Faculdade de Design,
Tecnologia e Comunicação
Universidade Europeia

Think Toilet

Matemática Discreta

Curso: Engenharia Informática
Semestre: 2024/2025 - Terceiro Semestre

Nycolas Souza - 20230989
Luan Ribeiro - 20230692
Lohanne Guedes - 20220085

Introdução

O projeto **Think Toilet** é uma aplicação móvel que ajuda os utilizadores a encontrar e avaliar casas de banho próximas. A aplicação exibe um mapa interativo com as casas de banho mais bem avaliadas e fornece informações adicionais, incluindo avaliações de usuários. Os utilizadores podem avaliar critérios como limpeza, acessibilidade, papel disponível e estrutura, além de deixar comentários. A aplicação também permite denunciar locais ou comentários inadequados e visualizar seu histórico de avaliações. Com integração ao Google Maps, oferece rotas para facilitar o acesso aos locais.

No desenvolvimento do projeto, utilizamos conceitos da Matemática Discreta, principalmente na base de dados relacional, onde aplicamos **teoria de conjuntos** para descrever as relações entre as tabelas, como a relação entre os usuários e casas de banho, ou usuários e comentários. Já para a **lógica proposicional** e **predicados**, utilizamos critérios de filtros e para descrever a relação e condição entre dois objetos, como por exemplo: “Se a casa de banho foi denunciada por esse usuário, então ele não verá essa casa de banho”.

Teoria dos Conjuntos

Cada tabela da nossa base de dados pode ser representada como um conjunto para análise de Matemática Discreta, assim como:

Usuários (U):

$$U = \{u_1, u_2, u_3, u_4, u_5, u_n)$$

Onde u_i representa um usuários com seus respectivos atributos, como nome, e-mail, senha e etc.

Casas de Banho (T):

$$T = \{t_1, t_2, t_3, t_4, t_5, t_n)$$

Onde t_i representa uma casa de banho com seus respectivos atributos, como nome, endereço, latitude, longitude e etc.

Comentário (C):

$$C = \{c_1, c_2, c_3, c_4, c_5, c_n)$$

Onde c_i representa um comentário com seus respectivos atributos, como texto, notas, data de postagem e etc.

E além desses conjuntos, também temos tabelas que são frutos de **operações conjuntas**, como é o caso das seguintes tabelas:

Interação (I):

$$I \subseteq U \times T$$

Ou seja, a tabela interação é um subconjunto do produto cartesiano entre os usuários e as casas de banho, nesse caso, os usuários que interagiram com tais casas de banho.

Denúncia (D):

$$D \subseteq U \times T$$

Ou seja, a tabela denúncia é um subconjunto do produto cartesiano entre os usuários e as casas de banho, relativamente aos usuários que denunciaram tais casas de banho.

Reação (R):

$$R \subseteq U \times C$$

Ou seja, a tabela reação é um subconjunto do produto cartesiano entre os usuários e os comentários, mais especificamente os usuários que reagiram a tais comentários.

E como utilização real desses conjuntos no nosso projeto, podemos citar casos de **operações conjuntas** que indicam filtros para listagem de dados, como por exemplo:

Listar casas de banho sem denúncias do usuário:

Nesse contexto, temos um usuário u_x, que pretende fazer uma requisição de busca para casas de banho, para isso, não podemos exibir as que tenham sidas denunciadas por ele, isso em Teoria dos Conjuntos pode se representar como:

$$T_{resultado} = T_{total} - T_{denunciadas}$$

Onde:

- T_{total} : Conjunto de todas as casas de banho disponíveis.
- $T_{denunciadas}$: Subconjunto de T_{total}, contendo as casas de banho denunciadas pelo usuário u_x, formalmente:

$$T_{denunciadas} = \{ t \in T \mid (u_x, t) \in D \}$$

- $T_{resultado}$: Conjunto que contém apenas as casas de banho não denunciadas pelo usuário u_x.

Listar casas de banho ativas e não denunciadas

Na aplicação, temos casas de banho com estados diferentes, como ativas, desativadas, em análise e sugeridas. Quando exibimos uma lista de casas de banho para o usuário, sempre optamos pela escolha de ativas, e como dito no exemplo anterior, não denunciadas, isso em Teoria dos Conjuntos pode se como:

$$T_{resultado} = T_{ativas} \cap T_{não-denunciadas}$$

Onde:

- T_{ativas} : Subconjunto de T_{total}, contendo as casas de banho ativas.
- $T_{não-denunciadas}$: Subconjunto de T_{total}, contendo as casas de banho não denunciadas pelo usuário u_x, formalmente:

$$T_{não-denunciadas} = T_{total} - T_{denunciadas}$$

- $T_{resultado}$: Conjunto que contém apenas casa de banhos ativas e não denunciadas

Lógica Proposicional

No nosso projeto, diversos critérios de filtragem podem ser definidos por meio de proposições lógicas. Essas proposições representam condições específicas que combinamos para determinar quais dados serão exibidos ao usuário, utilizando operadores como "e", "ou", "não". Isso nos permite criar regras claras e eficientes para a aplicação, como nos exemplos:

Filtrar comentários com média de notas entre 4 e 5 e não denunciados pelo usuário

Embora essa funcionalidade de filtragem por notas não esteja implementada atualmente na aplicação, ela pode ser facilmente implementada no futuro, mas a ideia é parecida com as ditas na Teoria de Conjuntos, existe um usuário que pretende fazer uma requisição filtrada de comentários com a nota entre 4 e 5, e que idealmente não sejam denunciadas por ele, podemos referir isso utilizando lógica proposicional da seguinte maneira:

Definindo então as proposições, temos:

P : O comentário tem uma média de avaliação entre 4 e 5.
Q : O comentário não foi denunciado pelo usuário.

