

Trabalho para P2 – 2024-1

versão 1.0 (29/03)

Tarefa:

Evoluir a aplicação web construída no trabalho de P1, utilizando controles de qualidade, de acordo com o conteúdo abordado na disciplina e os detalhes descritos neste documento.

Datas:

- Entrega : 05/06/2024 18:00
- Apresentação : 05/06/2024 18:10

Apresentação:

1. Duração: 25 minutos;
2. Obrigatória. Trabalhos entregues sem apresentação serão desconsiderados;
3. Com arguição do professor durante a apresentação (tempo descontado da duração);
4. Deve expor o projeto arquitetural criado para a aplicação;
5. Deve expor a aplicação construída em pleno funcionamento;
6. Deve expor todos os controles de qualidade adotados;
7. Com slides, *offline*, que devem ter pelo menos a seguinte estrutura:
 - a. **Capa**, com nome da disciplina, período, nome do(s) aluno(s) e e-mail(s) institucional(is);
 - b. **Modelagem Essencial**, com o Diagrama de Classes (UML) do Modelo Essencial;
 - c. **Arquitetura**, com um Diagrama de Classes (UML) Parcial que exemplifique parte da arquitetura construída para a aplicação (ex. Repositórios, Serviços, Controladoras, Visões, etc. que foram criados para a funcionalidade principal);
 - d. **Exposição da Aplicação**, em que a aplicação é mostrada em execução, sem uso de slides;
 - e. **Controles de Qualidade**, exposição, sem uso de slides, dos controles de qualidade adotados, tal como testes automatizados e ferramentas de análise estática, em execução;
 - f. **Referências**, com referências utilizadas para a construção da aplicação ou da apresentação. Devem usar o formato da ABNT;
 - g. **Perguntas**, com o(s) e-mail(s) institucional(is) do(s) aluno(s);

Entrega:

- No repositório no GitLab criado pelo professor para o grupo.
- No branch "main".

Repositório:

1. Será avaliado **ao longo** do projeto (frequência de uso, organização, boas práticas, etc.). Logo, deve-se enviar código frequentemente;
2. **Não deve** incluir a pasta **/node_modules**, nem pastas ou arquivos que possam ser gerados a partir do processo de instalação, compilação ou build (ex. **/dist**, **/output**, **/vendor**);
3. Deve incluir o arquivo de apresentação, **"/apresentacao.pdf"**;
4. Deve incluir todos os arquivos de **código-fonte** necessários;
5. Deve incluir um arquivo **/readme.md** com:
 - h. **nome completo** do(s) aluno(s);
 - i. **comandos** necessários para pôr o projeto em funcionamento;

- j. **referências** bibliográficas ou locais de onde foram obtidas imagens, estilos CSS, ou outros recursos aplicáveis. Indicar o que foi obtido de cada local.

Construção da aplicação:

1. Não utilizar geradores de aplicação/módulos/código de **nenhuma natureza**. A aplicação e seus recursos – exceto imagens e estilos CSS – devem ser construídos manualmente;
2. Haverá arguição oral sobre qualquer conteúdo do trabalho. Conteúdos não explicados satisfatoriamente serão **descontados** da nota do trabalho, podendo até perder toda a sua pontuação;
3. Deve empregar uma arquitetura de separação de responsabilidades, tal como MVP, MVC ou MVVM, tanto no Front-end quanto no Back-end;
4. Tecnologias do Front-end:
 - a. Utilizar o gerenciador de pacotes PNPM;
 - b. Utilizar a ferramenta de *build* Vite;
 - c. Utilizar o *framework* de testes unitários/integração ViTest;
 - d. Utilizar o *framework* de testes *end-to-end* Playwright;
 - e. Implementar a aplicação em linguagem TypeScript;
 - f. Não utilizar bibliotecas ou *frameworks* JavaScript/TypeScript para construção de interface de usuário, tal como ReactJS, VueJS, etc;
 - g. Recomenda-se utilizar um *framework* CSS para acelerar a estilização da interface de usuário, tal como Material Design (preferencialmente) ou Bootstrap;
5. Tecnologias do Back-end:
 - a. Utilizar PHP 7.4 e o gerenciador de pacotes Composer;
 - b. Utilizar a biblioteca de testes unitários/integração Kahlan;
 - c. Utilizar a ferramenta de Análise Estática PHPStan;
 - d. Utilizar o banco de dados MySQL ou MariaDB;
 - e. Pode-se utilizar uma biblioteca ou *framework* para tratar rotas RESTful, tal como a phputil/router;

Quaisquer dúvidas devem ser dirimidas com o professor o mais breve possível.

Enunciado da P1:

A Acme Serviços Financeiros (ASF) realiza empréstimos a clientes, mesmo que esses estejam em situação de crédito negativado. Para compensar o risco, ela cobra juros (simples) nas parcelas de seus empréstimos.

