by IBM which is called the IBM Quantum. In this exercise we are going to do together the simplest possible thing that one can do to a qubit: we are going to invert it. That is we are going to move it from quantum state $|0\rangle$ to quantum state $|1\rangle$, i.e. from pointing up to pointing down in the Bloch Sphere.

TASK: **To flip a single qubit from $|0\rangle$ to $|1\rangle$.**

(ONLINE) SOLUTION: **Create a quantum circuit with a single qubit and a X Gate in the online IBM Quantum platform**

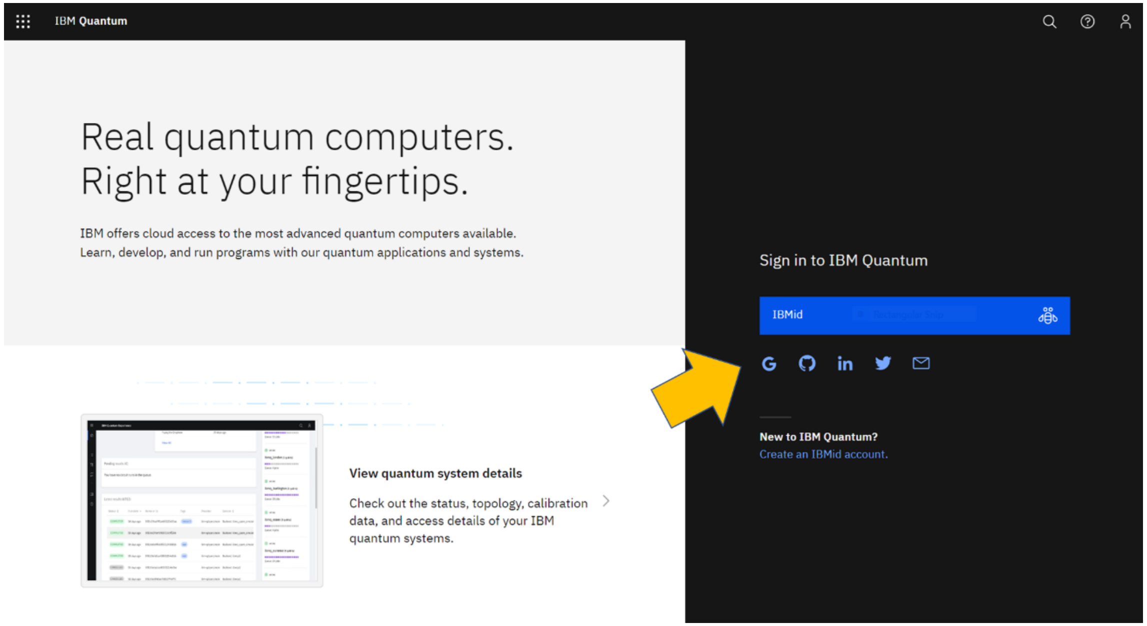
Step 1: Go to the IBM Quantum website

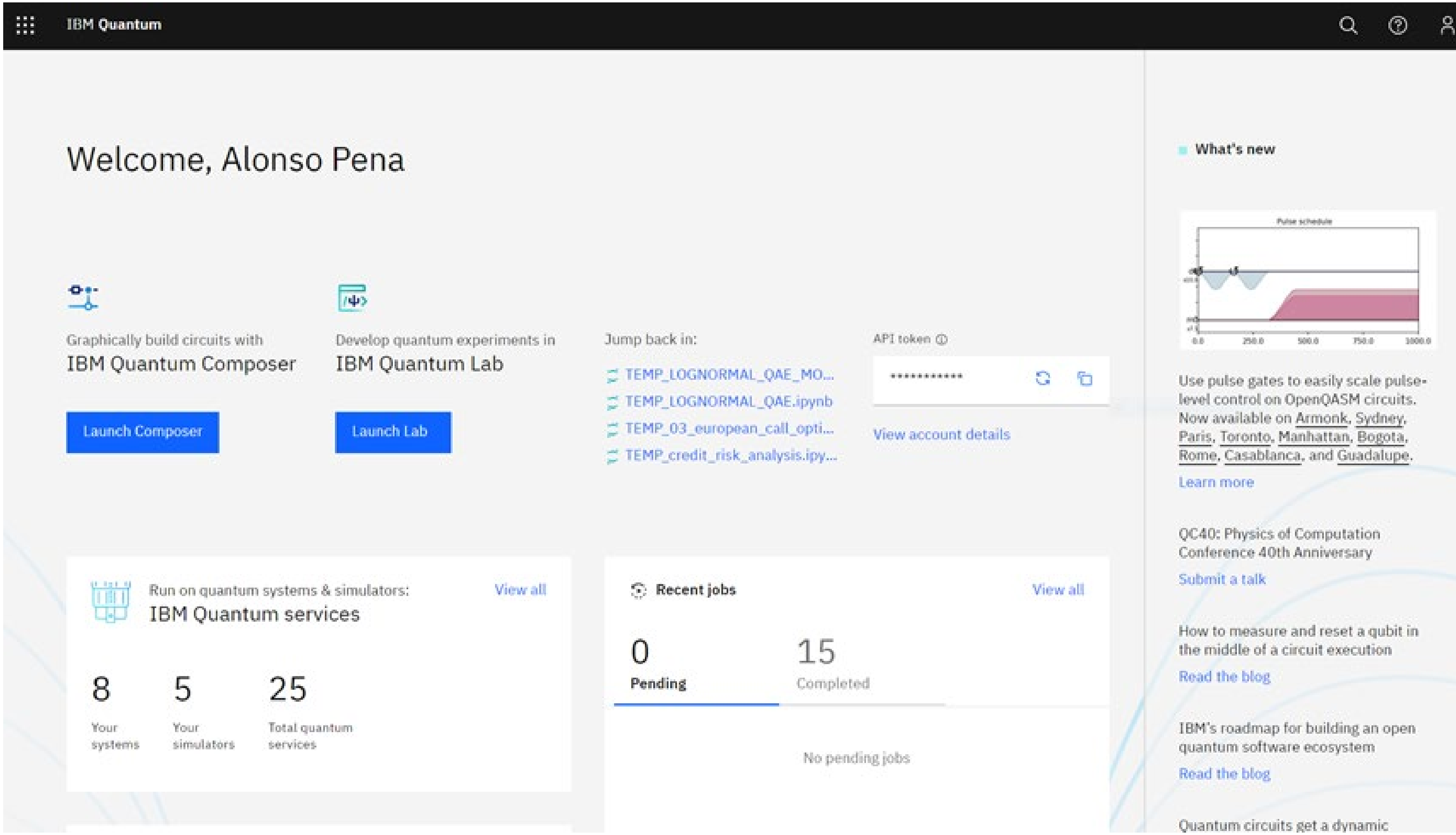
Visit the IBM Quantum Experience website https://quantum-computing.ibm.com/login. As illustrated in the next Figure, here is where you can access the various services offered by IBM for quantum computation.



Step 2: Register or sign-in into the system

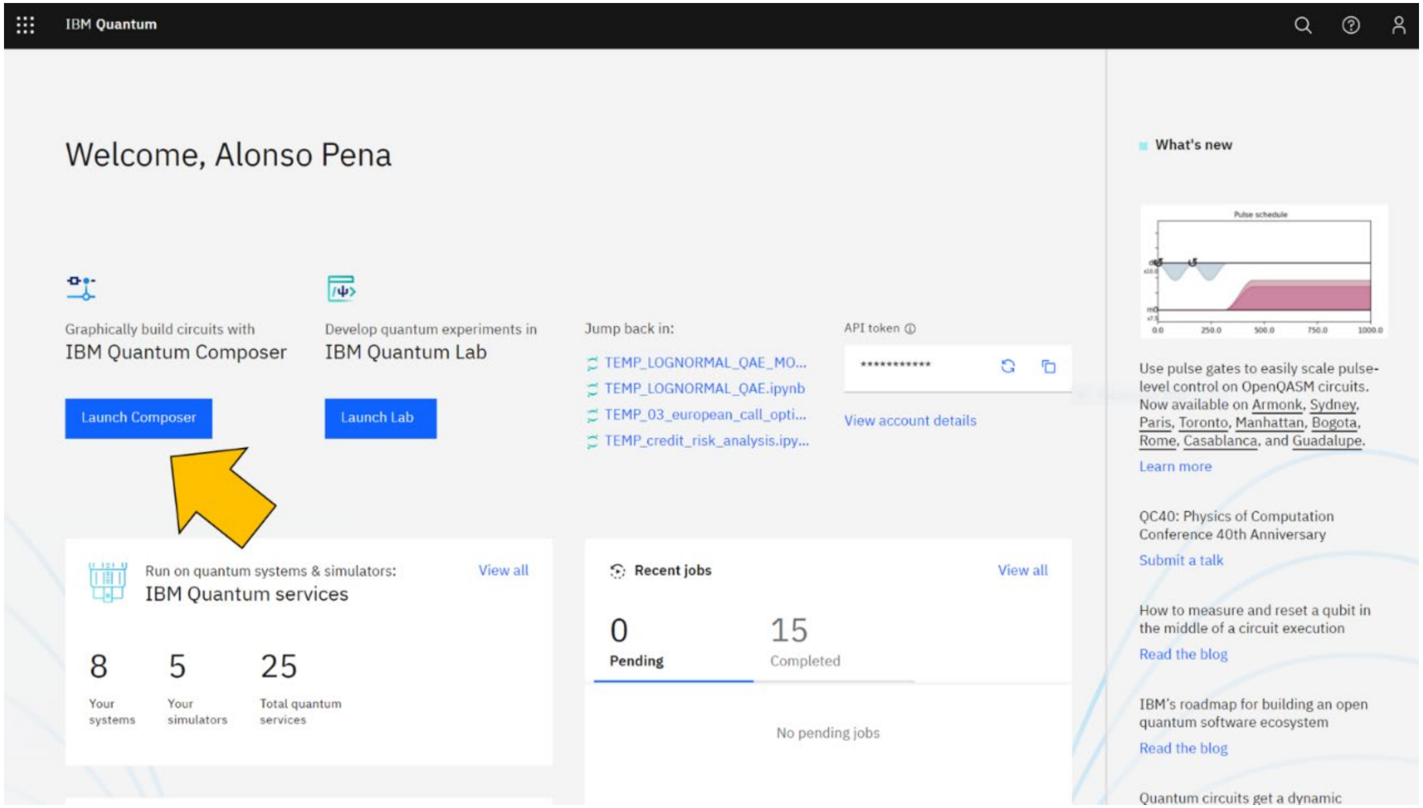
You ought to create a new **`IBMid`** which will allow you to use the environment. You can do this by filling the short registration form. Alternatively you can register by using your social profile from Google, LinkedIn or Twitter, amongst others.

