FACULDADE ENGENHARIA
Departamento de Informática

Sistema Portátil de Elaboração e Correção de Testes e Gestão da Avaliação e Assiduidade
Proposta de Projeto

Orientador: Pedro R. M. Inácio(inacio@di.ubi.pt)

Objetivos

Os principais objetivos deste projeto são desenhar e desenvolver um sistema portátil, baseado num mini-computador RaspberryPi, com uma plataforma de gestão da avaliação sobretudo direcionada para unidades curriculares de informática que congregue as seguintes funcionalidades principais, por ordem de interesse:

1. Configuração simples de uma turma através de carregamento de um ficheiro com informação dos alunos ou através de uma interface web;
2. Configuração simples de critérios de avaliação;
3. *Dashboard* (web), para o docente, geral sobre todos(as) os(as) estudantes vs notas e assiduidade;
4. *Dashboard* (web), para o docente, específico para cada estudante;
5. Funcionalidades de elaboração de testes de escolha múltipla para o docente;
6. Funcionalidades de elaboração e submissão de testes de escolha múltipla para o estudante via web;
7. Funcionalidades associadas à correção e analítica de testes (a apresentar na *Dashboard* do docente), bem como de notificação ou apresentação da nota final ao(à) estudante;
8. Funcionalidades de elaboração de testes ou exercícios práticos para ambiente de linha de comandos para o docente;
9. Funcionalidades de elaboração e submissão de testes ou exercícios práticos para ambiente de linha de comandos via SSH;
10. Funcionalidades associadas à correção e analítica de testes ou exercícios práticos efetuados em linha de comandos (a apresentar na *Dashboard* do docente), bem como de notificação ou apresentação da nota final ao(à) estudante;
11. *Dashboard* (web) direcionado para o estudante e que permita que este veja as suas notas e assiduidade via *Web*;
12. Funcionalidades associadas ao registo e consulta de assiduidade de cada estudante.

A ideia por detrás desta proposta é a de que o sistema a ser desenvolvido e preparado acompanhe o docente para as aulas ou o escritório, fornecendo as funcionalidades acima sempre que é ligado e acedido com as credenciais corretas. Depois de ligado, o sistema deve automaticamente criar uma rede de área local (*hotspot*) a que o docente pode ligar-se via *wifi*, estando a *Dashboard* para o docente também imediatamente acessível no endereço do *Gateway*. O acesso ao *Dashboard* deve ser controlado por verificação de credenciais. Uma vez conseguido o acesso, o docente deve aceder às funcionalidades acima referidas e que lhe dizem respeito, para além de funcionalidades de gestão dos seus dados (nomeadamente credenciais de acesso).

Os(as) estudantes também se podem ligar à rede *wifi* e aceder ao seu *Dashboard* (se desenvolvido no projeto). A página inicial (de acesso aos *Dashboards*) deve portanto permitir evoluir para um dos dois caminhos possíveis (Docente ou Estudante). Depois de autenticado, o(a) estudante deve conseguir ver as suas notas ou assiduidade, ou então fazer um teste de escolha múltipla, se assim estiver definido. Se houver tempo para desenhar e desenvolver a funcionalidade, deve ser também possível registar a sua assiduidade ou atualizar os seus dados na plataforma, incluindo credenciais de acesso. Adicionalmente, em casos em que esteja especificado um teste ou exercícios práticos em linha de comandos, deve ser possível ao estudante ligar-se ao sistema via SSH (acedendo via `guest@gateway`) para elaboração dos exercícios via linha de comandos.

O sistema deve ser concretizado por um conjunto de *scripts*, programas e por uma (ou mais) base de dados. Para os testes de escolha múltipla, as tecnologias deverão ser sobretudo orientadas para a web (PHP+CSS+HTML5+JavaScript), enquanto que para os testes ou exercícios em linha de comandos, deve optar-se por *bash scripting* e ferramentas disponíveis em linha de comandos. A tecnologia de bases de dados a usar em todo o sistema será sempre SQLite3.

Para os testes de escolha múltipla pode ser considerado que serão entregues enunciados impressos aos(às) estudantes, bem como uma folha de compromisso sobre como o estudante quer submeter o teste (na plataforma ou escrito). Para os testes ou exercícios na linha de comandos, será requisito que o sistema crie contas de utilizador para cada estudante que fizer *login* no sistema, sendo que os exercícios devem ser gerados individualmente a partir de lógica colocada pelo docente. Ao entrar na conta, o estudante pode correr os *scripts* numerados que contém para ver o enunciado do exercício e perguntas, bem como para responder e obter *feedback*. O jogo em [3] pode dar uma melhor ideia do que é pretendido. Alguns exercícios podem ter precedência em relação a outros e o sistema deve garantir estes detalhes. Note-se que já existem plataformas separadas desenvolvidas para cada um dos subsistemas referidos neste parágrafo, sendo necessária a sua integração.

