

# Projeto 02

## Tópicos de Informática para Automação

Mário Antunes

24 de Novembro de 2025

### Projetos

Formem grupos de dois ou três alunos (excecionalmente, os projetos podem ser realizados individualmente) e selecionem **um** dos seguintes projetos. Todos os projetos serão alojados no **GitHub**, utilizando o [GitHub Classroom](#). Verifiquem [aqui](#) para mais detalhes.

O repositório deve conter todos os scripts relevantes, ficheiros de configuração e um `README.md` com instruções sobre como fazer o *deploy* do projeto. Deve também conter um relatório do projeto em formato PDF.

Este é um projeto de três semanas (prazo 22/12/2025). Têm até ao final desta semana para notificar o vosso docente (via e-mail) sobre os membros do grupo e o tópico escolhido (a lista de tópicos pode ser consultada [aqui](#)).

Não se esqueçam de contactar o vosso docente caso tenham dúvidas. Poderão ser adicionadas instruções adicionais.

### Tópicos

#### 1. A Fábrica “Markdown para PDF”

- **Descrição:** Criar um serviço dockerizado que converte ficheiros Markdown em PDFs profissionais. Devem criar um `Dockerfile` que instale o **Pandoc** e uma distribuição LaTeX mínima (ex: `texlive-xetex`). O contentor deve executar um **script Bash** que monitoriza um **volume** de entrada específico. Quando um ficheiro `.md` for detetado nesse volume, o script deve convertê-lo automaticamente para `.pdf` e colocar o resultado numa pasta de saída.
- **Tópicos Principais:** Scripting em Bash (ciclos, monitorização de ficheiros), Volumes Docker, Compilação de Documentos (Markdown & LaTeX).

#### 2. Visualizador de Latência de Rede

- **Descrição:** Desenvolver uma ferramenta para analisar a estabilidade da rede utilizando um *pipeline* em contentores.
  1. Criar um **script Bash** que faça “ping” a um alvo (ex: `google.com` ou `ua.pt`) periodicamente e registe o *timestamp* e a latência (ms) num ficheiro **CSV**.
  2. Criar um **script Python** que leia este CSV usando **Pandas** ou **Polars** e gere um gráfico de linhas mostrando a latência ao longo do tempo usando **Matplotlib** ou **Seaborn**.
  3. Todo o processo deve correr dentro de um contentor, guardando o gráfico final num volume.
- **Tópicos Principais:** Redes (ICMP/Ping), Manipulação de Dados (CSV), Visualização de Dados, Docker.

#### 3. Dashboard de Geo-Dados (Tráfego ou Meteorologia)

- **Descrição:** Construir um dashboard web que visualize dados geográficos. Devem criar um script em **Python** que utilize uma API para obter dados meteorológicos ou de tráfego, ou (usando **Pandas** ou **Polars**) que processe um conjunto de dados (ex: um CSV de estações meteorológicas ou incidentes de tráfego com coordenadas Lat/Lon) e o exporte para JSON. De seguida, implementar um contentor de **Servidor Web** (Nginx ou Apache) alojando uma página HTML. Esta página deve usar a biblioteca JavaScript **Leaflet** para ler esses dados JSON e exibir marcadores num mapa interativo.

- **Tópicos Principais:** Programação Web (HTML/JS/Leaflet), Formatação de Dados (CSV para JSON), Docker, Servidores Web.

#### 4. O Plotter Universal de CSV

- **Descrição:** Criar uma ferramenta genérica de visualização de dados encapsulada num contentor Docker. O contentor deve executar um script em **Python** que aceite um ficheiro CSV (via página web) e gere um gráfico com base em argumentos ou num ficheiro de configuração simples. Por exemplo, o script deve ser capaz de ler `data.csv`, e usando **Matplotlib** ou **Seaborn**, gerar um gráfico de barras ou de dispersão para duas colunas específicas (ex: "Data" e "Valor"). A imagem de saída deve ser enviada de volta para a página e permitir o download.
- **Tópicos Principais:** Análise de Dados em Python (Pandas/Polars), Bibliotecas de visualização, Argumentos de CLI, Volumes Docker.

