

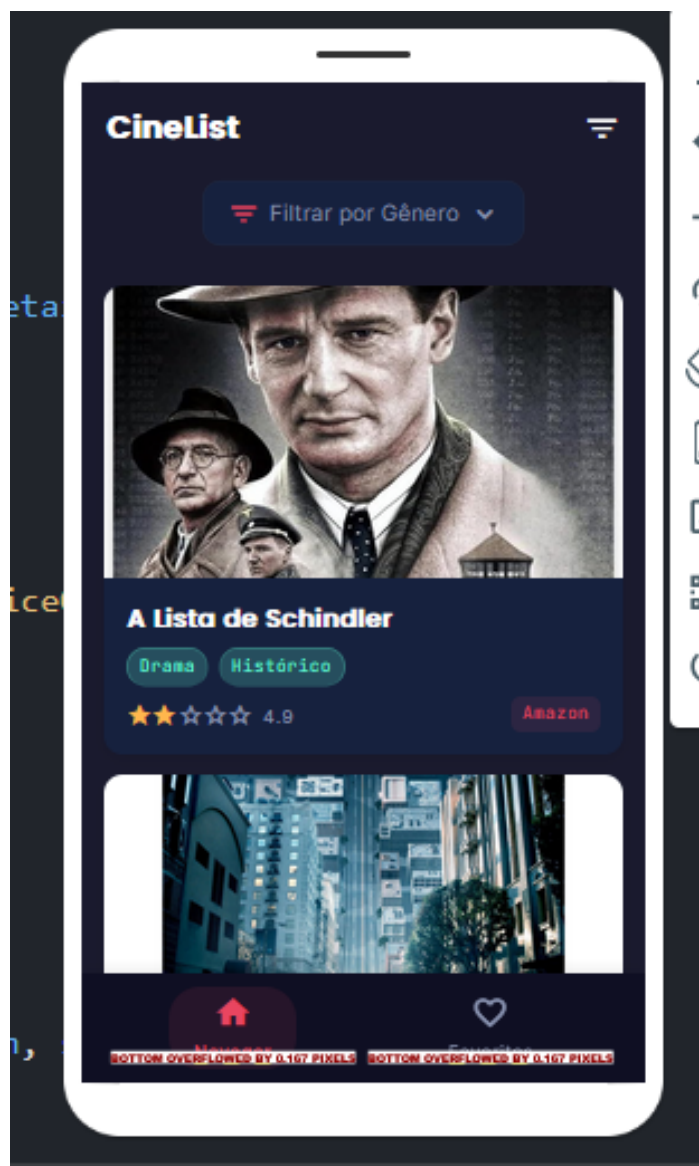
Relatório de Execução de Tarefas

– Desafio Bridge 2025

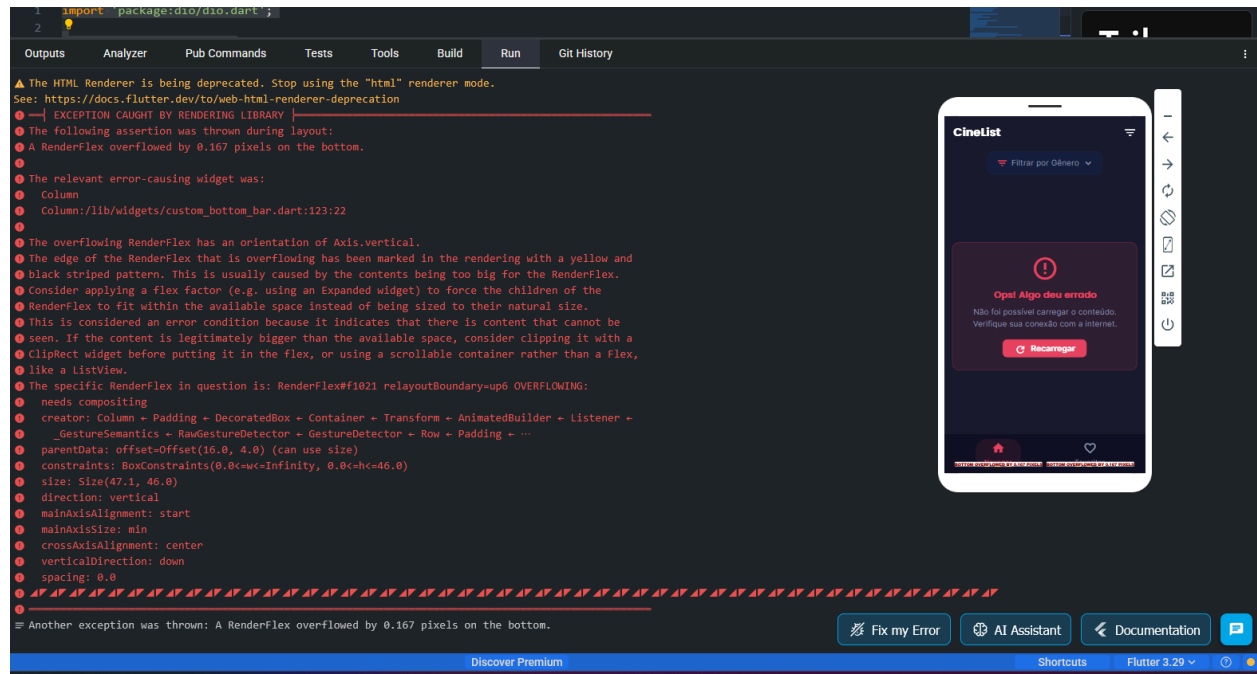
1. Introdução e Objetivo do Projeto

Este relatório documenta a execução de 6 (seis) tarefas de desenvolvimento propostas no Desafio Bridge 2025. O objetivo central do projeto foi aprimorar um aplicativo de catálogo de filmes e séries, focando na melhoria da experiência do usuário (UX), na refatoração da estrutura de dados e na implementação de novas funcionalidades essenciais. As tarefas abrangeram desde a correção de layouts e tratamento de dados da API até a implementação de um sistema de persistência local para a funcionalidade de "Favoritos".

Primeira versão do app:



OBS: Porém antes de iniciar qualquer mudança, percebi que ao rodar o programa sem qualquer alteração um aviso foi dado:



Como se trata de um erro de overflow por menos de 1 pixel, preferi focar primeiramente nas tarefas mais específicas, porém não pude deixar de notar que esta existe.

2. Detalhamento das Tarefas Executadas

Task 1: Correção na Visualização de Detalhes do Conteúdo

- **Problema Identificado:** A tela de detalhes de um filme ou série não exibia informações cruciais como ano de lançamento e duração, comprometendo a experiência do usuário.
 - **Solução Implementada:** As informações de ano e duração foram corretamente extraídas do modelo de dados (**Medium**) e integradas à interface do usuário.
- Resultado:** A tela de detalhes agora apresenta as informações completas, conforme o design esperado

Antes:



Depois:



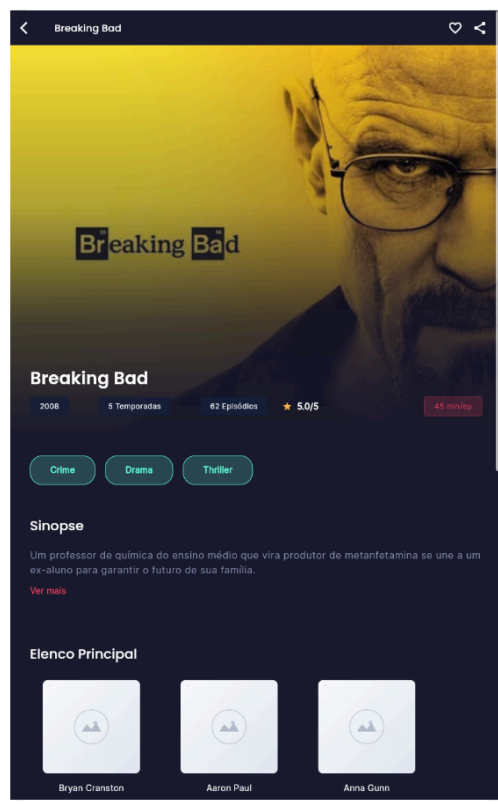
Criei uma função para facilitar a visualização dos campos que faltavam. Agora é só fazer uma simples chamada para cada um, e caso mais sejam vistos necessários também é fácil utilizar.