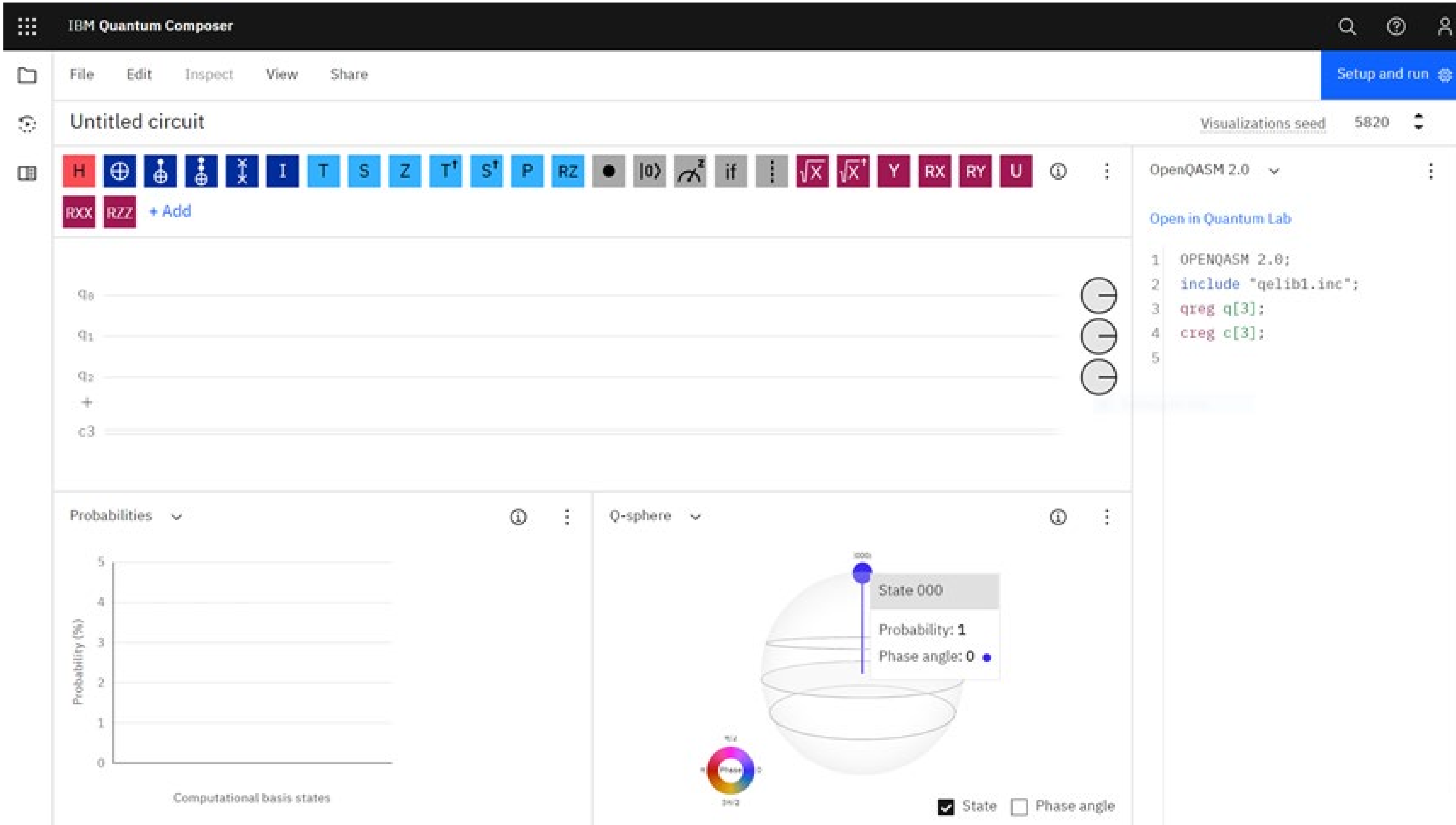



IBM Quantum
Welcome, Alonso Pena
Graphically build circuits with
IBM Quantum Composer
Launch Composer
Develop quantum experiments in
IBM Quantum Lab
Launch Lab
Jump back in:
TEMP_LOGNORMAL_QAE_MO...
TEMP_LOGNORMAL_QAE.ipynb
TEMP_03_european_call_opti...
TEMP_credit_risk_analysis.ipy...
API token
****
View account details
What's new
Pulse schedule
Use pulse gates to easily scale pulse-level control on OpenQASM circuits. Now available on Armonk, Sydney, Paris, Toronto, Manhattan, Bogota, Rome, Casablanca, and Guadalupe.
Learn more
QC40: Physics of Computation Conference 40th Anniversary
Submit a talk
How to measure and reset a qubit in the middle of a circuit execution
Read the blog
IBM's roadmap for building an open quantum software ecosystem
Read the blog
Quantum circuits get a dynamic
Run on quantum systems & simulators:
IBM Quantum services
View all
8
Your systems
5
Your simulators
25
Total quantum services
Recent jobs
View all
0
Pending
15
Completed
No pending jobs

Step 3: Go to the Quantum Composer

Go now to *IBM Quantum Composer* in order to construct our first circuit by clicking on the Launch Composer button as indicated in the following Figure.

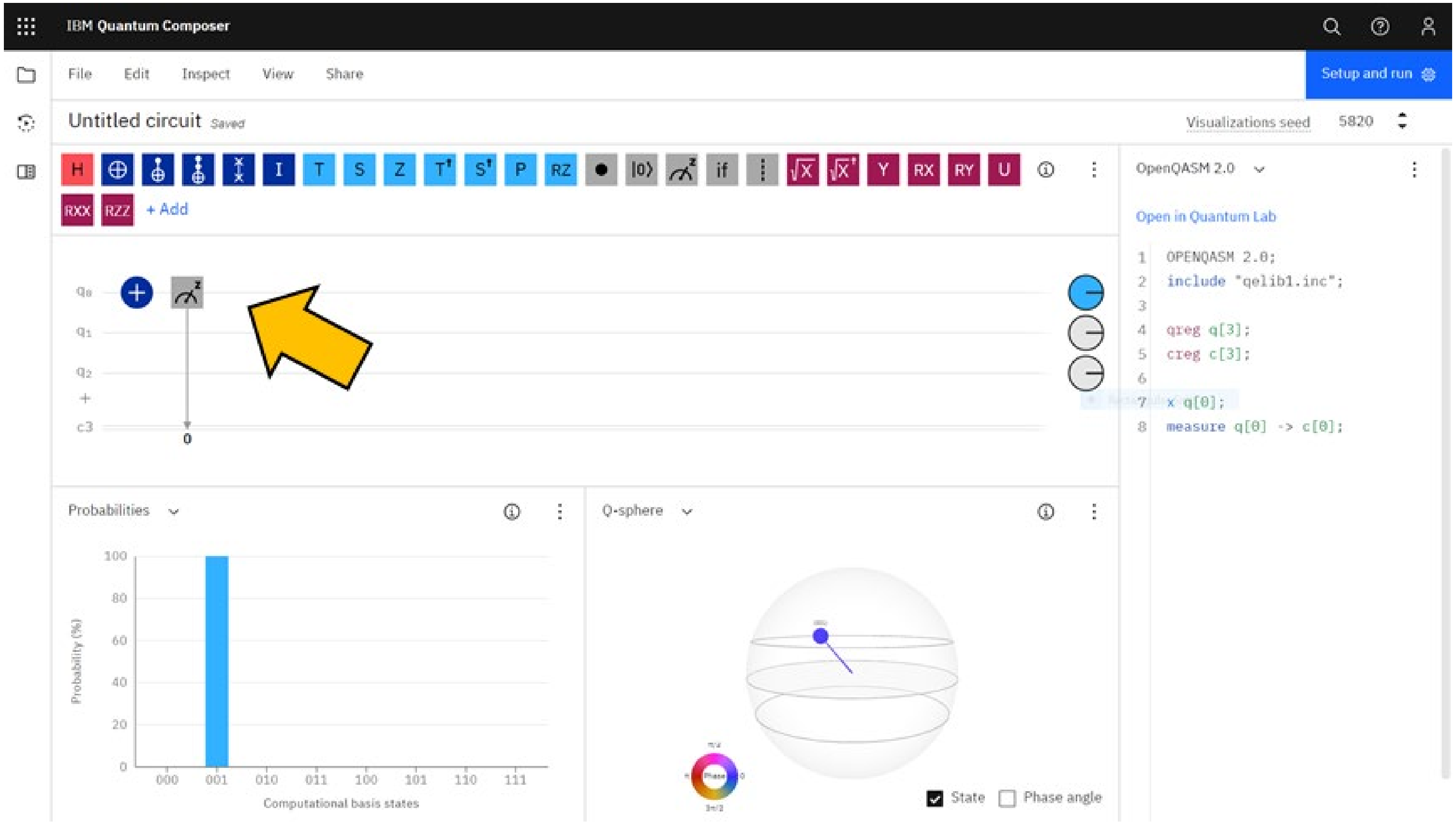



IBM Quantum Composer
File
Edit
Inspect
View
Share
Setup and run
Untitled circuit
Visualizations seed
5820
H
I
T
S
Z
P
RZ
if
Y
RX
RY
U
RXX
RZZ
+ Add
OpenQASM 2.0
Open in Quantum Lab
OPENQASM 2.0;
include "qelib1.inc";
qreg q[3];
creg c[3];
Probabilities
Probability (%)
Computational basis states
Q-sphere
State 000
Probability: 1
Phase angle: 0
State
Phase angle

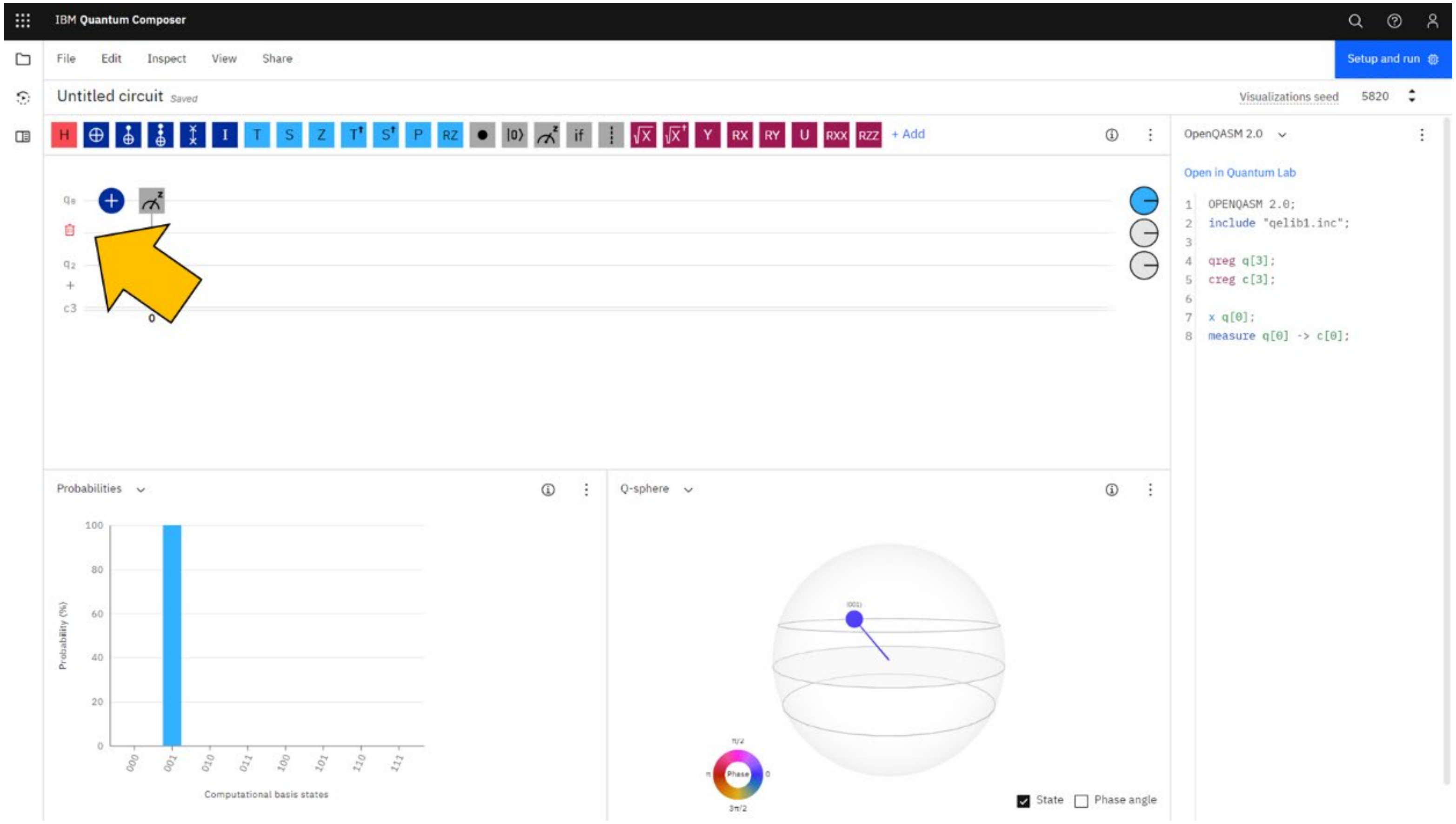
The coloured icons represent different types of quantum gates. The horizontal lines represent the cables to connect the quantum gates to the qubits and classical bits (indicated by *q* and *c*, respectively). The Bloch sphere is shown on the lower part as well as the statistical graph for the measurements. On the right the equivalent of the circuit is represented in a low-level code called QASM (Quantum Assembler Language).

