RESOLUÇÃO PARCIAL PARA A PROPOSTA DE PÁGINA INICIAL DO USUÁRIO LOGADO

Rifa (MODEL)

```
class Rifa extends Model {
  // ... Outras coisas

  bilhetes() {
    return this.hasMany("App/Models/Bilhete");
  }
}
```

Explicação:

- Uma **Rifa** tem muitos **Bilhetes**.

User (MODEL)

```
class User extends Model {
  // ... Outras coisas

  bilhetesComprados() {
    return this.hasMany("App/Models/Bilhete");
  }

  rifasCompradas() {
    return this.manyThrough("App/Models/Bilhete", "rifa");
  }
}
```

Explicação:

- Um **User** tem muitos **Bilhetes** comprados (método *bilhetesComprados*). Mapeado pelo atributo **user_id** na tabela **bilhetes** no banco de dados.
- Um **User** tem muitas **Rifas** compradas (método *rifasCompradas*), porém esta propriedade não pode ser mapeada diretamente, visto que é a entidade **Bilhete** que faz a relação entre **User** e **Rifa**, por este motivo utilizamos um **manyThrough** aqui. Este método retorna um objeto **Rifa** para cada **Bilhete** comprado, gerando duplicação de registros (que será resolvida no controller através de um **distinct**).

Controller

```
async index({ view, auth }) {
  const user = auth.user;

  const minhasRifas = (await user.rifas().fetch()).rows;

  const rifasCompradas = (
    await user
      .rifasCompradas()
      .distinct("rifas.id")
      .fetch()
  ).rows;

  for (const rifa of rifasCompradas) {
    rifa.totalBilhetesComprados = (
      await user
        .bilhetesComprados()
        .where("rifa_id", rifa.id)
        .count("* as total")
    )[0].total;

    rifa.valorInvestido = rifa.totalBilhetesComprados *
  rifa.valor_bilhete;

    rifa.totalBilhetesCompradosGeral = (
      await rifa
        .bilhetes()
        .whereNotNull("user_id")
        .count("* as total")
    )[0].total;

    rifa.totalBilhetes = (
      await rifa.bilhetes().count("* as total")
    )[0].total;
  }
  return view.render("rifas.index", { minhasRifas, rifasCompradas });
}
```

Explicação:

- A variável **minhasRifas** contém a lista com todas as rifas que o usuário criou (linha 4).
- A variável **rifasCompradas** contém a lista com todas as rifas que o usuário comprou (linha 6). Ela se baseia no *manyThrough* **rifasCompradas** do model *User*.
- Na linha 13 inicia-se um laço de repetição dentre todas as rifas que o usuário participa (comprou bilhetes), com o objetivo de calcular algumas métricas para cada uma.
- Na linha 14 é "injetada" a propriedade **totalBilhetesComprados** em cada uma das rifas. Esta propriedade é derivada do *hasMany* **bilhetesComprados** de *User*, com um where para selecionar apenas a rifa atual e um **count**.
- Na linha 21 é calculado o valor que o usuário investiu (R$) para a rifa em questão.
- Na linha 23 é calculado o total de bilhetes já comprados para a rifa em questão, indiferente de qual usuário comprou.
- Na linha 30 é calculado o total de bilhetes que a rifa possui.
- **Linha 34**: como as propriedades foram injetadas diretamente em cada **Rifa**, basta passar **minhasRifas** e **rifasCompradas** para a view.

view.edge

```
<h2>Minhas Rifas</h2>
<div class="rifas">
  @each(rifa in minhasRifas)
    <div class="rifa">
      <div>{{ rifa.titulo }}</div>
    </div>
  @endeach
</div>

<h2>Rifas que eu participo</h2>
<div class="rifas">
  @each(rifa in rifasCompradas)
    <div class="rifa">
      <div>{{ rifa.titulo }}</div>
      <div>Meus Bilhetes: {{ rifa.totalBilhetesComprados }}</div>
      <div>Valor Investido: {{ rifa.valorInvestido }}</div>
      <div>Bilhetes comprados no geral / Total de Bilhetes: {{
rifa.totalBilhetesCompradosGeral }} / {{ rifa.totalBilhetes }}</div>
    </div>
  @endeach
</div>
```