É visível que há um problema com a visualização da nota do filme/série mostrado caso a dimensão do celular não tenha largura suficiente, porém isso foi considerado secundário para focar no desenvolvimento das tarefas principais.

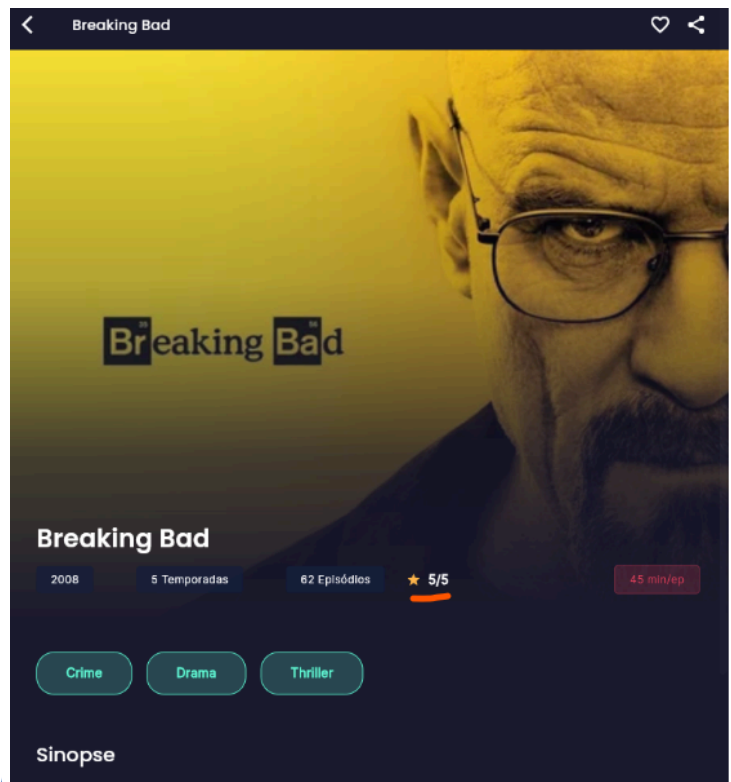
Task 2: Padronização da Exibição de Avaliações

- **Problema Identificado:** A avaliação numérica do conteúdo estava sendo exibida em uma escala de 10 pontos (ex: "X/10"), o que gerava uma inconsistência visual e cognitiva com a representação gráfica de 5 estrelas utilizada na interface.
- **Solução Implementada:** A lógica de exibição da avaliação foi ajustada para normalizar a nota para uma escala de 5 pontos (ex: "X/5"). Essa mudança alinha o valor numérico diretamente com o sistema de estrelas, tornando a informação mais clara e intuitiva para o usuário.
- **Resultado:** A exibição da nota foi padronizada em todo o aplicativo, eliminando a inconsistência e melhorando a clareza da interface de avaliações. Durante a tarefa, foi observada uma divergência pontual entre a nota principal e a da seção "Avaliações do Usuário" em alguns itens, registrada como um ponto para investigação futura.

Antes:

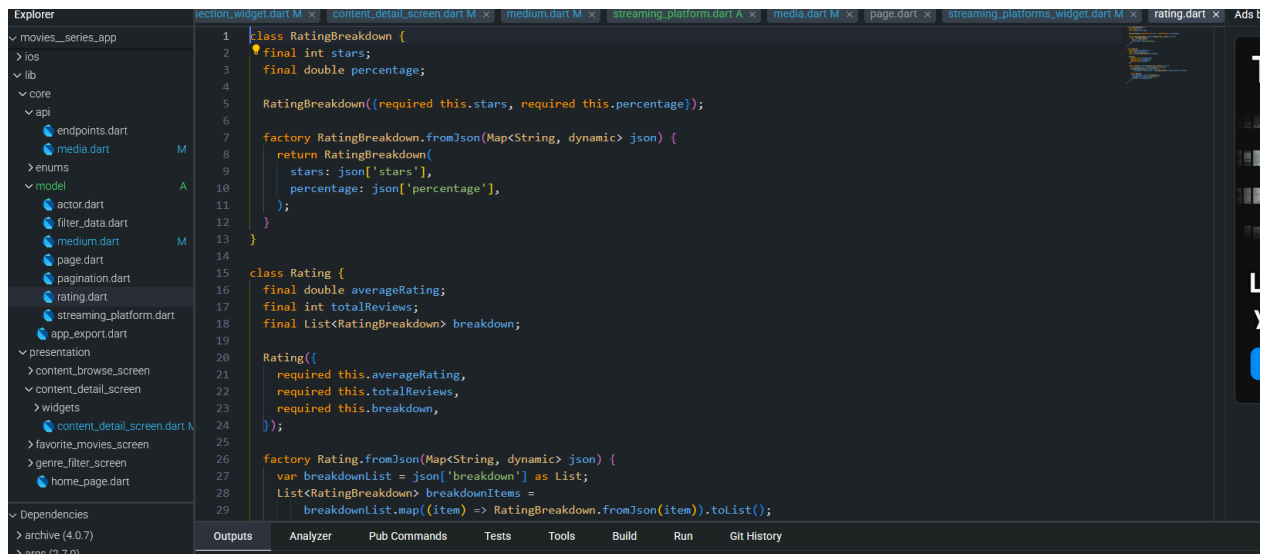


Depois:



Task 3: Refatoração do Modelo de Dados de Avaliações

- **Problema Identificado:** A estrutura de dados para representar as avaliações de conteúdo dentro do código era genérica e pouco estruturada. Isso dificultava a manutenção e não estava preparado para futuras expansões da funcionalidade.
- **Solução Implementada:** Para resolver essa questão, foi criado um modelo de dados dedicado (`rating.dart`). Este arquivo agora encapsula todas as informações pertinentes às avaliações (nota média, total de votos, etc.) em uma classe específica, centralizando a lógica e tornando a representação dos dados mais clara e robusta.
- **Resultado:** Com a nova estrutura, o código se tornou mais organizado, legível e escalável, facilitando a adição de novas funcionalidades relacionadas às avaliações no futuro.



```
1 class RatingBreakdown {
2   final int stars;
3   final double percentage;
4
5   RatingBreakdown({required this.stars, required this.percentage});
6
7   factory RatingBreakdown.fromJson(Map<String, dynamic> json) {
8     return RatingBreakdown(
9       stars: json['stars'],
10      percentage: json['percentage'],
11    );
12  }
13 }
14
15 class Rating {
16   final double averageRating;
17   final int totalReviews;
18   final List<RatingBreakdown> breakdown;
19
20   Rating({
21     required this.averageRating,
22     required this.totalReviews,
23     required this.breakdown,
24   });
25
26   factory Rating.fromJson(Map<String, dynamic> json) {
27     var breakdownList = json['breakdown'] as List;
28     List<RatingBreakdown> breakdownItems =
29       breakdownList.map((item) => RatingBreakdown.fromJson(item)).toList();
30   }
31 }
```

Task 4: Consumo Dinâmico de Dados de Plataformas de Streaming

- **Problema Identificado:** As informações sobre as plataformas de streaming onde cada conteúdo estava disponível eram estáticas e "hardcoded" (fixas no código-fonte). Isso impedia que o aplicativo exibisse os dados reais e atualizados fornecidos pela API, resultando em informações potencialmente incorretas para o usuário e dificultando a manutenção.
- **Solução Implementada:** Após análise do JSON retornado pelo endpoint `/media`, foi confirmado que a API já fornecia uma lista dinâmica das plataformas de streaming para cada título. O código foi então refatorado para parsear e utilizar esses dados. A estrutura de dados interna e os widgets da interface foram ajustados para consumir diretamente as informações da API, eliminando a necessidade de qualquer dado estático.
- **Resultado:** O aplicativo agora exibe de forma precisa e dinâmica as plataformas de streaming para cada filme ou série, refletindo fielmente os dados do servidor. A solução aumenta a confiabilidade da informação para o usuário e torna o sistema mais flexível e de fácil manutenção.