Step 4: Create your first circuit

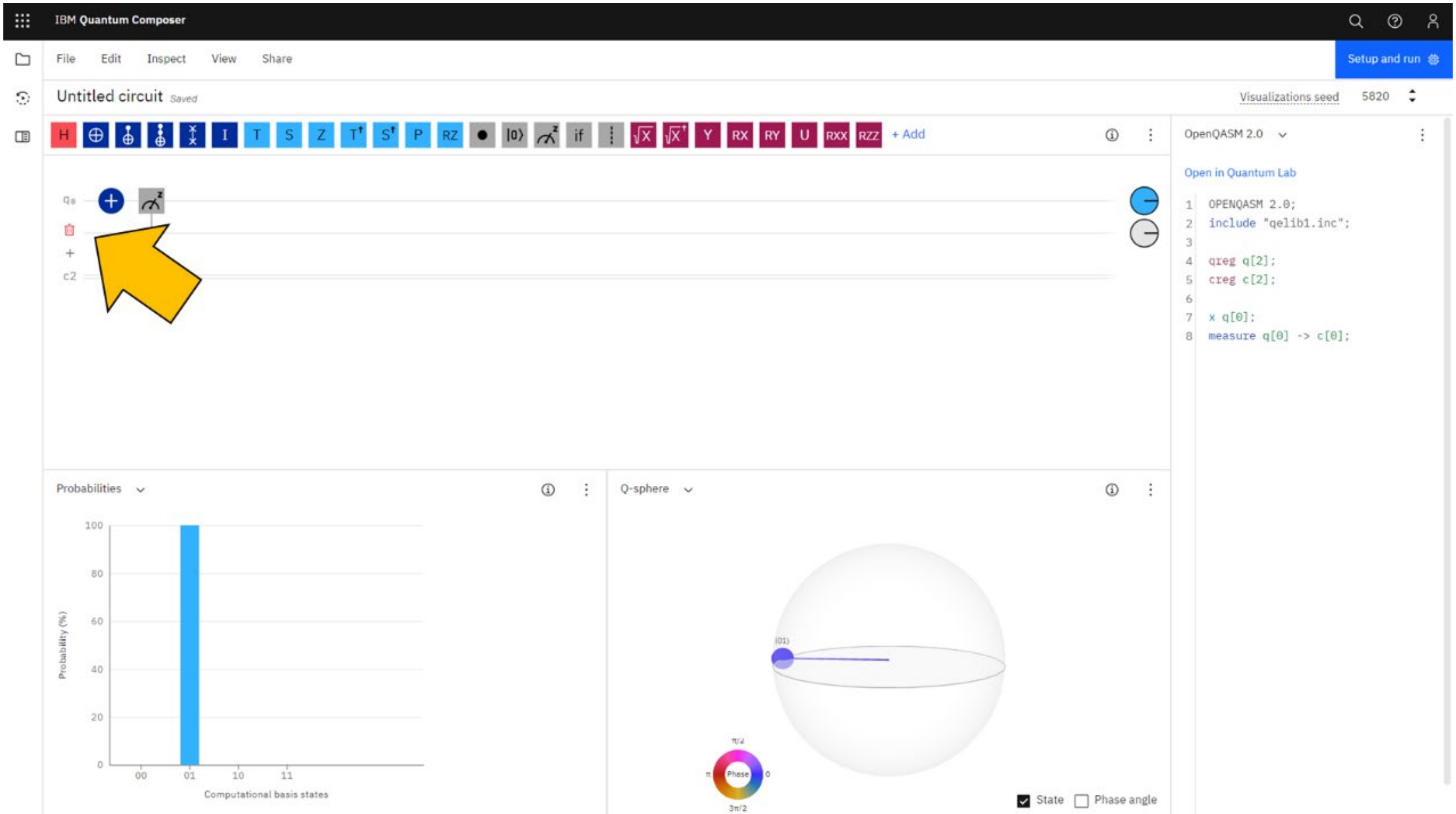
Into the lines indicating the circuit, drag the X gate (circle with cross icon) to q0, then the measurement to q0.



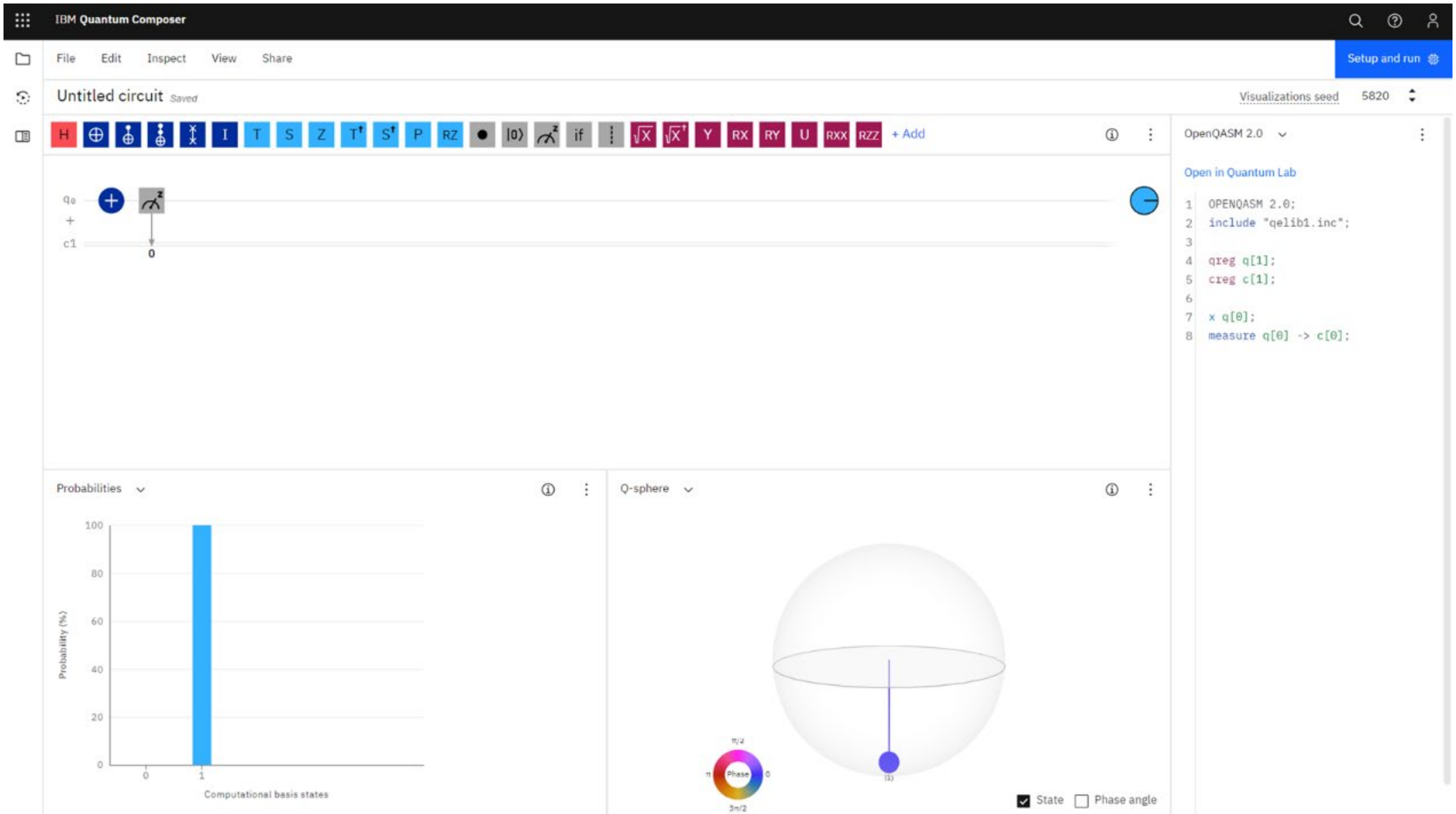
Now we need to eliminate the unnecessary qubits q1 and q2, that are present by default in the circuit design. By hovering directly above q1 a red rubbish bin icon will appear, click on it to delete:



Do the same for q2 to delete:

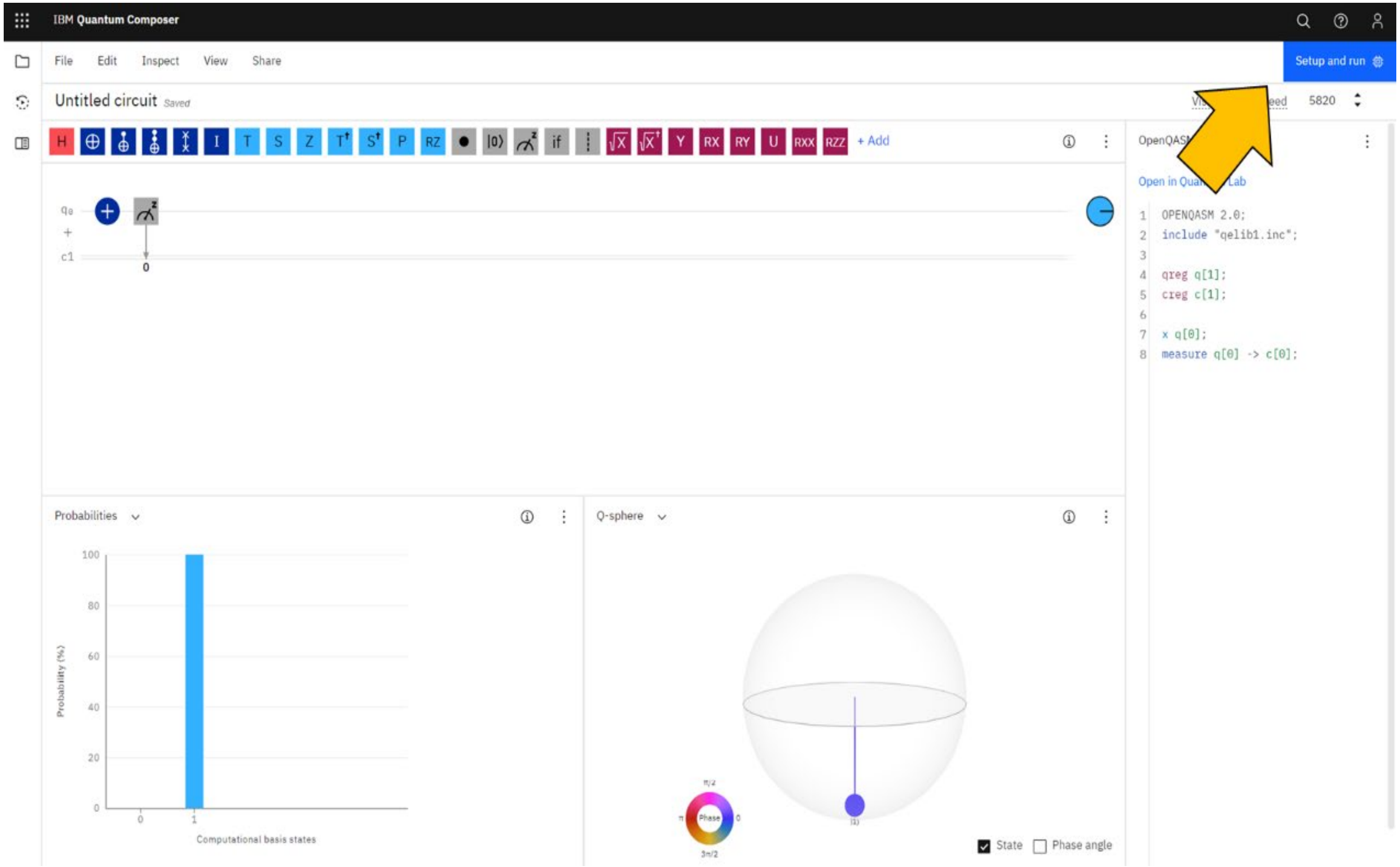


Your circuit is now looking like this

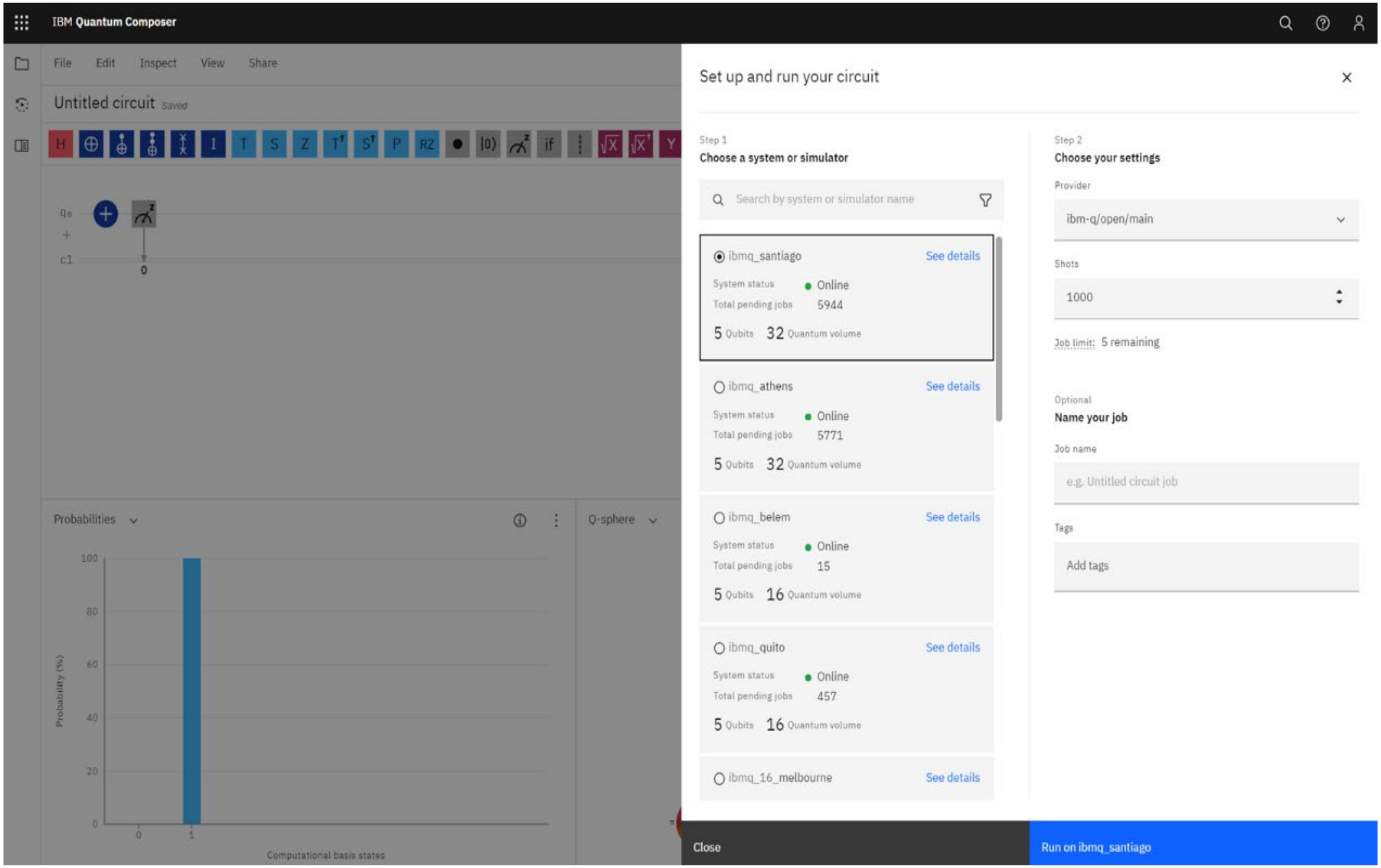


Step 5 : Setup your Circuit

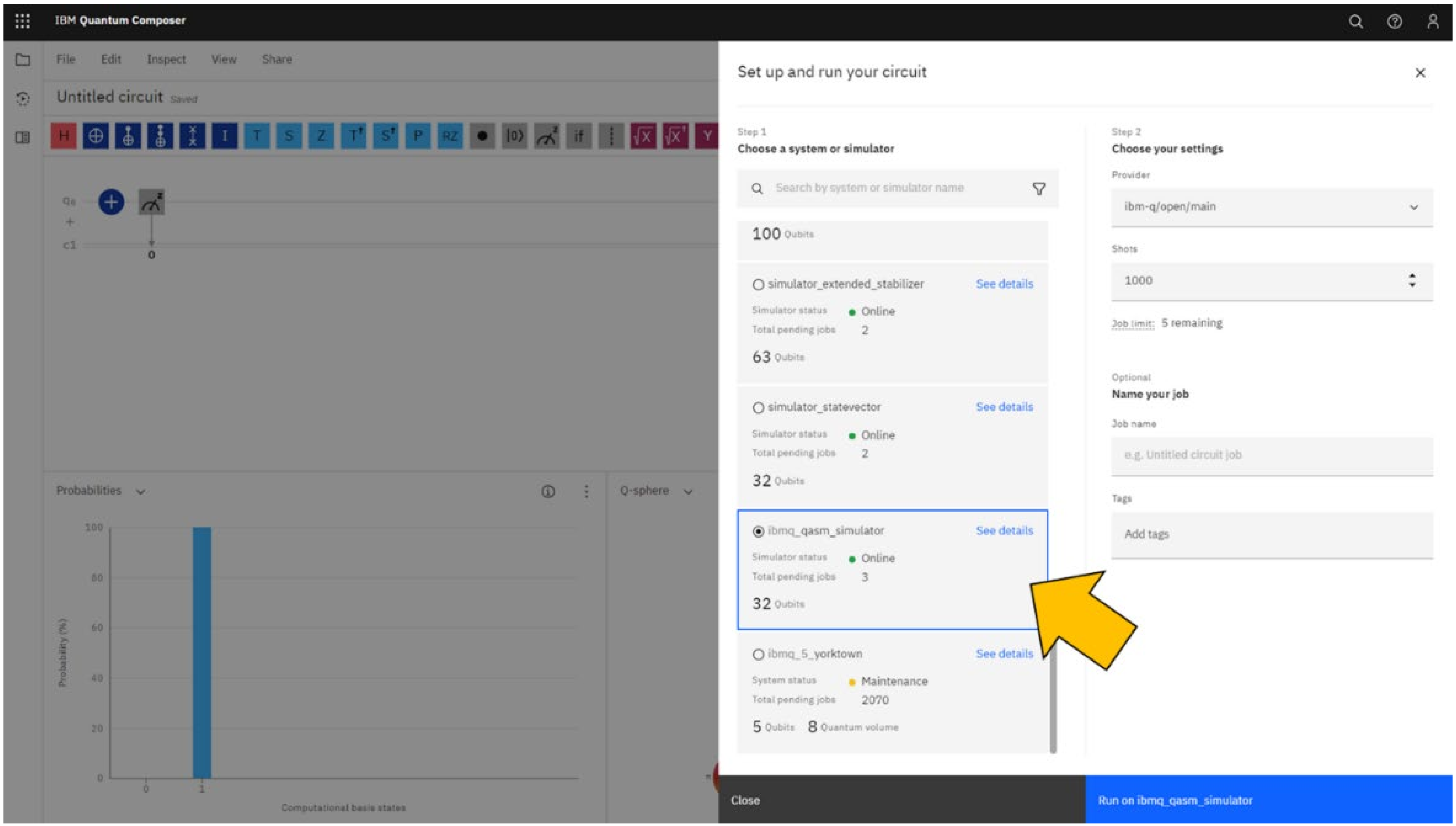
Select the parameters to run your circuit by clicking on the top right button:



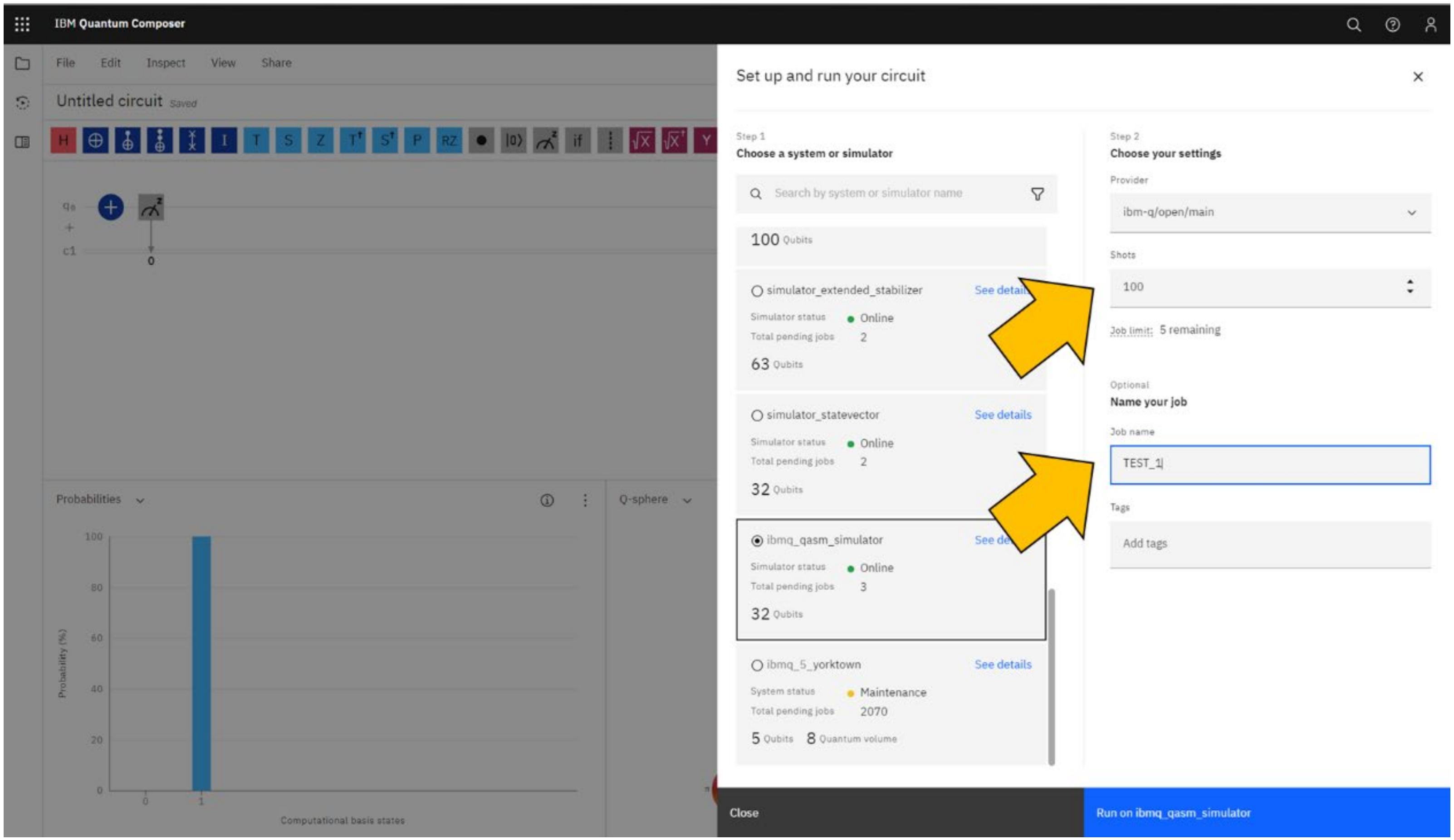
You should now see



Complete the information requested. In *Step 1: Choose a system or simulator*, select the **ibm_qasm_simulator**.

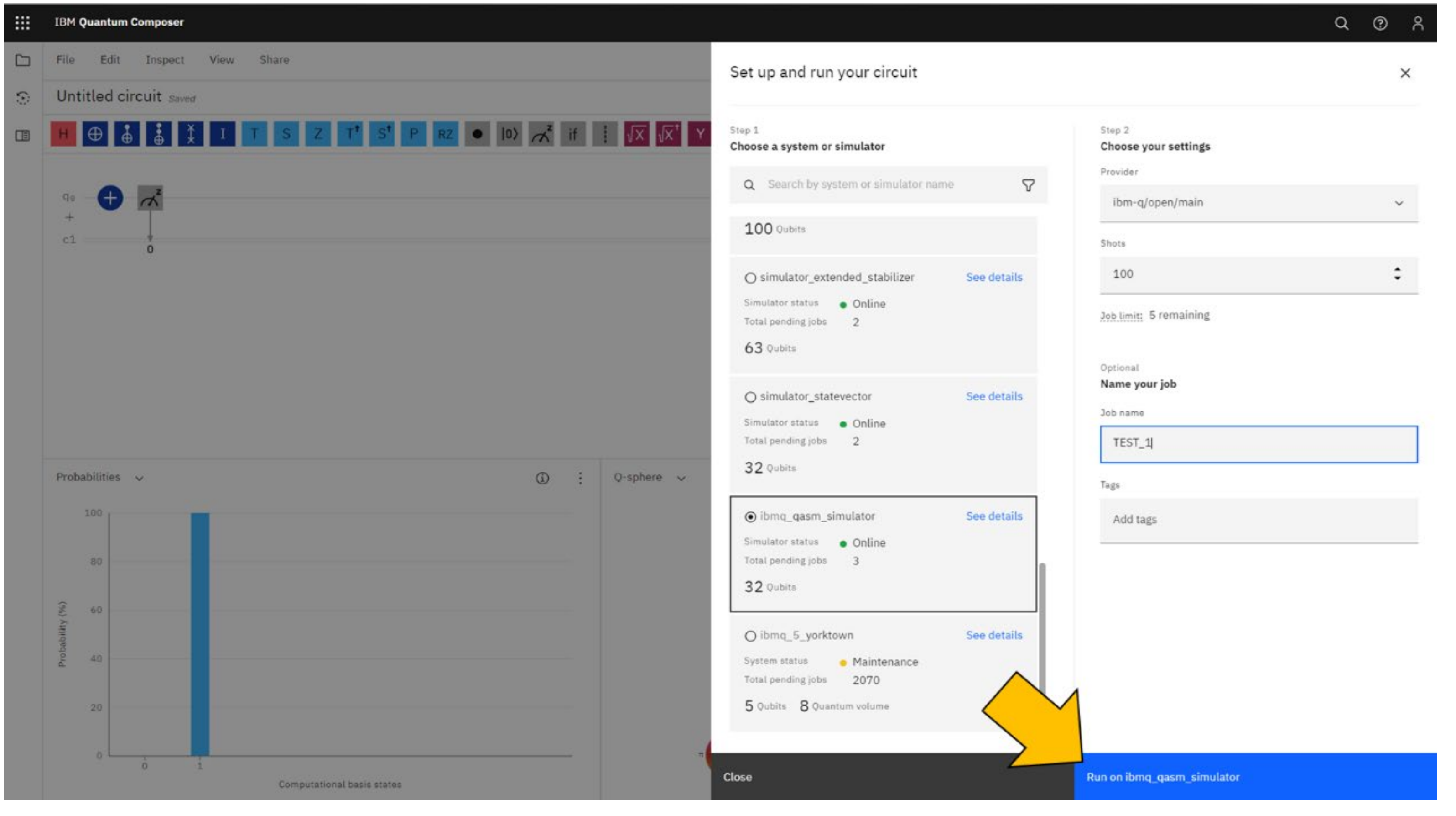


Then in *Step 2: Choose your settings*, type 100 to use 100 simulations or steps and name the circuit TEST_1.



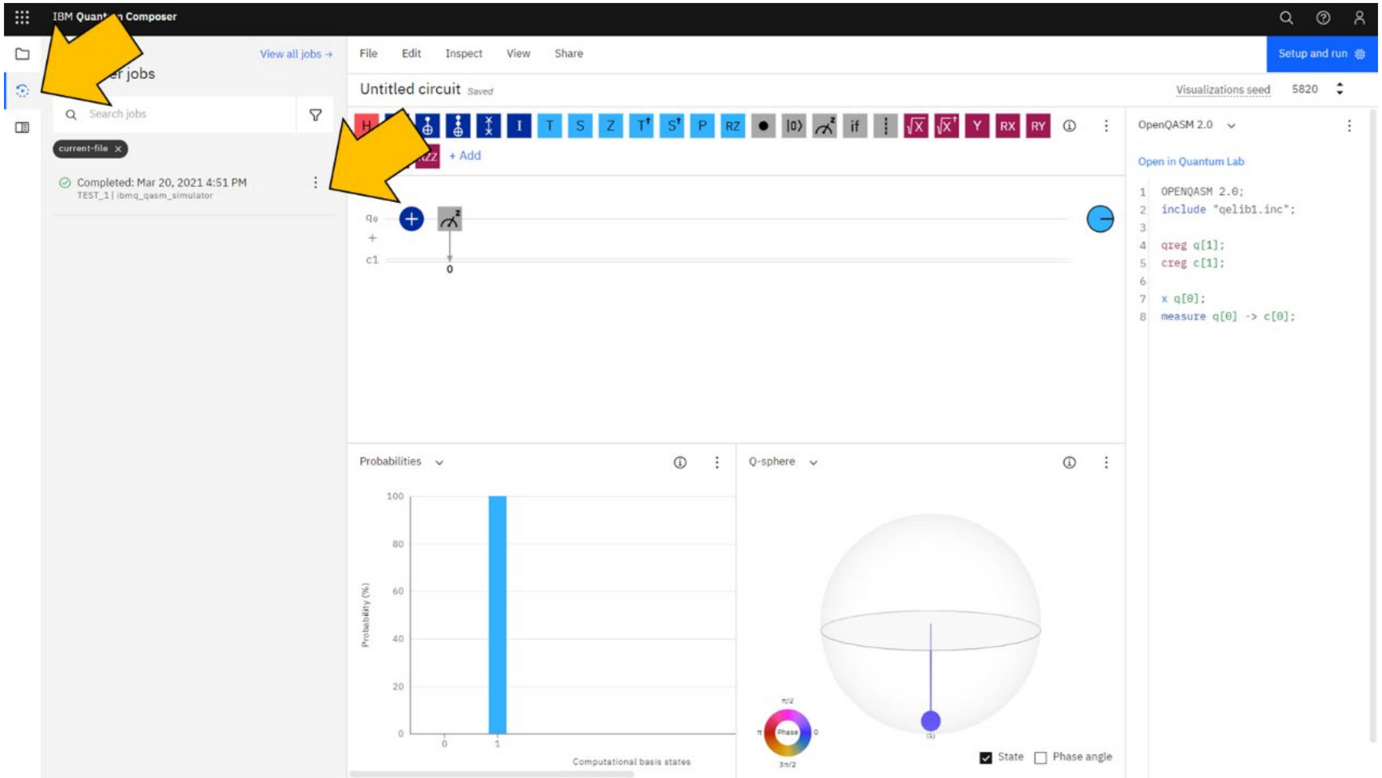
Step 6 Run your Circuit

Click on the bottom right button to run the ibm-qasm-simulator as below:



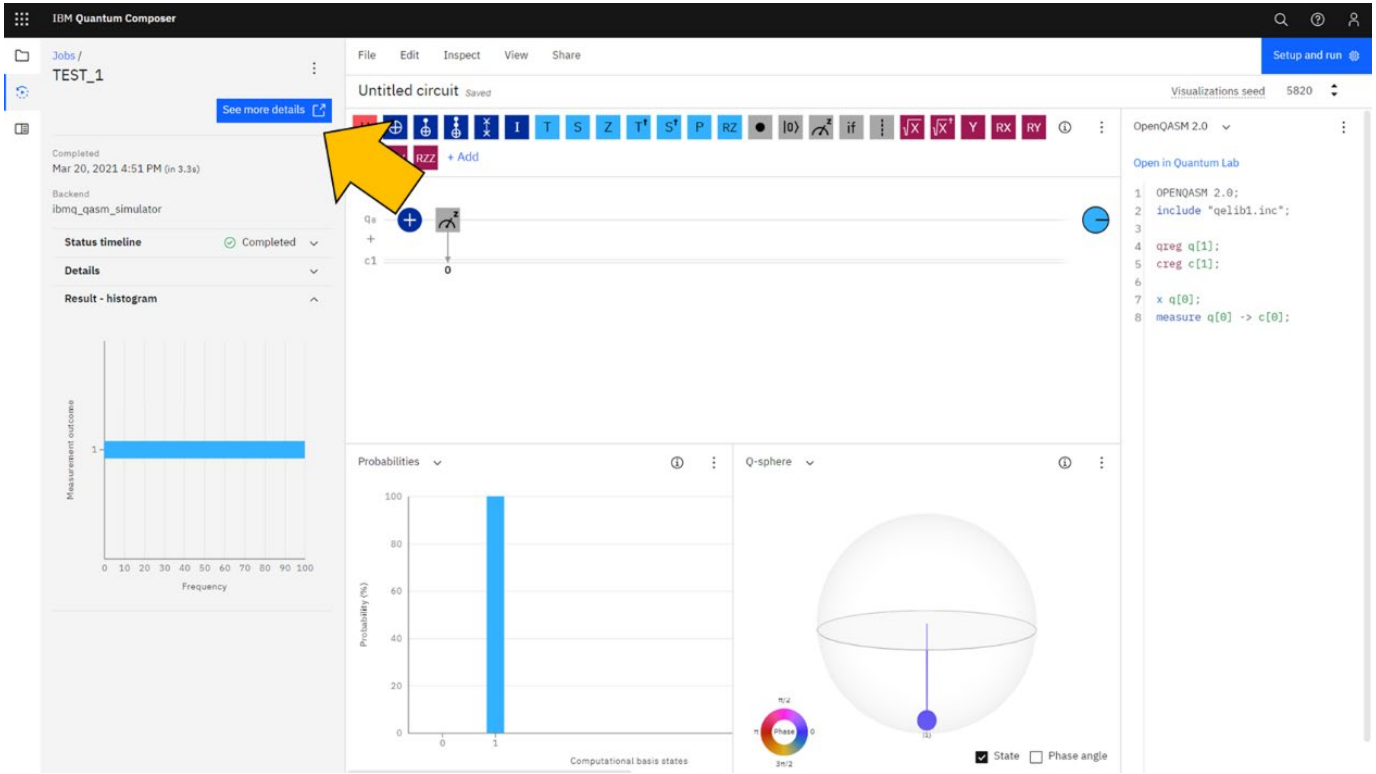
Step 7 Review the progress of your job

After a few seconds, click on the left button **Composer Jobs** to review the status of your job. In our case we have done a very simple circuit which will be calculated very quickly. For more complex jobs, more time will be required.

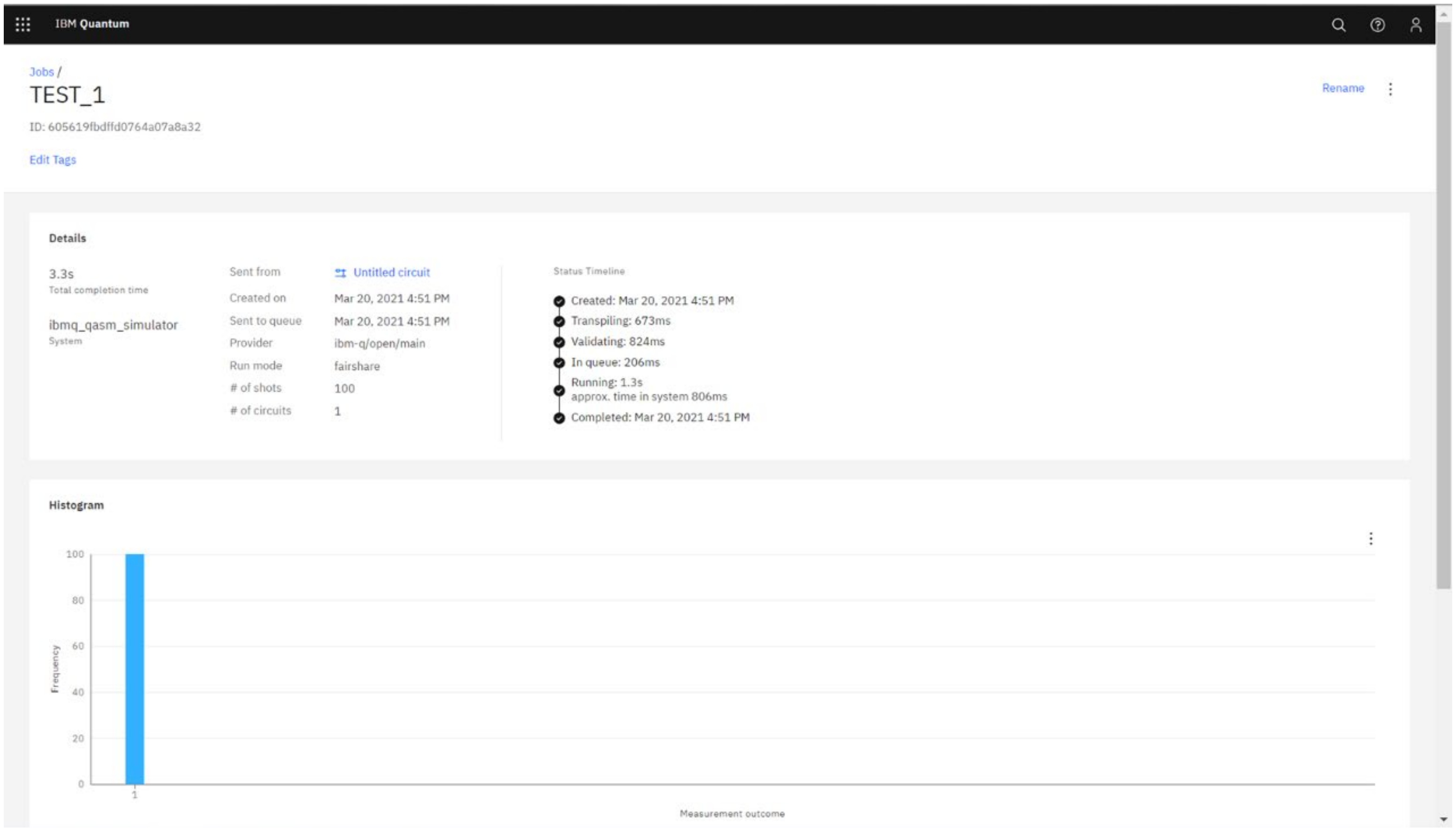


Step 8 : Get your simulation results

Clicking on the name of the job will open the results of the simulation in a new tab in your browser.



Clicking on the blue button “See More Details” will show more results in a new tab in the browser:



Congratulations! You have now created and executed your first quantum circuit. As you can see in the Histogram section, the probabilities of measuring a 1 in the qubit are 100%, just as we expected. In the next section, we will explore a complementary tool called Qiskit, which will allow us to do the same thing within Python.

Qiskit

Qiskit: an open-source framework for quantum computing

Qiskit is an open-source software development kit (SDK) for working with quantum computers. It is developed and maintained by IBM. It allows users to easily develop quantum programs as part of a Python environment. It is extremely flexible, allowing direct and detailed manipulation of circuits, pulses, and algorithms. These can then be run locally (in your own computer) or remotely (in one of the real quantum computers of IBM). Qiskit can be found in https://qiskit.org/.

Lab 2:
Your first circuit step-by-step with QISKIT

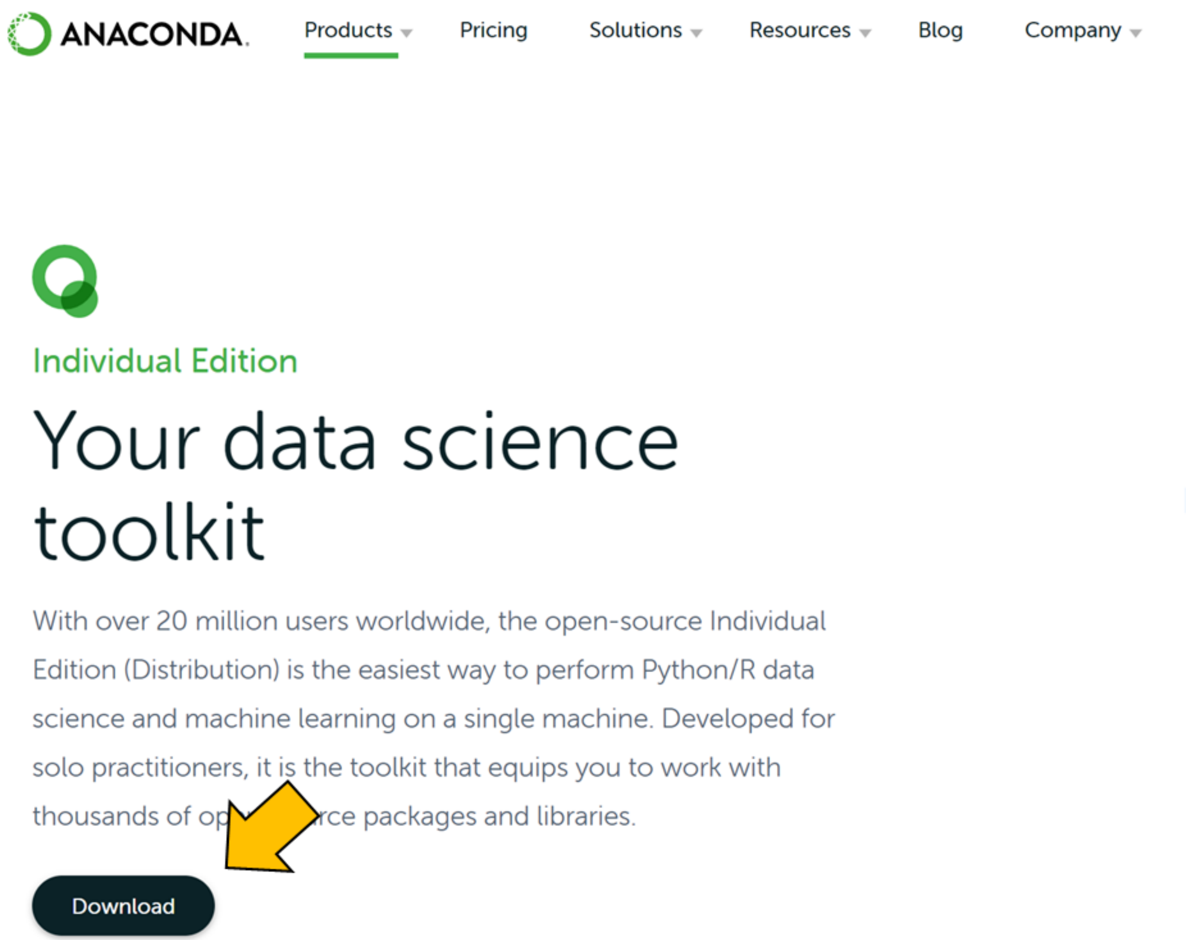
Lab 2: Your first circuit step-by-step with QISKIT

As we said, Qiskit then can be understood as module or library for Python, which allow us to run quantum programs. Note that it can be used for both: running the quantum program in a simulator or running the program in a real quantum computer. Here, we are going to repeat the exercise we did before with the IBM Quantum inLab 1, i.e. flipping a single qubit. But instead of running it online, we are going to run it in our own computer using our local Python environment.