Logo, essa filtragem pode ser dada pela expressão:

$$P \ \wedge \ Q$$

Portanto, a expressão indica os comentários que atendem tanto à condição de média entre 4 e 5 quanto à condição de não terem sido denunciados, e podem ser retornados na requisição filtrada.

Filtrar casas de banho ativas e presentes em Lisboa

As casas de banho presentes na nossa base de dados, também possuem informações de localização, como cidade e país, nesse caso, podemos também fazer uma filtragem de cidades. Nesse caso em específico, existe um usuário que quer visualizar casas de banho exclusivamente em Lisboa, e como norma, também precisa estar ativa, podemos referir isso utilizando lógica proposicional da seguinte maneira:

Definindo então as proposições, temos:

P : A casa de banho está ativa.
Q : A casa de banho está presente em Lisboa.

Logo, essa filtragem pode ser dada pela expressão:

$$P \ \wedge \ Q$$

Portanto, a expressão indica as casas de banho que atendem tanto à condição de estar ativa e presente em Lisboa, e podem ser retornados na requisição filtrada.

Predicados

Os predicados são usados para descrever relações e condições específicas entre objetos, como por exemplo:

Reagir a um comentário

No sistema, os usuários podem reagir aos comentários de outros usuários, criando reações associadas a esses comentários. Para isso, definimos dois predicados importantes:

$R(u,\ c,\ t)$: O usuário u, reage a um comentário c, pelo tipo de reação t
$A(c,\ t)$: Cria uma reação do comentário c, com o tipo de reação t

Sendo:

- u : Usuário.
- c : Comentário.
- t : Tipo de reação (pode ser por exemplo, um Like, Dislike ou as denúncias predefinidas da aplicação).

Na nossa aplicação, a reação de um usuário para um comentário pode então, ser descrita da seguinte maneira:

$$R(u,\ c,\ t) \Leftrightarrow A(c,\ t)$$

O que pode ser descrito como “Se e somente se o usuário u, reage a um comentário c, pelo tipo de reação t, então é criado uma reação do comentário c, com o tipo de reação t”

Buscar casas de banho dentro dos limites de tela

No sistema, as casas de banho possuem informações de localização, como latitude e longitude, e o usuário deseja visualizar as casas de banho por um mapa, que precisa fazer uma consulta de casas de banho, dentro daquele limite de tela (para não precisar carregar todas as casas de banho, mostramos no mapa apenas as casas de banho nos limites das coordenadas).
Para isso, definimos os seguintes predicados:

$L(x_{max}, x_{min}, y_{max}, y_{min}, u)$: O usuário u tem os limites de tela definidos pela latitude máxima x_{max} e mínima x_{min}, e longitude máxima y_{max} e mínima y_{min}.
$D(x_{max}, x_{min}, y_{max}, y_{min}, t)$: É mostrada a casa de banho t , dentro dos limites de latitude máxima x_{max} e mínima x_{min}, e longitude máxima y_{max} e mínima y_{min}.

Sendo:

- x_{max} : Latitude máxima.
- x_{min} : Latitude mínima.
- y_{max} : Longitude máxima.
- y_{min} : Longitude máxima.
- u : Usuário
- t : Casa de banho

Na nossa aplicação, essa listagem de casas de banho nos limites de tela do usuário podem ser descritas da seguinte maneira:

$$L(x_{max}, x_{min}, y_{max}, y_{min}, u) \Rightarrow D(x_{max}, x_{min}, y_{max}, y_{min}, t)$$

O que pode ser descrito como “Se usuário usuário u tem os limites de tela definidos pela latitude máxima x_{max} e mínima x_{min}, e longitude máxima y_{max} e mínima y_{min}, é mostrada a casa de banho t , dentro dos limites de latitude máxima x_{max} e mínima x_{min}, e longitude máxima y_{max} e mínima y_{min}”.

Conclusão

O uso de conceitos da **Matemática Discreta** no projeto **Think Toilet** tem sido fundamental para garantir a eficiência e a lógica na manipulação e exibição dos dados. A **Teoria dos Conjuntos** tem sido aplicada na organização e relacionamento entre as tabelas da base de dados, permitindo uma consulta eficaz e uma estrutura clara para o sistema. A **Lógica Proposicional** tem sido essencial para definir filtros e condições que determinam quais informações são exibidas para os usuários, garantindo que eles recebam apenas dados relevantes. Além disso, os **Predicados** são cruciais para descrever de forma precisa as interações entre usuários, casas de banho e comentários, possibilitando um sistema altamente dinâmico e interativo.

Esses conceitos não apenas facilitam o desenvolvimento da aplicação, mas também garantem que a manipulação de dados e a experiência do usuário sejam realizadas de maneira otimizada e sem erros. O **Think Toilet** é, assim, um exemplo claro de como os fundamentos da **Matemática Discreta** podem ser aplicados com sucesso no desenvolvimento de soluções tecnológicas voltadas para o bem-estar e a praticidade no cotidiano dos usuários.