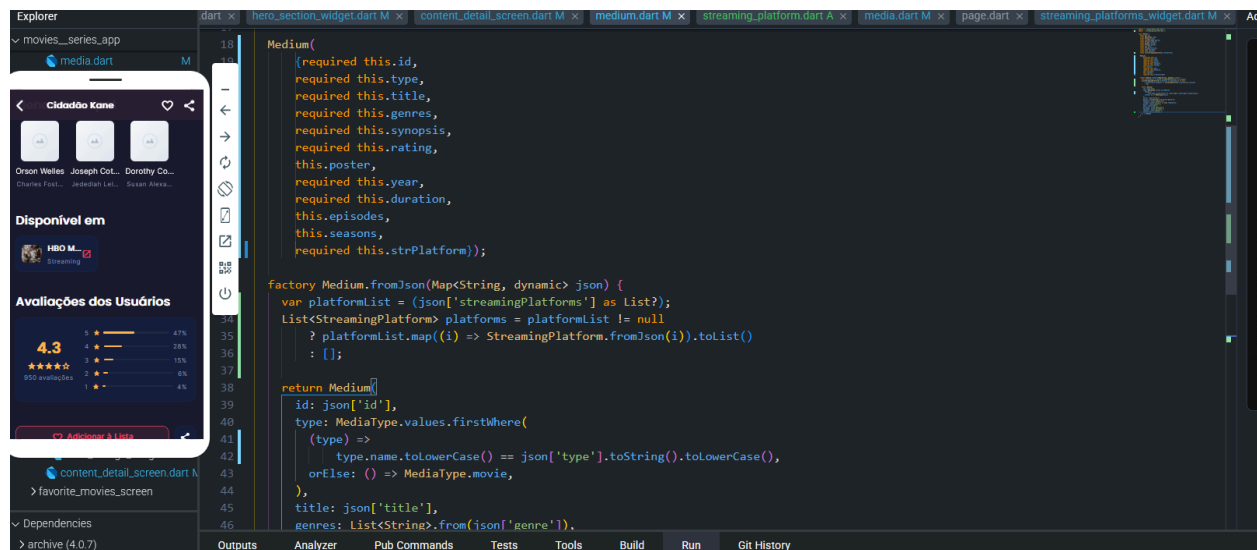
Retorno da API de media:

```
Save Copy Collapse All Expand All Filter JSON
{
  synopsis: "Baseado em uma história real, Oskar Schindler, um empresário alemão, arrisca sua vida e fortuna para salvar centenas de judeus durante o Holocausto, mostrando o poder da compaixão em tempos sombrios.",
  year: 1993,
  duration: "195 min",
  1: {
    id: 6,
    title: "A Origem",
    type: "movie",
    genre: {
      0: "Ficção Científica",
      1: "Suspense"
    },
    rating: 4.9
  },
  streamingPlatforms: {
    0: {
      name: "Amazon Prime",
      type: "rent",
      logoUrl: "https://via.placeholder.com/100x50?text=Prime",
      deeplink: "https://www.amazon.com/dp/B0047NJJ1G"
    },
    1: {
      name: "Netflix",
      type: "streaming",
      logoUrl: "https://via.placeholder.com/100x50?text=Netflix",
      deeplink: "https://www.netflix.com/title/70131314"
    },
    poster: "https://uaposters.com.br/media/catalog/product/cache/1/thumbnail/800x330/df278eb31525d88d6e5f8d27136e95/5/0/508120201013-uau-posters-filmes-a-origem-inception.jpg",
    synopsis: "Don Cobb, um ladrão especializado em extrair segredos do subconsciente, é oferecido uma chance de redenção ao realizar a tarefa impossível de implantar uma ideia na mente de um alvo.",
    year: 2010,
    duration: "148 min"
  },
  2: {
    id: 33,
    title: "Breaking Bad",
    type: "series",
    duration: "45 min/ep",
    synopsis: "Um professor de química do ensino médio que vira produtor de metanfetamina se une a um ex-aluno para garantir o futuro de sua família."
  }
}
```

Parte hard coded em medium.dart:

```
13
14 class ContentDetailScreen extends StatelessWidget {
15   final Medium medium;
16   final List<Map<String, dynamic>>? streamingPlatforms;
17
18   const ContentDetailScreen({
19     super.key,
20     required this.medium,
21     this.streamingPlatforms = const [
22       {
23         "name": "Netflix",
24         "type": "Streaming",
25         "logoUrl": "https://images.unsplash.com/photo-1611162617474-5b21e879e113?fm=jpg&q=60&w=3000&ixlib=rb-4.0.3",
26         "deepLink": "netflix://title/80057281"
27       },
28       {
29         "name": "Amazon Prime",
30         "type": "Aluguel",
31         "logoUrl": "https://images.unsplash.com/photo-1560472354-b33ff0c44a43?fm=jpg&q=60&w=3000&ixlib=rb-4.0.3",
32         "deepLink": "primevideo://detail/0GZQT3YWHGWCKV"
33       }
34     ],
35   });
36
37   @override
38   Widget build(BuildContext context) {
39     return Scaffold(
40       backgroundColor: AppTheme.primaryDark,
```

