FUNDAÇÃO ESCOLA DE COMÉRCIO ÁLVARES PENTEADO

CAMPUS LIBERDADE

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

GABRIEL CARVALHO MOTA - 24026334

GUILHERME DE LIMA SIQUEIRO - 21010649

RODRIGO LUIZ MENEZES DOS REIS - 24025708

VITORIA LETICIA MACIEL DA SILVA - 24026593

SAFERIDE - UBER

ESTRUTURA DE DADOS UTILIZADAS

São Paulo

2025

1 ESTRUTURAS DE DADOS UTILIZADAS NO PROJETO SAFERIDE

O projeto **SafeRide** foi desenvolvido com o objetivo de garantir maior segurança aos motoristas e passageiros da Uber. Para a implementação da funcionalidade, foram utilizadas diversas estruturas de dados da linguagem Java, que possibilitaram um gerenciamento eficiente das informações do sistema. Abaixo, listamos as estruturas utilizadas e justificamos suas escolhas dentro do contexto do projeto.

1. List (Interface de Lista)

Descrição: A interface List representa uma coleção ordenada de elementos, onde é possível armazenar e acessar elementos por suas posições (índices). Essa estrutura garante que os elementos permaneçam na ordem em que foram inseridos.

Uso no Projeto SafeRide: A List foi utilizada para armazenar listas de dados dinâmicos, nós utilizamos, principalmente, para armazenar os destinos mais acessados por um passageiro disponíveis em uma determinada área, em estruturas de RecyclerView, para armazenar dados do passageiro, rotas e local de origem e destino. Sua flexibilidade permite que os elementos sejam adicionados, removidos ou modificados de maneira eficiente.

- **Por que foi escolhida?**

1. Permite acesso rápido aos elementos por meio de índices.
2. É ideal para situações onde a ordem dos dados deve ser mantida.

2. ArrayList (Implementação de Lista Baseada em Arrays)

Descrição: A classe ArrayList é uma implementação da interface List que utiliza um array dinâmico para armazenar elementos. Ela cresce automaticamente conforme novos elementos são adicionados.

Uso no Projeto SafeRide: O ArrayList foi utilizado para gerenciar listas de dados em tempo real, como os trajetos sugeridos para uma corrida e armazenamento de corridas já concluídas. Como o número de trajetos pode variar significativamente, a flexibilidade do ArrayList foi essencial.

- **Por que foi escolhida?**

1. Cresce dinamicamente conforme necessário.
2. Fácil integração com algoritmos de busca e ordenação.

3. HashMap (Mapa Baseado em Tabela de Dispersão)

Descrição: A classe HashMap armazena pares de chave-valor, permitindo acesso eficiente aos valores através de suas respectivas chaves. Internamente, utiliza uma tabela de dispersão (hash table) para garantir a velocidade das operações.

Uso no Projeto SafeRide: O HashMap foi utilizado para associar informações importantes, como:

- Chave: Identificador único do passageiro (ID).
- Valor: Informações do passageiro (nome, avaliações, localizações recentes, etc.).

Além disso, ele foi usado para relacionar trajetos a preços.

- **Por que foi escolhida?**

1. Acesso muito rápido aos elementos pela chave.
2. Flexibilidade para armazenar e acessar informações relacionadas de maneira lógica.
3. Escalabilidade para lidar com grandes volumes de dados sem uma perda muito grande de desempenho.

4. Map (Interface de Mapa)

Descrição: A interface Map define um mapeamento entre chaves e valores. Diferentemente de listas ou conjuntos, os elementos de um Map são acessados por suas chaves, e não por suas posições.

Uso no Projeto SafeRide: A interface Map foi utilizada como base para a implementação de estruturas mais específicas, como o HashMap. Utilizamos ela para definir a estrutura de mapeamento de situações como, associação de IDs de passageiros com suas avaliações.

- **Por que foi escolhida?**

1. Proporciona flexibilidade na implementação de mapas específicos (ex.: HashMap).
2. Permite gerenciar associações complexas de dados com facilidade

2 Conclusão

As estruturas de dados escolhidas para nosso projeto SafeRide, foram selecionadas com base em sua eficiência, flexibilidade e compatibilidade com os requisitos do sistema. O uso de List e ArrayList possibilitou o gerenciamento de dados dinâmicos de forma ordenada e escalável. Já o HashMap e a interface Map garantiram acesso rápido e eficiente às associações de dados críticas, como informações de passageiros e trajetos.