EAPLI

Projeto PL

Aplicação para Registo de despesas pessoais

Pretende-se um sistema de registo pessoal de despesas onde o utilizador possa registar os seus gastos diários (devidamente tipificados, ex., refeições, vestuário) bem como estabelecer objetivos de poupança

Casos de uso

1. *Definição de tipo de despesas* – o utilizador deve ter possibilidade de definir tipos de despesas, ex., refeições, vestuário, transportes, lazer
2. *Definição de meios de pagamento* – o utilizador deve poder criar meios de pagamento dos vários tipos, ex., cartão visa XPTO nr. 12345678, cheques da conta 1234 do banco XYZ, etc.
3. *Registar despesa* – o utilizador deve introduzir uma despesas indicando o montante em causa, o tipo de despesa, o meio de pagamento utilizado e a data da mesma. Pode ainda incluir um pequeno comentário. No caso de pagamentos com cheque deve ser indicado o número do cheque. Após esta operação, os saldos são atualizados.
    - *Tipos de pagamento* – o sistema já possui os seguintes tipos de pagamento: dinheiro, cartão débito, cartão crédito e cheque. Alguns tipos de pagamento exigem a introdução de informação adicional no registo, ex., uma despesa paga em cheque será necessário introduzir o número de cheque.
4. *Visualização de gasto desta semana* – o utilizador deve ter sempre visível o gasto da semana corrente
5. *Visualização de gasto deste mês* – o utilizador deve ter sempre visível o gasto do mês corrente
6. *Consulta de gastos mensais* – o utilizador pode consultar uma listagem de todos os gastos de um dado mês com agrupamento por tipo de despesas
7. *Consulta de gastos mensais em formato gráfico* – o utilizador pode consultar os gastos de um dado mês agrupado por tipo de despesas num gráfico de barras
8. *Definição de tipo de rendimento* – o utilizador deve ter possibilidade de definir tipos de rendimento, ex., salário, mesada, prestação de serviços.
9. *Registo de entrada de rendimento* – o utilizador introduz um rendimento indicando o montante, a data, uma descrição e tipo de rendimento. Após esta operação, os saldos são atualizados.
10. *Visualização de saldo* – o utilizador deve ter sempre visível o seu saldo (rendimentos – despesas)
11. *Inicialização de saldo* - o utilizador deve poder configurar um saldo inicial
12. *Exportação para CSV* – o utilizador deve poder exportar os seus gastos e receitas (de um dado período de tempo) para formato CSV

13. *Exportação para XML* – o utilizador deve poder exportar os seus gastos e receitas (de um dado período de tempo) para um formato XML
14. *Importação de CSV* – o utilizador deve poder importar movimentos de um ficheiro CSV podendo optar por juntar essa informação à existente ou sobrepor os movimentos existentes no sistema
15. *Importação de XML* – o utilizador deve poder importar movimentos de um ficheiro XML podendo optar por juntar essa informação à existente ou sobrepor os movimentos existentes no sistema
16. *Visualização de tendência de gasto mensal e semanal* – o utilizador deve ter sempre visível a tendência de gasto mensal e semanal face o mês anterior (à mesma data)
17. *Configurar limites de alertas* – o utilizador deve poder configurar os limites "amarelo" e "vermelho" a partir do qual o sistema emitirá alertas:
    - Limite despesas semanais
    - Limite despesas mensais
    - Desvio em relação à média das despesas de um dado tipo
    - Limite mínimo de saldo
18. *Alerta se ultrapassar determinado montante de despesa* – o utilizador deve ser alertado se o seu total de despesas (ex., semanais) ultrapassar um valor configurado anteriormente
19. *Alerta se uma despesa sair x% da média das despesas desse tipo* – o utilizador deve ser alertado sempre que introduza uma despesa que ultrapasse x% a média desse tipo de despesa
20. *Alertas se saldo da conta abaixo de determinado montante* – o utilizador deve ser alterado se o seu saldo for inferior a u valor previamente configurado.
21. *Estabelecimento de poupança* – o utilizador pode definir um valor objetivo de poupança anual para um dado fim (texto livre), ex., férias, máquina fotográfica.
22. *Colocação em poupança* – o utilizador pode afetar um montante a uma poupança em concreto. Caso essa colocação permita atingir o objetivo pretendido, o utilizador será congratulado.
23. *Tirar de poupança* – o utilizador pode tirar um montante de uma rúbrica de poupança, seja porque o vai usar para o objetivo pretendido ou porque necessita de esse dinheiro para cobrir uma despesa. Deve ser indicado a razão do movimento.
24. *Verificação de objetivos de poupança* – o utilizador pode consultar o estado de cumprimentos dos seus objetivos de poupança sendo-lhe mostrado para cada objetivo o montante pretendido, o valor atual e a percentagem de execução.
25. *Verificação de objetivos de poupança em formato gráfico* – o utilizador pode consultar num gráfico de barras "100%" o estado de cumprimentos dos seus objetivos de poupança sendo-lhe mostrado para cada objetivo o montante pretendido e o valor atual em percentagem de execução.

Modo de funcionamento

O projeto será desenvolvido por cada turma PL. Em cada aula, o docente em conjunto com os alunos planeiam o trabalho dessa sessão. Cada par de alunos ficará responsável por uma

tarefa, tendo que a analisar, desenhar, desenvolver, testar e no fim colocar no repositório comum devidamente integrado.

No repositório deverão ser guardados os artefactos de design criados, o código e os testes. Na parte final de cada aula os alunos integrarão o seu contributo no repositório do projeto e atualizarão a execução das tarefas (ex., fechar *issues* concluídos) para que a mesma possa ser utilizada na próxima sessão.

Na 1ª sessão:

1. O professor de PL cria um repositório vazio no Github por cada PL, ex., EAPLI_PL_2DD_2ED.
2. Cada aluno deve indicar ao docente a sua conta GitHub
3. O docente adiciona esses alunos como colaboradores desse repositório
4. Cada aluno (ou par de alunos) faz o *clone* do repositório

Em todas as sessões:

1. Cada par de alunos faz o *pull* das últimas alterações para o seu repositório local por forma a garantir que tem a base de código mais atual.
2. O professor discute com a equipa (a turma PL) as tarefas existentes. Para tal utilizam a ferramenta de *issues* do repositório escolhido, ex., GitHub.
3. Cada par de alunos escolhe uma trafea/issue e trata de a analisar, desenhar, desenvolver e testar. Todos os artefactos devem ser colocados no repositório.
4. Quando concluída a tarefa, fazem o commit no repositório local com uma mensagem expressiva do trabalho efetuado, indicando o número dos launos envolvidos nessa tarefa e fechando o *issue* (“fixes #999”).
5. Em seguida tentam fazer o push para o repositório partilhado.
6. Em caso de conflito,
    a. efectuam o pull,
    b. realizam o merge,
    c. compilam e testam novamente o seu código,
    d. fazem commit
    e. fazem push

Os alunos que trabalhem em pares devem alternar a conta do repositório com que fazem *commit*. Não esquecer de colocar o número dos alunos envolvidos na mensagem de *commit*.