Projects 02

Tópicos de Informática para Automação

Mário Antunes

24 de Novembro de 2025

Projects

Form groups of two or three students (exceptionally, projects can be done individually) and select **one** of the following projects. All projects will be hosted on **GitHub**, using GitHub Classroom. Check here for details.

The repository must contain all relevant scripts, configuration files, and a `README.md` with instructions on how to deploy the project. It should also contain a project report in PDF format.

This is a three-week project (deadline 22/12/2025). You have until the end of this week to notify your professor (via e-mail) of your group members and chosen topic (the list of topics can be found here).

Do not forget to contact your professor with any questions. Further instructions may be added.

Topics

1. The "Markdown to PDF" Factory

- **Description:** Create a Dockerized service that converts Markdown files into professional PDFs. You must create a `Dockerfile` that installs **Pandoc** and a minimal LaTeX distribution (e.g., `texlive-xetex`). The container should run a **Bash script** that monitors a specific input **volume**. When a `.md` file is detected in that volume, the script must automatically convert it to `.pdf` and place the result in an output folder.
- **Core Topics:** Bash Scripting (loops, file monitoring), Docker Volumes, Document Compilation (Markdown & LaTeX).

2. Network Latency Visualizer

- **Description:** Develop a tool to analyze network stability using a containerized pipeline.
  1. Create a **Bash script** that "pings" a target (e.g., `google.com` or `ua.pt`) periodically and logs the timestamp and latency (ms) to a **CSV** file.
  2. Create a **Python script** that reads this CSV using **Pandas** or **Polars** and generates a line chart showing latency over time using **Matplotlib** or **Seaborn**.
  3. The entire process must run inside a container, saving the final plot to a volume.
- **Core Topics:** Networking (ICMP/Ping), Data Manipulation (CSV), Data Visualization, Docker.

3. Geo-Data Dashboard (Traffic or Weather)

- **Description:** Build a web dashboard that visualizes geographical data. You must create a **Python** script that uses an API to get Weather or traffic data or (using **Pandas** or **Polars**) that processes a dataset (e.g., a CSV of weather stations or traffic incidents with Lat/Lon coordinates) and exports it to JSON. Then, deploy a **Web Server** container (Nginx or Apache) hosting an HTML page. This page must use the **Leaflet** JavaScript library to read that JSON data and display markers on an interactive map.
- **Core Topics:** Web Programming (HTML/JS/Leaflet), Data Formatting (CSV to JSON), Docker, Web Servers.

4. The Universal CSV Plotter

- **Description:** Create a generic data visualization tool encapsulated in a Docker container. The container should run a **Python** script that accepts a CSV file (via webpage) and generates a plot based on arguments or a simple config file. For example, the script should be able to read `data.csv`, and using **Matplotlib** or **Seaborn**, generate a bar chart or scatter plot for two specific columns (e.g., "Date" and "Value"). The output image must be sent back into the page and allows download.
- **Core Topics:** Python Data Analysis (Pandas/Polars), Visualization libraries, CLI arguments, Docker Volumes.

5. Interactive Portfolio

- **Description:** Build and deploy a personal portfolio website using a lightweight web server container (like Nginx). Unlike Project 1, you must write the code yourself. The site must include:
  1. **HTML/CSS:** A responsive layout (Flexbox/Grid) for your bio and skills.
  2. **JavaScript:** An interactive component, such as a "Contact Me" form that validates input (e.g., ensures email format is correct) before showing a success alert, or a theme toggler (Dark/Light mode).
- **Core Topics:** Web Programming (HTML5, CSS3, JavaScript), Web Servers, Docker.

6. Server Resource Report

- **Description:** Simulate a system administrator task. Create a script that generates a "server log" CSV file (Columns: Timestamp, CPU_Usage, RAM_Usage). Then, use **Pandas** or **Polars** to analyze this log and generate a warning report: identify rows where usage exceeded 90%. Finally, use **Seaborn** or **Matplotlib** to generate a graph of the resource usage trends and save it to disk. The project report should be compiled from Markdown, embedding this generated graph.
- **Core Topics:** Data Analysis, System Concepts, Markdown integration, Python.

Github Classroom Access

Here are detailed instructions to access GitHub Classroom. Most students can skip several step, given that these were completed in project 01.

1. Join the Assignment and Form Your Team

1. **Access the link:** Go to here
2. **Find your name:** Select your name from the student list. > **Can't find your name?** All names registered on PACO were added. If yours is missing, please contact **Prof. Mário Antunes**.
3. **Create a team (ONE member only):** Only **one** person from your group should create a team. Follow this exact naming structure (the nmec should be sorted): `[nmec1]_[nmec2]_[nmec3]_project02`
   - *(Example:* `132745_133052_project02`*)*
4. **Join the team (All other members):** The remaining project members must find and join the team created in the previous step.

---

2. Access the Organization and Repository

1. **Accept the email invite:** After joining a team, all members will receive an email invitation to join the `detiuaveiro` GitHub organization.
2. **You must accept this invitation** before you can continue.
3. **Refresh the page:** Go back to the GitHub Classroom page and refresh it.
4. **Verify access:** You should now see and have access to your team's working repository.

---

3. Configure an SSH Key for Access

This will allow you to clone and push to the repository from your command line without entering your password every time.

1. **Check for an existing SSH key:** Open your terminal and run this command:

   ```
   cat ~/.ssh/id_ed25519.pub
   ```

2. **Generate a key (if needed):**
   - If you see a key (starting with `ssh-ed25519 ...`), copy the entire line and skip to step 3.
   - If you see an error like "No such file or directory," run the following command to create a new key:

     ```
     ssh-keygen -q -t ed25519 -N ''
     ```

   - After it's generated, run `cat ~/.ssh/id_ed25519.pub` again to view your new key and copy it.

3. **Add the key to your GitHub account:**
   - Go to your GitHub **Settings**.
   - On the left menu, click **SSH and GPG keys**.
   - Click the **New SSH key** button.
   - Give it a **Title** (e.g., "My UA Laptop").
   - Paste the key you copied into the **Key** field.
   - Make sure the "Key type" is set to **Authentication Key**.
   - Click **Add SSH key**.

4. **Authorize the key for SSO:**
   - After adding the key, find it in your list on the same page.
   - Click **Configure SSO**.
   - Select the **detiuaveiro** organization, fill in your login details, and grant access.