Todas as *interfaces* devem ser o mais simples possível, e.g., adotando o `w3css`[1] e evitando o uso de imagens. A documentação deve ser rica e uniforme e devem ser usados repositórios de código para manutenção e gestão de versões (e.g., github). Deve apontar-se para fluxos de utilização intuitivos e inclusão de mensagens simples sobre como funcionam as várias partes do sistema.

O sistema deve ser auto-arrumado. Por exemplo para sistema de testes e exercícios em linha de comandos, isto significa que, após um(a) estudante se autentica, devem ser-lhe criados os recursos necessários à elaboração dos exercícios e, ao elaborar os exercícios, devem ser guardadas as classificações na base de dados. No

[1] https://www.w3schools.com/w3css/default.asp

final, deve ser facultado um resumo dos resultados obtidos, e dar indicações do que é necessário estudar melhor (*feedback* parcialmente personalizado). Ainda no final, devem ser limpos os ficheiros e diretorias antes criadas. Para o utilizador docente e para além da forma de criar exercícios novos e de os atribuir, o sistema deve ainda permitir obter resumos integrados das classificações obtidas, bem como estatísticas acerca da dificuldade dos exercícios. Outros aspetos podem vir a ser discutidos ao longo do semestre.

A plataforma deve ter em consideração aspetos de segurança (e.g., HTTPS na parte web, armazenamento seguro de credenciais e anti-fraude) e de desempenho. O conjunto de testes a desenvolver no final do semestre deve contemplar alguns testes de *stress*.

Dada a sua natureza, este projeto requer conhecimentos sólidos em Segurança Informática e Redes de Computador, bem como em Sistemas Operativos e Programação. O(a) aluno(a) terá uma oportunidade de solidificar o seu conhecimento nas várias áreas abrangidas por este projeto, e também trabalhar em ambiente laboratorial com elementos do grupo *Multimedia Signal Processing*-Covilhã, do Instituto de Telecomunicações.

Tarefas a Realizar e Cronologia

T1 Contextualização com as tecnologias envolvidas e trabalho desenvolvido, nomeadamente com o Raspberry Pi. Preparação do ambiente de desenvolvimento e infraestrutura de suporte (0,5 meses);

T2 Elaboração detalhada dos requisitos do sistema e plataforma (0,5 meses);

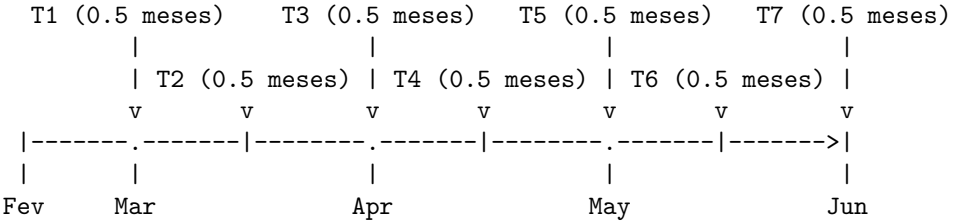
T3 Integração e melhoria de sistemas de elaboração de testes e exercícios práticos no sistema em desenvolvimento (0,5 meses);

T4 Implementação da primeira versão do *Dashboard* para Docentes e Estudantes (0,5 meses);

T5 Elaboração de conteúdos para testes dos sistema e plataforma (0,5 meses);

T6 Desenho e integração da funcionalidade de registo de assiduidade. Testes e aprimoramento da plataforma (0,5 meses);

T7 Escrita do relatório de projeto (0,5 meses).



Requisitos Técnicos / Académicos

Ter boas classificações e conhecimentos em segurança informática, redes de computadores, programação, sistemas operativos e bases de dados.

Elementos de Avaliação a Entregar

Para além do relatório, o(a) aluno(a) deverá entregar todos os *scripts* e código fonte desenvolvido, bem como toda a documentação associada.

Resultados Esperados

- Vários *scripts* e programas que cristalizem o sistema pretendido;
- O sistema portátil a funcionar com base num Raspberry Pi (modelo 2 ou 3) (uma imagem do sistema operativo pronta a instalar no Raspberry Pi e com o sistema desenvolvido a funcionar deve ser fornecida);
- O desenho da base de dados de suporte ao sistema;
- Um conjunto de aulas laboratoriais e testes traduzidos e configurados no sistema;
- 1 relatório de projeto.

Referências Bibliográficas

[1] C. Collberg and S. Kobourov, *Self-plagiarism in Computer Science*, Communications of the ACM, 48(4): 88 - 94, 2005.

[2] Kali Linux – Raspberry Pi2, http://docs.kali.org/kali-on-arm/kali-linux-raspberry-pi2, 2017. Online. Last Access: 22 January 2017.

[3] Pedro R. M. Inácio, Labirinto-de-Codigo. Available in https://github.com/in4cio/Labirinto-de-Codigo