Projeto da Segunda Unidade

Objetivo

Desenvolver, utilizando somente recursos básicos da linguagem Python, um programa para **simular o funcionamento de um aplicativo de caronas** (como o BlaBlaCar), via **terminal**, **sem interface gráfica**, **sem funções**, **sem modularização** e **sem uso de bibliotecas externas**.

Regras Gerais

- O sistema será utilizado por pelos usuários que podem oferecer ou pegar carona.
- O código deve ser desenvolvido em um único arquivo `.py`.
- Não utilizar: `def`, `import`, nem estruturas de orientação a objetos.
- Utilizar apenas as estruturas básicas da linguagem Python: `input()`, `print()`, `if`, `while`, `for`, listas, dicionários, strings, entre outras vistas em sala.
- O foco é na **lógica algorítmica**, **organização dos dados**, e **estrutura sequencial** do programa.
- O sistema deve ser capaz de cadastrar caronas, permitir reservas e exibir os dados no terminal.

Requisitos do Projeto

R0 - Cadastro de Usuário

Descrição: Permitir que uma nova pessoa se cadastre no sistema.
Entrada: nome completo, email (único), senha.
Saída: mensagem de confirmação de cadastro.
Regras:

- O email não pode estar repetido.
- Os dados devem ser armazenados para serem utilizados no login.

R1 - Login

Descrição: Usuário deve informar seu email e senha para acessar o sistema.
Entrada: email, senha.
Saída: mensagem de login bem-sucedido ou erro.
Observação:

- Apenas usuários logados podem executar os demais requisitos.
- O sistema só deve continuar funcionando para o usuário se ele estiver autenticado.

R2 - Cadastro de Carona *(requer login)*

Descrição: Usuário logado pode cadastrar uma nova carona que ele oferece como motorista.
Entrada: local de origem, destino, data da carona, horário, quantidade de vagas, valor por vaga.
Saída: Mensagem de confirmação.
Regras:

- O nome do motorista deve ser o do usuário logado.
- Cada carona pertence a um único usuário.

R3 - Listar Todas as Caronas Disponíveis *(requer login)*

Descrição: Exibir no terminal todas as caronas cadastradas com vagas disponíveis.
Saída: Nome do motorista, origem, destino, data, horário, vagas disponíveis e valor por vaga.

R4 - Buscar Carona por Origem e Destino *(requer login)*

Descrição: Usuário insere origem e destino e o sistema mostra caronas compatíveis.
Entrada: origem, destino
Saída: lista de caronas que batem com os critérios

R5 - Reservar Vaga em Carona *(requer login)*

Descrição: Usuário logado pode reservar uma vaga em uma carona existente.
Entrada: email do motorista, data da carona
Saída: Mensagem de sucesso ou erro (caso não tenha mais vagas).
Regras:

- Reduz uma vaga da carona.
- Registra o passageiro como reserva.

R6 - Cancelar Reserva *(requer login)*

Descrição: Usuário pode cancelar uma reserva feita anteriormente.
Entrada: email do motorista, data da carona
Saída: mensagem de confirmação
Regras:

- A vaga da carona volta a ser disponível.
- A reserva é excluída.

R7 - Remover Carona *(requer login)*

Descrição: Usuário logado pode excluir uma carona que ele mesmo cadastrou.
Entrada: data da carona
Saída: confirmação de remoção
Regras:

- Apenas o criador da carona pode removê-la.

R8 - Mostrar Detalhes de Carona *(requer login)*

Descrição: Exibe todos os dados de uma carona específica.
Entrada: email do motorista, data da carona
Saída: origem, destino, horário, valor, vagas restantes, passageiros.

R9 - Mostrar Carona cadastradas *(requer login)*

Descrição: Exibe todas as caronas cadastradas do usuário logado.
Saída: origem, destino, horário, valor, vagas restantes, passageiros.

R10 - Logout

Descrição: Finaliza a sessão do usuário logado.
Saída: Mensagem de confirmação
Regras:

- Após o logout, o usuário não pode realizar nenhuma ação até efetuar novo login.
-

R11 - Funcionalidade EXTRA

Descrição: Use sua criatividade em algo que seja interessante para o usuário.

Entrega

- O arquivo `.py` deve ser enviado até **MUDAR**.

- Os alunos apresentarão o código em aula e responderão perguntas sobre sua lógica.