Após alterações:



```
18 Medium(
19   (required this.id,
20   required this.type,
21   required this.title,
22   required this.genres,
23   required this.synopsis,
24   required this.rating,
25   this.poster,
26   required this.year,
27   required this.duration,
28   this.episodes,
29   this.seasons,
30   required this.strPlatform));
31
32 factory Medium.fromJson(Map<String, dynamic> json) {
33   var platformList = (json['streamingPlatforms'] as List?)
34     ? platformList.map((i) => StreamingPlatform.fromJson(i)).toList()
35     : [];
36
37   return Medium(
38     id: json['id'],
39     type: MediaType.values.firstWhere(
40       (type) =>
41         type.name.toLowerCase() == json['type'].toString().toLowerCase(),
42       orElse: () => MediaType.movie,
43     ),
44     title: json['title'],
45     genres: List<String>.from(json['genre']),
46
```


Task 5: Implementação e Persistência da Funcionalidade de Favoritos

- **Problema Identificado:** A funcionalidade de "favoritar" um item era apenas um efeito visual momentâneo. O estado de favorito era perdido assim que o usuário saía da tela de detalhes, não havendo persistência dos dados. Adicionalmente, foi identificada uma inconsistência de estado entre o ícone de favoritar no topo da página e o botão de ação correspondente na parte inferior.
- **Solução Implementada:** A solução foi dividida em duas frentes:
 1. **Persistência de Dados:** Foi implementado um sistema de persistência local utilizando o pacote `shared_preferences`. Agora, ao favoritar um item, seu ID é salvo em uma lista no armazenamento do dispositivo, garantindo que a escolha do usuário seja mantida entre diferentes sessões de uso do aplicativo.
 2. **Sincronização de Estado:** O gerenciamento de estado na tela `content_detail_screen` foi refatorado para garantir que ambos os botões (o `IconButton` no topo e o `ActionButtonsWidget` no final da página) compartilhassem e reagissem à mesma fonte de estado. Isso assegura que, ao interagir com um deles, o outro seja atualizado instantaneamente, mantendo a consistência da interface.
- **Resultado:** A funcionalidade de "Favoritos" agora está totalmente funcional e consistente. As escolhas do usuário são salvas de forma persistente no dispositivo, e a interface reflete o estado correto em todos os seus componentes. Como melhoria futura, foi identificada a oportunidade de exibir um indicador de "favorito" também nos itens listados na tela inicial do aplicativo (mas considerando que já existe a aba "Favoritos", talvez não seja tão necessário).

