PLANQK Docs

Menu On this page

Quickstart

A crash course on using the PLANQK Platform to run an entire quantum workflow, from development to deployment 🚀.

Installation

Create an account

If you don't have one yet, create an account

Install the PLANQK CLI

To install the PLANQK CLI, you must install Node.js 18 or higher and the npm command line interface using either a Node version manager or a Node installer .

Then install the PLANQK CLI globally using npm:

bash

```
npm install -g @planqk/planqk-cli
```

IMPORTANT

Make sure you have uninstalled the deprecated CLI:

bash

```
npm uninstall -g @anaqor/planqk
```

For details read the CLI reference.

Login to your account

- Copy your personal access token to your clipboard.

- Login to your account using your access token:

```bash
planqk login -t <your access token>
```

Create your first project

Create your first project by running the following command:

```bash
planqk init
```

You will be prompted to provide some information about your project configuration. For this quickstart, select the following configuration:

- Name: `my-project`
- Coding Template: `IonQ Qiskit Starter`
- vCPU: `1 vCPU`
- Memory: `1GB`
- GPU Support: `No GPU Support`

> TIP
>
> Access to IonQ's quantum simulator is available in basic accounts. At least a Pro account is required, if you want to use IonQ QPUs or other quantum hardware.

This will create a new directory called `my-project` containing all required files to run your quantum code on the PLANQK Platform. The starter templates implement quantum random number generation using either the IonQ Simulator or the Qiskit Aer Simulator. You can find a detailed description of the templates in this GitLab repository . It also contains a `planqk.json` file, which contains the project configuration. The file should look like this:

```json
{
  "name": "my-project",
  "descriptionFile": "README.md",
  "resources": {
    "cpu": 1,
    "memory": 1
```

```
  },
  "runtime": "PYTHON_TEMPLATE"
}
```

Test your service locally

Let's test your service locally before deploying it to the PLANQK Platform. First, switch to your project directory:

bash

```
cd my-project
```

Then, install the required dependencies. We recommend using a dedicated Conda environment. As an alternative, you can use the requirements.txt to install the dependencies with the tooling of your choice.

With conda run:

bash

```
conda env create -f environment.yml
conda activate my-project
```

Finally, run your service locally:

bash

```
python -m src
```

The output should look like this:

json

```
{
  "result": {
    "random_number": 216
  },
  "metadata": {
    "execution_time": 9.327
  }
}
```

Test your service locally with PLANQK CLI

To begin, navigate to your project directory:

```bash
cd my-project
```

Next, run the following command:

```bash
planqk serve
```

Once the server is operational, you can access http://localhost:8081/docs . This interface provides you the ability to manage service executions via the API. Further information can be found in the documentation.

Deploy your service

Deploy your service to the PLANQK Platform. Within your project directory, run:

```bash
planqk up
```

This will build your service, deploy it, and make it accessible to you via a REST API.

Execute your service

Execute your service with the example input data stored in the `input/data.json` and `input/params.json` files by running the following command:

```bash
planqk run
```

After a successful execution, the output should look like this:

```
Running Job (a7a3422b-9522-408b-96c9-32cdb497b12b)... Job succeeded.
See result at https://platform.planqk.de/jobs/a7a3422b-9522-408b-96c9-32cdb497b12b
```

For more details and options see the CLI reference.

What's next?

Get Started with our SDK

Learn how to use our SDK to interact with our supported quantum backends.

Deep Dive into Jobs

Learn everything about Jobs and how to use them.

Custom Docker Images

Run your own Docker images on the platform.

Explore our Tutorials

Check out our tutorials to get started with the platform.

Help us improve this page!

Last updated: 22.01.25, 23:16

Next page
Accessing Quantum Backends