TASK: **To flip a single qubit from $|0\rangle$ to $|1\rangle$.**

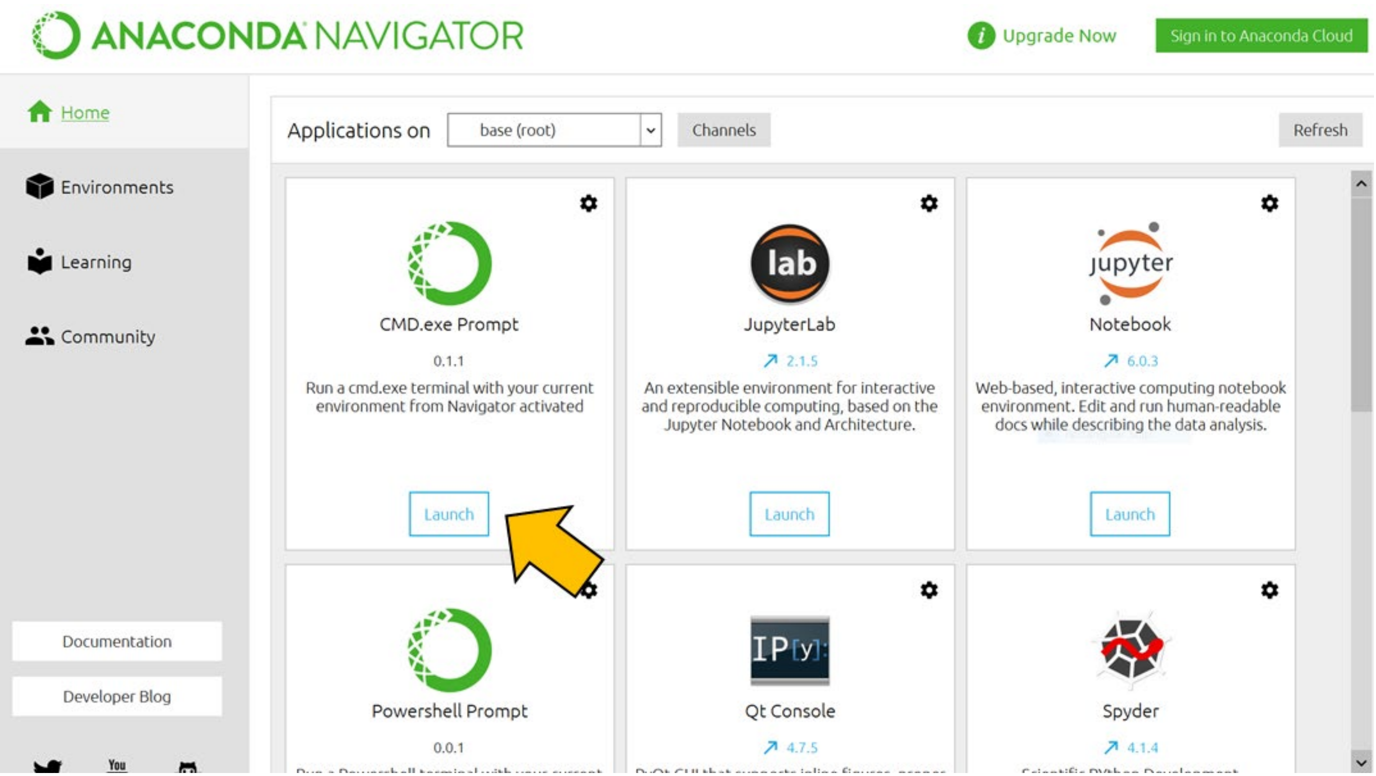
(LOCAL) SOLUTION: **Create a quantum circuit with a single qubit and a X Gate in Python using the module Qiskit.**

Step 1: Install Python in your computer



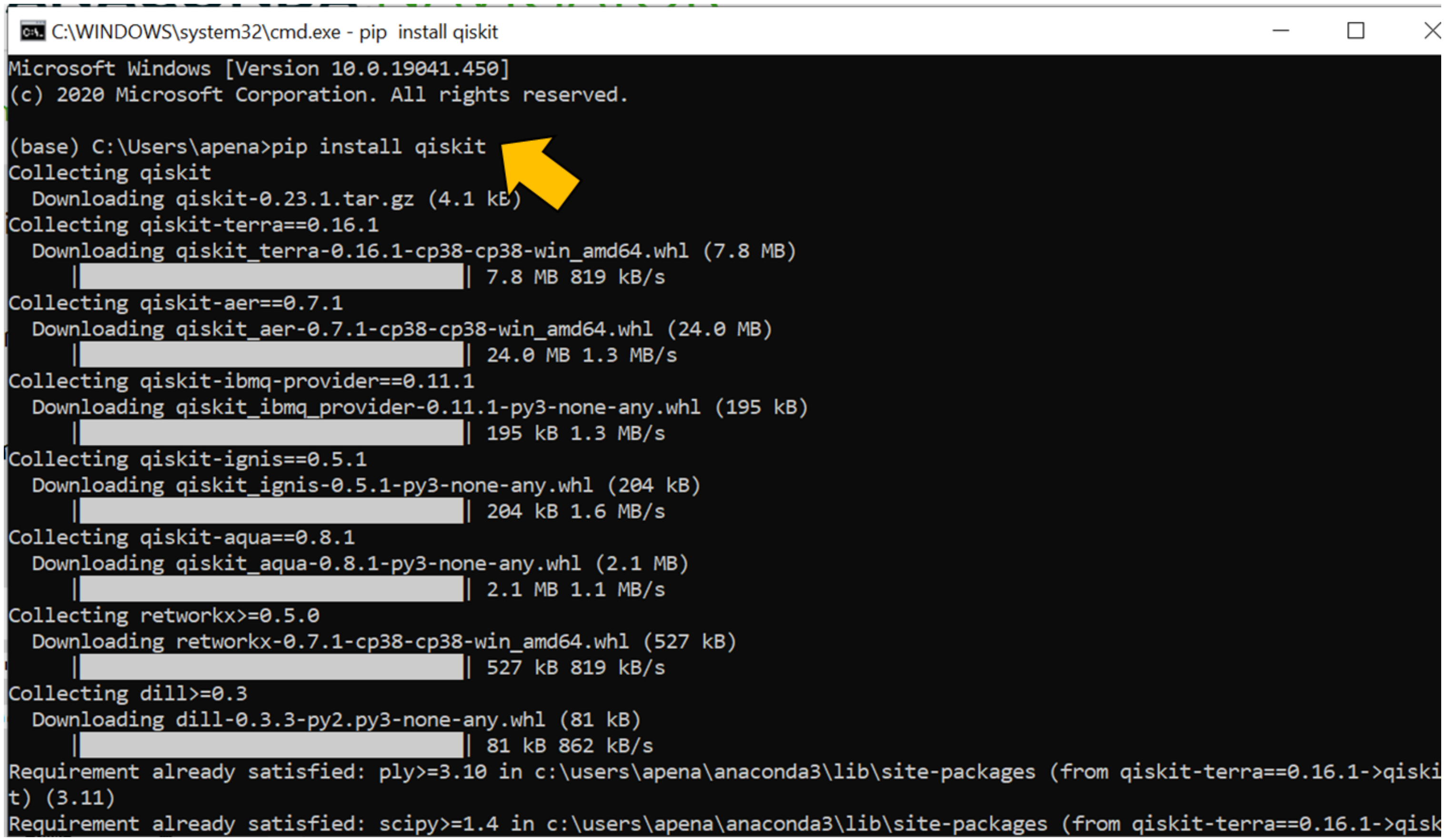
Step 2: Install the Qiskit module into your Python

Open the Anaconda Navigator and launch the CMD.exe Prompt.



