CIUK Cluster Challenge 2025 – UCL Challenge

Welcome to the UCL challenge. For this challenge you will be accessing our cloud compute platform, Condenser.

Goal

In this challenge your team will have to create a dashboard to visualise a provided dataset with economic information about countries around the world. To do this you will need to create an application inside a container that could be deployed and provide the website with the dashboard.

What technologies you will be using

- Programming language: Python
- Dashboard framework: Streamlit
- Version control system: Git
- Collaboration platform: GitHub
- Deployment tool: Docker (and docker-compose)
- Virtual machine access: Bash
- Optional - Database: any SQL database (sqlite, postgresql, mariadb, ...)

Description of the challenge

Each team will have a virtual machine available to them on the Condenser platform. In the virtual machine we would like you to **Build and deploy a container**, the container should show a **web-based dashboard** that you'll have to build using Python and Streamlit. The dashboard will be used to **visualise the provided GapMinder data**. You are free to decide how to design the dashboard and what to visualise from the dataset, but the dashboard should provide **interactive way** to change the visualisations presented. **Digital accessibility** will be evaluated. To execute this challenge, you will be working on your team **using GitHub**, not only using it for saving your code, but also to **keep track of the features, bugs, and decisions** taken using the issue tracker provided.

At the end you'll have to create a 5 min video explaining your choice of visualisation(s) and why – what story are you telling?

Scoring

We will be scoring you on:

Scoring Item	Max points	Quality metric
Container set-up	**25**	How clear is the container recipe? How large is the container created? **Documentation**: How well is the deployment documented?
Extra: container separation between data and code	10	How are the containers orchestrated? How do the containers connect to each other?
Code development	**30**	Broken down into:
Version control	10	How has the team worked together using git (branches strategy, pull-requests, code review)?
Collaboration	5	How has GitHub issue tracker been used?
Code quality	15	**Readability**: how well the code is structured? How useful are the comments? **Structure**: how the code integrates reading and pre-process the data and its visualisation? **Documentation**: how well is the code documented?
Extra: tests	10	Unitest are included and check for correctness as well as for edge cases
Dashboard design	**25**	
Meaningful	10	Is the data shown in a meaningful manner?
Digital accessibility	15	Is the website accessible? Teams will be ranked by accessibility and the points will be distributed evenly between the first and the last.
Total	80 (+20 extra)	

Accessing Condenser

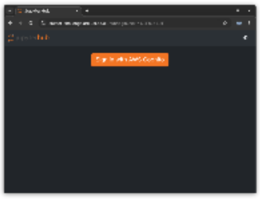
You will access condenser through a JupyterHub interface. Once there, you'll have access to a terminal and Jupyter Lab as IDE. The virtual machine has access to internet,

Please check that you have received the following:

1. You should have received an email containing your username and a temporary password. These are used to login to JupyterHub. This email will be from noreply@verificationemail.com. If you have not received this, please check your spam folder.
2. As part of the challenge materials, you should also have received an IP address and SSH key. These are for your team's shared server.

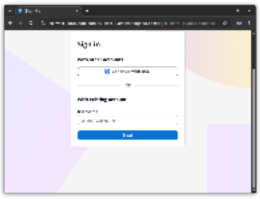
To access the platform:

1. Visit the following URL: https://cluster-challenge.arc.ucl.ac.uk in a modern web browser.
2. Choose "Sign in with AWS Cognito":

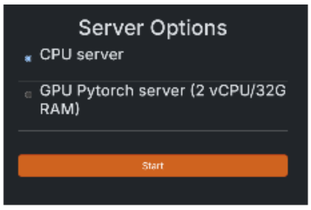


3. Under the "with existing account" option, sign in with the username/password emailed to you. You will be prompted to change your password the first time you

login:



4. Once you’ve logged in, choose “CPU Server” (GPUs are not available for this challenge) and choose “Start”:



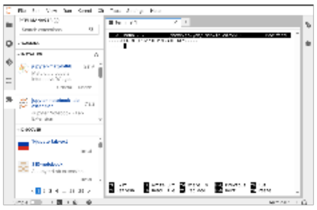
Server Options

● CPU server

○ GPU Pytorch server (2 vCPU/32G RAM)

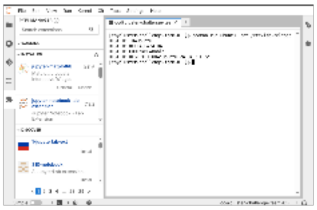
5. These will be a short delay the first time you login whilst your environment is provisioned. Once this is complete, you should have access to a JupyterHub environment.

6. Using a text editor of your choice, place the private key you will have received into the file “~/.ssh/id_ed25519”:



7. Set file permissions on the SSH key using the command: “chmod 600 ~/.ssh/id_ed25519”. Y**ou may need to re-run this command if your JupyterHub container stops/restarts.**

8. You can now SSH to your teams shared server, using the command: “ssh cc@<IP Address Provided>”. Once you have ssh’d into the server, test that docker is working by running “podman run ubuntu cat /etc/lsb-release”:



9. You will not have root access to this server – however you should note that should the server need to be rebuilt for any reason, there are no backups. For this reason, we recommend utilising tools such as GitHub for your workflow.

Developing your code

Your team Capitan, or another nominated team member, needs to create the team repository using this GitHub link:

`https://classroom.github.com/a/qtTaGRfE`

Then share the name of the team with the rest of the team. The rest of the team will be able to join selecting the correct team. **Note:** you won’t be able to change this later.

In that repository you should save all the components of your code (docker files, streamlit code, etc …) and it should have a `README.md` file that explains how is that structured and what needs to be done to deploy your application.

You will be able to push and pull from the virtual machine, but each user will need to configure it. We strongly recommend using the GitHub CLI to create the ssh keys to interact with GitHub as well as for other usages it provides. On the machine itself you may not have a graphical user interface to edit the files, so editing locally on your machine on your preferred editor (e.g., VS Code), on GitHub or through codespaces may make your experience better.

Deploying your dashboard

By default, the virtual machine can't be seen from the outside world. You can deploy the website that your application locally, but it must serve it on port 8080.

We will use a combination of your video and your files to assess your dashboard.

About the data set

You have been provided with a subset of Gapminder data. It contains the population, life expectancy and GDP per capita (US dollars) in 5 yearly increments from 1952 to 2007. The data is from 142 countries across Africa, the Americas, Asia, Europe and Oceania.

About GDP (gross domestic product) per capita

Useful resources

- Building a dashboard in Python using Streamlit
- Data camp Docker tutorial
- Gapminder data documentation
- GitHub courses (in particular the Introduction to GitHub, Communicate using Markdown, Review pull requests)
- FreeCodeCamp introduction to Git and GitHub