Crie uma aplicação web que permita um Cliente pegar um empréstimo de no mínimo 500 reais e no máximo 50 mil reais. A tela de empréstimo deve permitir localizar um Cliente pré-cadastrado pelo seu CPF, informar o Valor a Ser Empréstado (R\$), escolher a Forma de Pagamento (pré-cadastrada) que o cliente deseja utilizar para pagar o empréstimo e acionar a opção "Realizar Empréstimo" para concluir a operação. Ao informar um CPF existente, o nome completo do cliente e sua idade (que deve ser computada com base em sua data de nascimento do cliente e o dia atual) devem ser exibidos na tela (ex. "Ana da Silva, 25 anos). Ao escolher a Forma de Pagamento, todas as parcelas devem ser exibidas na tela, com: Número, Vencimento (data) e Valor (R\$). Ao concluir com sucesso um empréstimo, a aplicação deve exibir uma mensagem para o usuário e redirecioná-lo para a listagem dos empréstimos.

A tela que lista os empréstimos realizados deve apresentar os dados a seguir em forma de tabela e ordenados de forma decrescente pela primeira coluna: Data e hora, Cliente

(nome), CPF, Valor Empréstimo (R\$), Forma de Pagamento, Valor Final (R\$). O Valor Final (R\$) é o valor total a ser pago pelo cliente, após os juros.

Uma Forma de Pagamento deve ter Descrição (o que é exibido para o usuário selecionar, ex. "6 meses"), Número de Meses e Juros (%). Ao aplicar uma Forma de Pagamento a um Empréstimo, as parcelas devem considerar a data atual e o número de meses para calcular os vencimentos, a cada 30 dias. Vencimentos podem cair em fins de semana, pois serão pagos no próximo dia útil. O valor dos vencimentos deve ser calculado aplicando o juros ao valor total e então distribuindo o valor pelas parcelas. Por exemplo, um empréstimo R\$ 1.000,00 com juros de 10% e pagamento em 3 meses, ficaria R\$ 1.100,00 com a 1ª parcela de R\$ 366,67; a 2ª parcela de R\$ 366,67; e a 3ª parcela de R\$ 366,66. Observe que os valores das duas primeiras parcelas estão maiores que o da terceira parcela. Isso ocorreu por conta da distribuição do valor, uma vez que R\$ 1.100,00 dividindo em 3 vezes dá parcelas de R\$ 366,66. Porém, elas somadas perfazem R\$ 1.099,98 e não R\$ 1.100,00. Logo, os R\$ 0,02 (dois centavos) restantes foram distribuídos igualmente entre as primeiras duas parcelas.

Para a entrega da aplicação, não é necessário criar funcionalidades para gestão de Clientes ou de Formas de Pagamento. Basta seus dados existirem na base de dados.

Enunciado da P2:

A aplicação deve ser utilizável por diferentes tipos de usuário, que devem se autenticar para acessá-la. Também deve ser possível sair – **fazer *logout*** – do sistema.

Apenas um usuário **gerente** deve ser capaz de acessar o **Relatório de Empréstimos por Período**. Esse relatório deve solicitar uma data de início, que por padrão deve ser o primeiro dia do mês, uma data de término, que por padrão deve ser o último dia do mês e disponibilizar uma opção "Gerar". O relatório deve apresentar um gráfico de colunas, com os dias do período em que houve empréstimos e o valor total emprestado em cada dia. Abaixo do gráfico deve ser exibido: o Total do Período (R\$), com o somatório do valor dos empréstimos no período; e a Média do Período (R\$), com o valor médio dos empréstimos no período. Também deve haver uma opção "Mostrar dados" que faz exibir uma **listagem** contendo os dados resumidos dos empréstimos no período: # (número sequencial que começa em 1), Data, e Total (R\$).

O **login** no sistema deve poder ser realizado pelo nome de usuário, ou e-mail, mais uma senha. A senha não deve ter tamanho máximo, apenas mínimo, de 6 caracteres. Ela deve ser armazenada em forma de *hash* SHA 256 com *pepper* e *salt*.

O sistema deve prover o **cadastro de clientes**, com Nome, Data de Nascimento, CPF, Telefone, E-mail, Endereço e Limite de Crédito. Esse limite de ser calculado de forma inversamente proporcional ao valor máximo de um empréstimo. Por exemplo, se o valor máximo do empréstimo for 50 mil e o cliente pegou 5 mil emprestado, ou seja, 10% do total, ele terá 90% de crédito, ou R\$ 45 mil. O valor e o percentual de crédito do cliente devem ser mostrados na tela de **empréstimo**, de forma não alterável.

Um **empréstimo** agora passa a ter como valor máximo o crédito restante do cliente.

Agora deve ser possível **pagar uma parcela** de um empréstimo. Para isso, o sistema deve possibilitar **visualizar um empréstimo** e então acionar a opção “Marcar como Paga”, que deve fazer com que a situação da respectiva parcela mude para “Paga”. Quando isso ocorrer, o sistema deve registrar a data e hora do pagamento e o usuário do sistema que marcou a parcela como paga. Esses dados devem aparecer na lista de parcelas do empréstimo visualizado. Não deve ser possível pagar uma parcela do empréstimo caso houver uma parcela mais antiga ainda em aberto.