#### 5. Portfólio Interativo

- **Descrição:** Construir e lançar um website de portfólio pessoal usando um contentor de servidor web leve (como Nginx). Ao contrário do Projeto 1, devem escrever o código vocês mesmos. O site deve incluir:
  1. **HTML/CSS:** Um layout responsivo (Flexbox/Grid) para a vossa biografia e competências.
  2. **JavaScript:** Um componente interativo, tal como um formulário "Contacte-me" que valide a entrada (ex: garantir que o formato do email está correto) antes de mostrar um alerta de sucesso, ou um alternador de tema (Modo Escuro/Claro).
- **Tópicos Principais:** Programação Web (HTML5, CSS3, JavaScript), Servidores Web, Docker.

#### 6. Relatório de Recursos do Servidor

- **Descrição:** Simular uma tarefa de administrador de sistemas. Criar um script que gera um ficheiro CSV de "log de servidor" (Colunas: Timestamp, Uso\_CPU, Uso\_RAM). De seguida, usar **Pandas** ou **Polars** para analisar este log e gerar um relatório de aviso: identificar linhas onde a utilização excedeu 90%. Finalmente, usar **Seaborn** ou **Matplotlib** para gerar um gráfico das tendências de utilização de recursos e guardá-lo em disco. O relatório do projeto deve ser compilado a partir de Markdown, incorporando este gráfico gerado.
- **Tópicos Principais:** Análise de Dados, Conceitos de Sistema, Integração com Markdown, Python.

### Acesso ao Github Classroom

Aqui estão instruções detalhadas para aceder ao GitHub Classroom. A maioria dos alunos pode saltar vários passos, dado que estes foram concluídos no projeto 01.

#### 1. Aceder ao Trabalho e Formar a Equipa

1. **Aceder ao link:** Vão [aqui](#)
2. **Encontrar o nome:** Seleccionem o vosso nome da lista de estudantes. > **Não encontram o vosso nome?** Todos os nomes registados no PACO foram adicionados. Se o vosso estiver em falta, por favor contactem o [Prof. Mário Antunes](#).
3. **Criar uma equipa (APENAS um membro):** Apenas **uma** pessoa do grupo deve criar uma equipa. Sigam esta estrutura exata de nomeação (os nmec devem estar ordenados):  
`[nmec1]_[nmec2]_[nmec3]_project02`  
 • (Exemplo: 132745\_133052\_project02)
4. **Juntar-se à equipa (Todos os outros membros):** Os restantes membros do projeto devem encontrar e juntar-se à equipa criada no passo anterior.

#### 2. Aceder à Organização e Repositório

1. **Aceitar o convite por e-mail:** Após se juntarem a uma equipa, todos os membros receberão um convite por e-mail para se juntarem à organização `deti-uaveiro` no GitHub.
2. **Devem aceitar este convite** antes de poderem continuar.

3. **Atualizar a página:** Voltem à página do GitHub Classroom e atualizem-na (refresh).
  4. **Verificar acesso:** Devem agora ver e ter acesso ao repositório de trabalho da vossa equipa.
- 

### 3. Configurar uma Chave SSH para Acesso

Isto permitir-vos-á clonar e fazer *push* para o repositório a partir da vossa linha de comandos sem introduzir a palavra-passe sempre que o fizerem.

1. **Verificar se existe uma chave SSH:** Abram o vosso terminal e executem este comando:

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

2. **Gerar uma chave (se necessário):**

- Se virem uma chave (a começar por `ssh-ed25519 ...`), copiem a linha inteira e saltem para o passo 3.
- Se virem um erro como “No such file or directory”, executem o seguinte comando para criar uma nova chave:

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

- Após ser gerada, executem `cat ~/.ssh/id_ed25519.pub` novamente para ver a vossa nova chave e copiem-na.

3. **Adicionar a chave à vossa conta GitHub:**

- Vão às **Settings** (Definições) do vosso GitHub.
- No menu à esquerda, cliquem em **SSH and GPG keys**.
- Cliquem no botão **New SSH key**.
- Dêem-lhe um **Título** (ex: “O meu Portátil UA”).
- Colem a chave que copiaram no campo **Key**.
- Certifiquem-se que o “Key type” está definido como **Authentication Key**.
- Cliquem em **Add SSH key**.

4. **Autorizar a chave para SSO:**

- Após adicionar a chave, encontrem-na na vossa lista na mesma página.
- Cliquem em **Configure SSO**.
- Seleccionem a organização **detiuaveiro**, preencham os vossos dados de login, e concedam acesso.