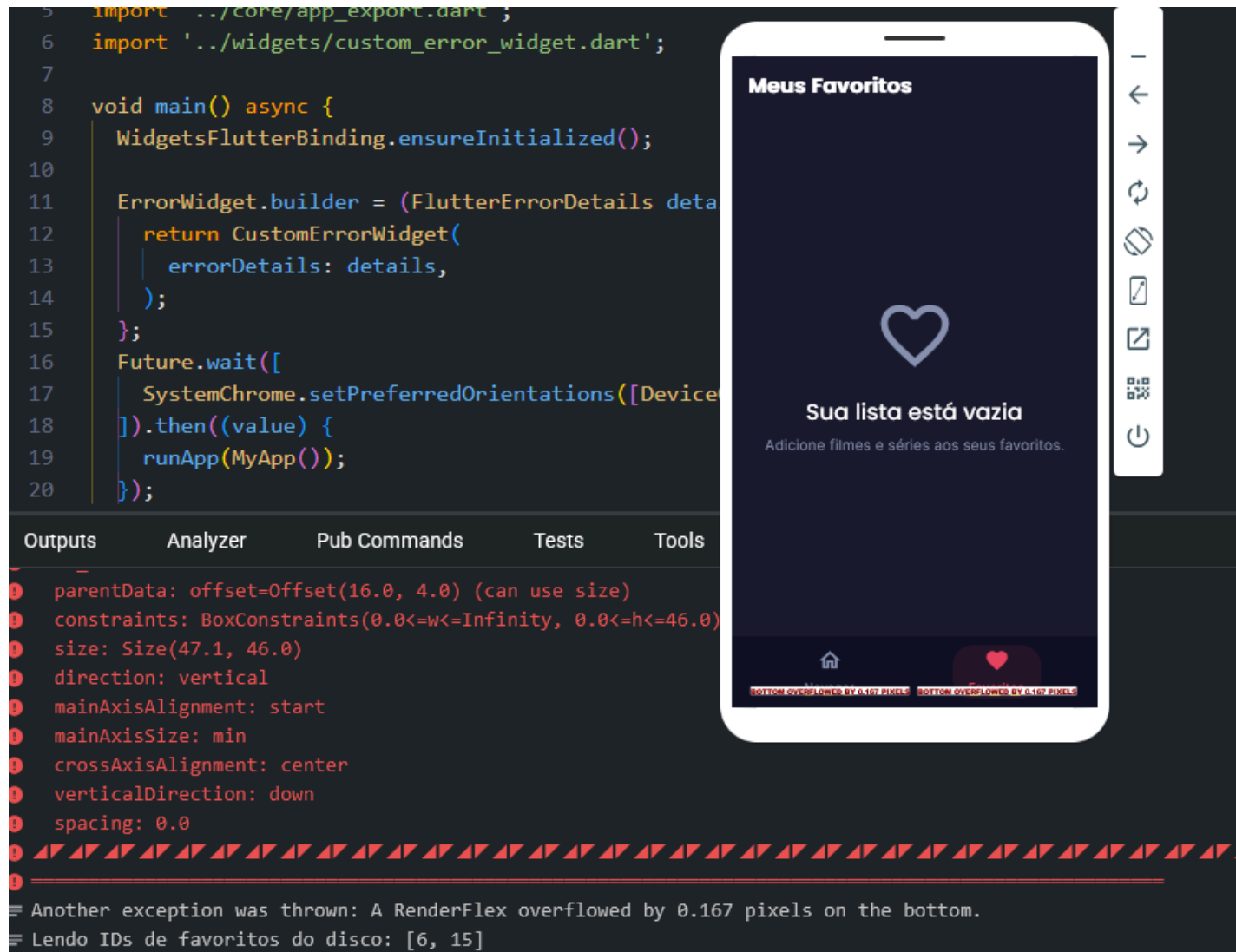


Imagem das duas partes sincronizadas.