Step 3: Install the Qiskit tool into your local Python

Type in the command line: `pip install qiskit`

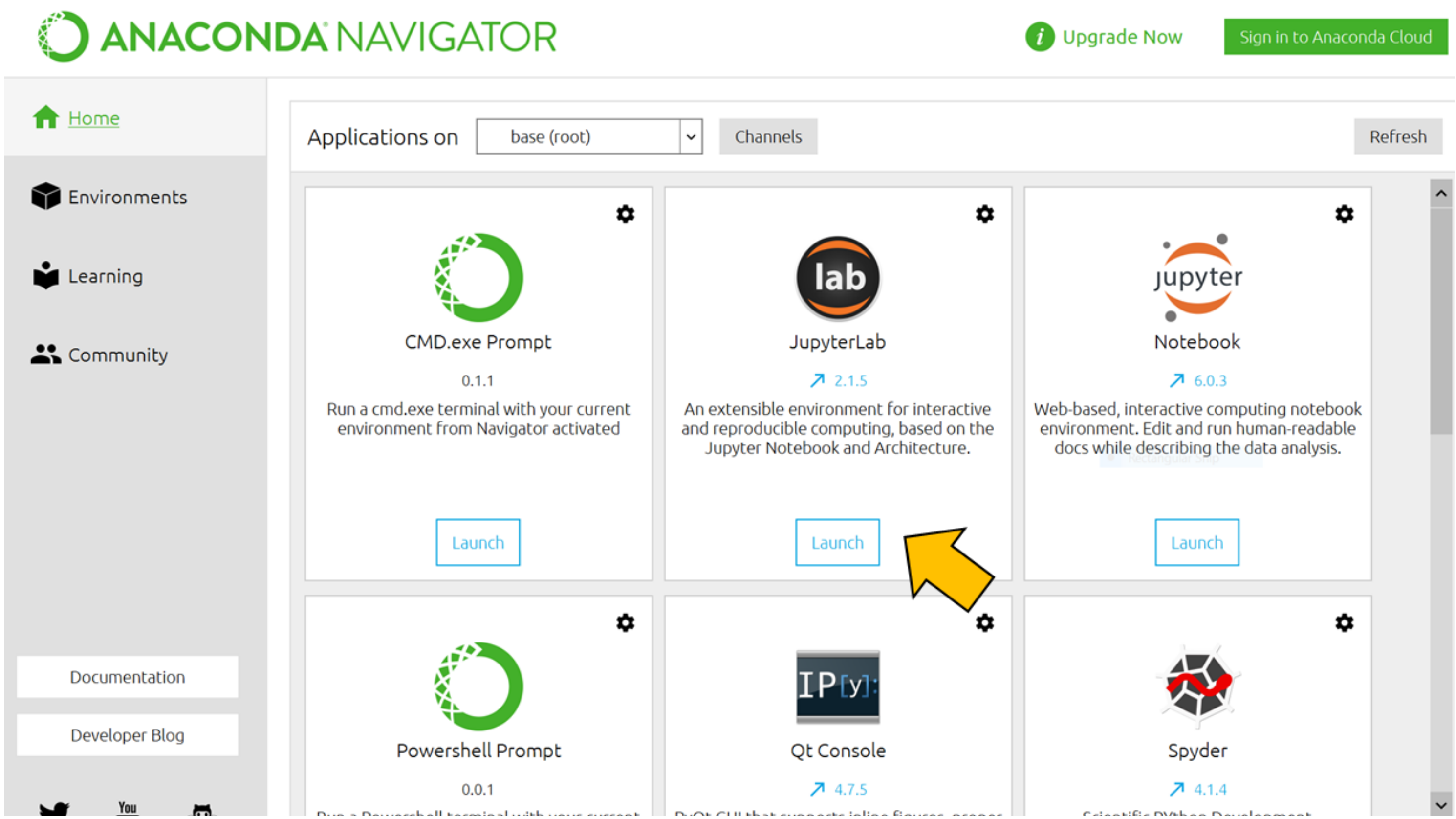


```
C:\WINDOWS\system32\cmd.exe - pip  install qiskit
Microsoft Windows [Version 10.0.19041.450]
(c) 2020 Microsoft Corporation. All rights reserved.

(base) C:\Users\apena>pip install qiskit
Collecting qiskit
  Downloading qiskit-0.23.1.tar.gz (4.1 kB)
Collecting qiskit-terra==0.16.1
  Downloading qiskit_terra-0.16.1-cp38-cp38-win_amd64.whl (7.8 MB)
     |████████████████████████████████| 7.8 MB 819 kB/s
Collecting qiskit-aer==0.7.1
  Downloading qiskit_aer-0.7.1-cp38-cp38-win_amd64.whl (24.0 MB)
     |████████████████████████████████| 24.0 MB 1.3 MB/s
Collecting qiskit-ibmq-provider==0.11.1
  Downloading qiskit_ibmq_provider-0.11.1-py3-none-any.whl (195 kB)
     |████████████████████████████████| 195 kB 1.3 MB/s
Collecting qiskit-ignis==0.5.1
  Downloading qiskit_ignis-0.5.1-py3-none-any.whl (204 kB)
     |████████████████████████████████| 204 kB 1.6 MB/s
Collecting qiskit-aqua==0.8.1
  Downloading qiskit_aqua-0.8.1-py3-none-any.whl (2.1 MB)
     |████████████████████████████████| 2.1 MB 1.1 MB/s
Collecting retworkx>=0.5.0
  Downloading retworkx-0.7.1-cp38-cp38-win_amd64.whl (527 kB)
     |████████████████████████████████| 527 kB 819 kB/s
Collecting dill>=0.3
  Downloading dill-0.3.3-py2.py3-none-any.whl (81 kB)
     |████████████████████████████████| 81 kB 862 kB/s
Requirement already satisfied: ply>=3.10 in c:\users\apena\anaconda3\lib\site-packages (from qiskit-terra==0.16.1->qiski
t) (3.11)
Requirement already satisfied: scipy>=1.4 in c:\users\apena\anaconda3\lib\site-packages (from qiskit-terra==0.16.1->qisk
```

Step 4: Launch Jupyter Lab

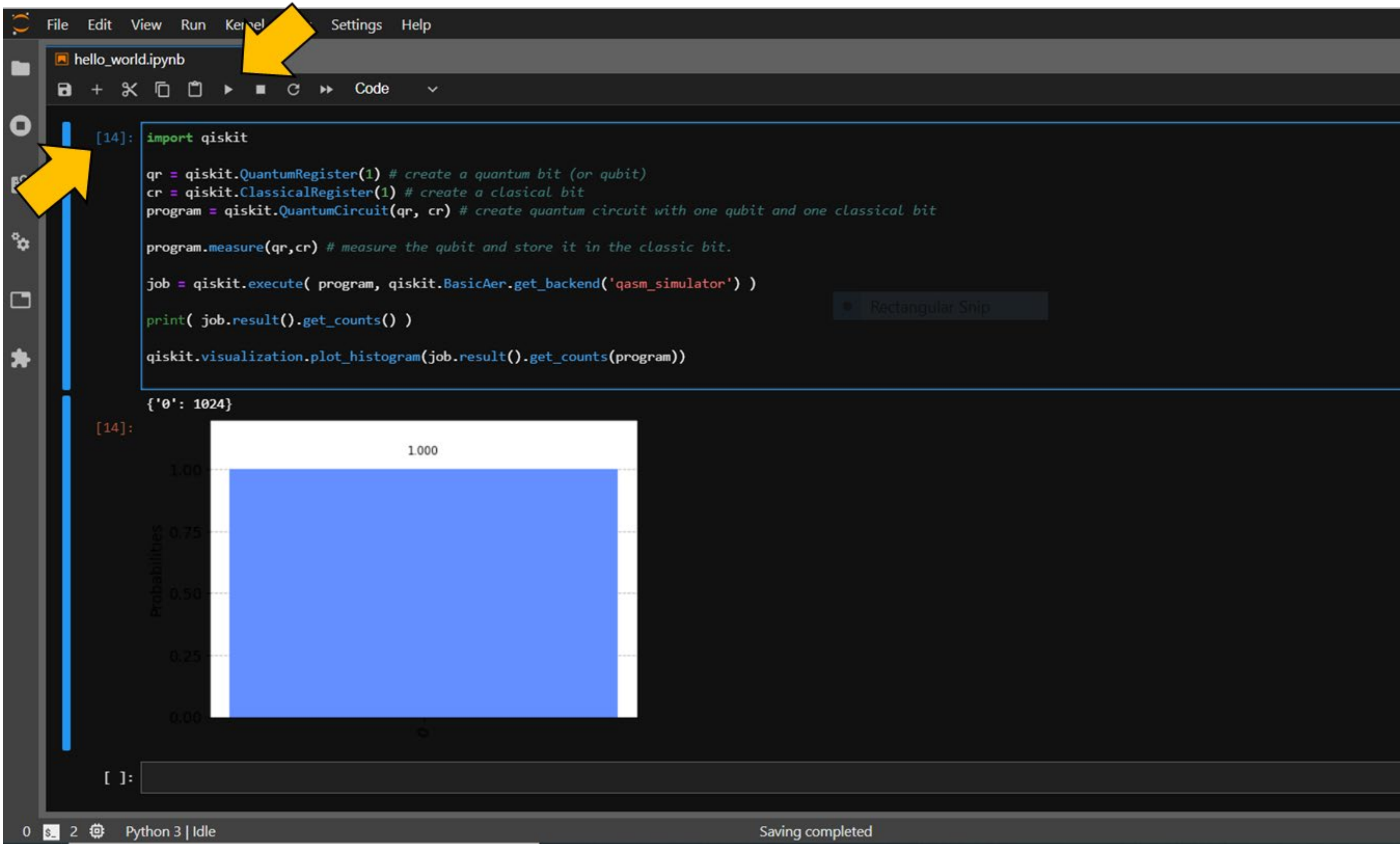
From the Anaconda Explorer launch Jupyter Lab



Step 5: Type the code below into the first line of a Jupyter Lab Notebook and Run

```
import qiskit
qr = qiskit.QuantumRegister(1) # create qubit
cr = qiskit.ClassicalRegister(1) # create clasical bit
program = qiskit.QuantumCircuit(qr, cr) # create quantum circuit with one qubit and one classical bit
program.x(qr[0]) # apply the X Gate to invert qubit
program.measure(qr,cr) # measure the qubit and store it in the classic bit

job = qiskit.execute( program, qiskit.BasicAer.get_backend('qasm_simulator') ) # run job
print(job.result().get_counts()) # print numerical results
qiskit.visualization.plot_histogram(job.result().get_counts(program)) # draw histogram
```


File Edit View Run Kernel Settings Help
hello_world.ipynb
Code
[14]:
import qiskit
qr = qiskit.QuantumRegister(1) # create a quantum bit (or qubit)
cr = qiskit.ClassicalRegister(1) # create a clasical bit
program = qiskit.QuantumCircuit(qr, cr) # create quantum circuit with one qubit and one classical bit
program.measure(qr,cr) # measure the qubit and store it in the classic bit.
job = qiskit.execute(program, qiskit.BasicAer.get_backend('qasm_simulator'))
print(job.result().get_counts())
qiskit.visualization.plot_histogram(job.result().get_counts(program))
{'0': 1024}
[14]:
1.000
[ ]:
0
2
Python 3 | Idle
Saving completed

Well done! Your result indicates that the simulation has been called 1024 times (the so-called shots in the IBM Q Experience parlance) and that in all cases the result is that the qubit has been flipped from 0 to 1.

References

Main reference book for quantum computing:

Michael A. Nielsen, Isaac L. Chuang. Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press, 2010.

On quantum computing in general:

Robert S. Sutor. Dancing with Qubits: How quantum computing works and how it can change the world. Packt Publishing, 2019

Noson S. Yanofsky, Mirco A. Mannucci. Quantum Computing for Computer Scientists. Cambridge University Press, 2008.

Wolfgang Scherer. Mathematics of Quantum Computing: An Introduction. Springer-Nature, 2019.

Rajendra K. Bera. The Amazing World of Quantum Computing. Springer Nature, 2020.

On using IBM solutions:

Robert Loredo. Learn Quantum Computing with Python and IBM Quantum Experience. Packt Publishing, 2020.

Hassi Norlén. Quantum Computing in Practice with Qiskit and IBM Quantum Experience. Packt Publishing, 2020.

Online Courses:

Quantum Computing Fundamentals, MIT xPRO

https://learn-xpro.mit.edu/quantum-computing

Understanding Quantum Computers

https://www.futurelearn.com/courses/intro-to-quantum-computing

Research Papers:

Woerner, S., Egger, D.J. Quantum risk analysis. npj Quantum Inf 5, 15 (2019).

D. Egger, R. Garcia Gutierrez, J. Cahue Mestre and S. Woerner, "Credit Risk Analysis using Quantum Computers" in IEEE Transactions on Computers, vol. , no. 01, pp. 1-1, 5555, 2019.

Jacak, J.E., Jacak, W.A., Donderowicz, W.A. et al. Quantum random number generators with entanglement for public randomness testing. Sci Rep 10, 164 (2020).

Patrick Rebentrost, Brajesh Gupt, and Thomas R. Bromley. Quantum computational finance: Monte Carlo pricing of financial derivatives. Phys. Rev. A 98, 022321, 2018.

Nikitas Stamatopoulos, Daniel J. Egger, Yue Sun, Christa Zoufal, Raban Iten, Ning Shen, and Stefan Woerner. Option Pricing using Quantum Computers. Quantum 4, 291 (2020).

Panagiotis K. Barkoutsos, Giacomo Nannicini, Anton Robert, Ivano Tavernelli, and Stefan Woerner. Improving Variational Quantum Optimization using CVaR. Quantum 4, 256 (2020).