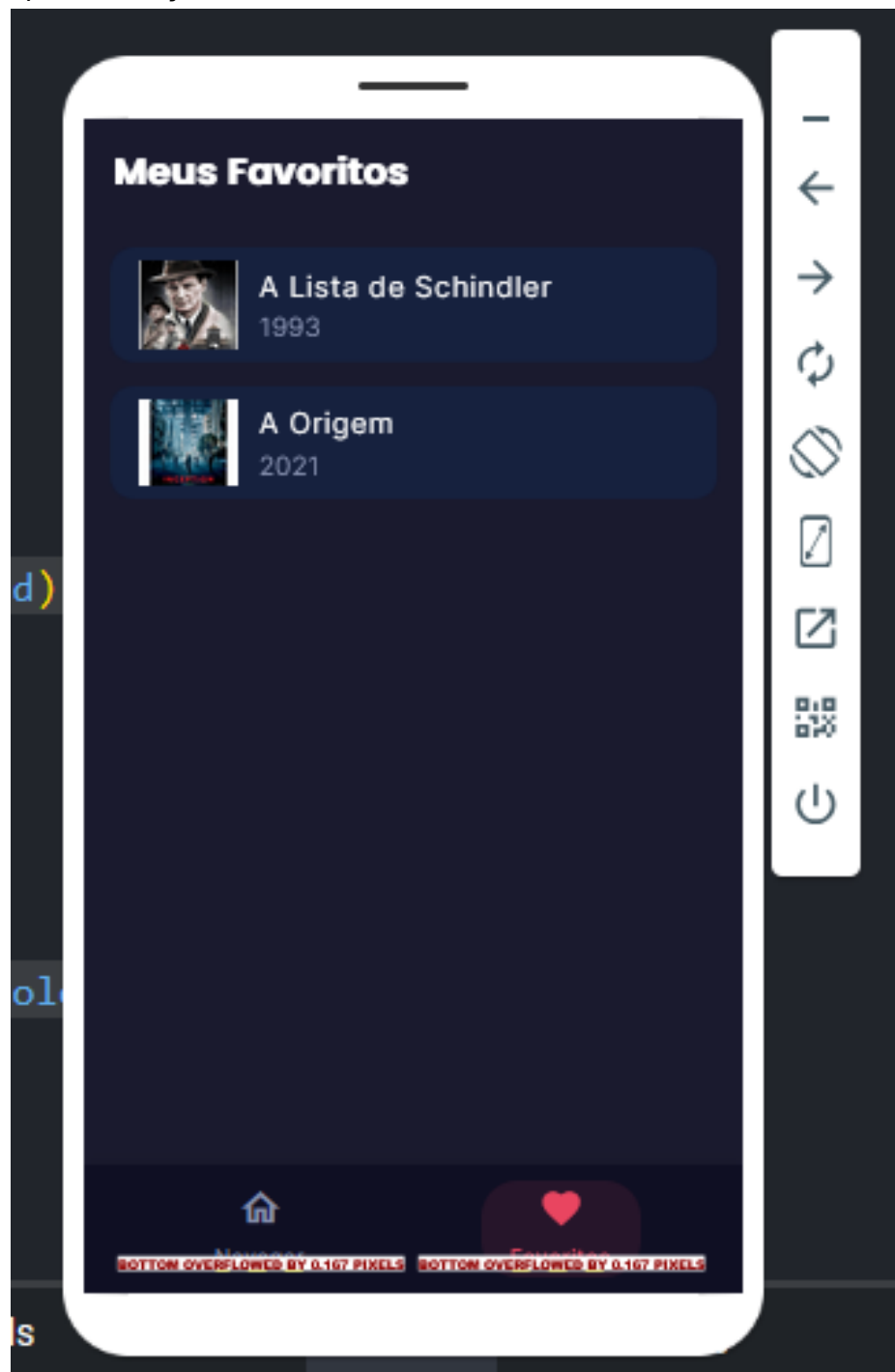
Task 6: Desenvolvimento da Tela de Exibição de Favoritos

- **Problema Identificado:** O aplicativo permitia ao usuário marcar títulos como favoritos, mas não oferecia uma tela dedicada para visualizar essa lista. A ausência desse fluxo fundamental tornava a funcionalidade incompleta e limitava o engajamento do usuário.
- **Solução Implementada:** Foi desenvolvida a `FavoriteMoviesScreen`, uma nova tela dedicada a exibir os conteúdos favoritos. A lógica implementada recupera a lista de IDs salvos localmente (`shared_preferences`), utiliza esses IDs para filtrar os itens correspondentes da lista geral obtida da API e os renderiza para o usuário. A tela também trata estados de interface importantes, como um indicador de carregamento (`loading`) e uma mensagem informativa para quando a lista está vazia.
- **Resultado:** A implementação resultou em um fluxo de usuário completo para a funcionalidade de "Favoritos". Os usuários agora podem salvar e revisitar facilmente seu conteúdo preferido, o que melhora significativamente a experiência de uso e a utilidade geral do aplicativo.
- **Oportunidades de Melhoria Identificadas:** Durante a execução e revisão do projeto, foram identificados outros pontos que poderiam ser aprimorados em um próximo ciclo de desenvolvimento:
 - **Layout da Tela de Favoritos:** Aprimorar o espaçamento e o layout da lista para uma visualização mais agradável.
 - **Padronização da Escala de Filtro:** Ajustar a escala de notas na funcionalidade de filtro de 0-10 para 0-5, mantendo a consistência com o restante da interface.
 - **Exibição do Elenco:** Corrigir a exibição dos avatares dos atores na seção de elenco.
 - **Funcionalidade de Compartilhar:** Implementar a lógica para o botão de compartilhar localizado no topo da tela de detalhes.

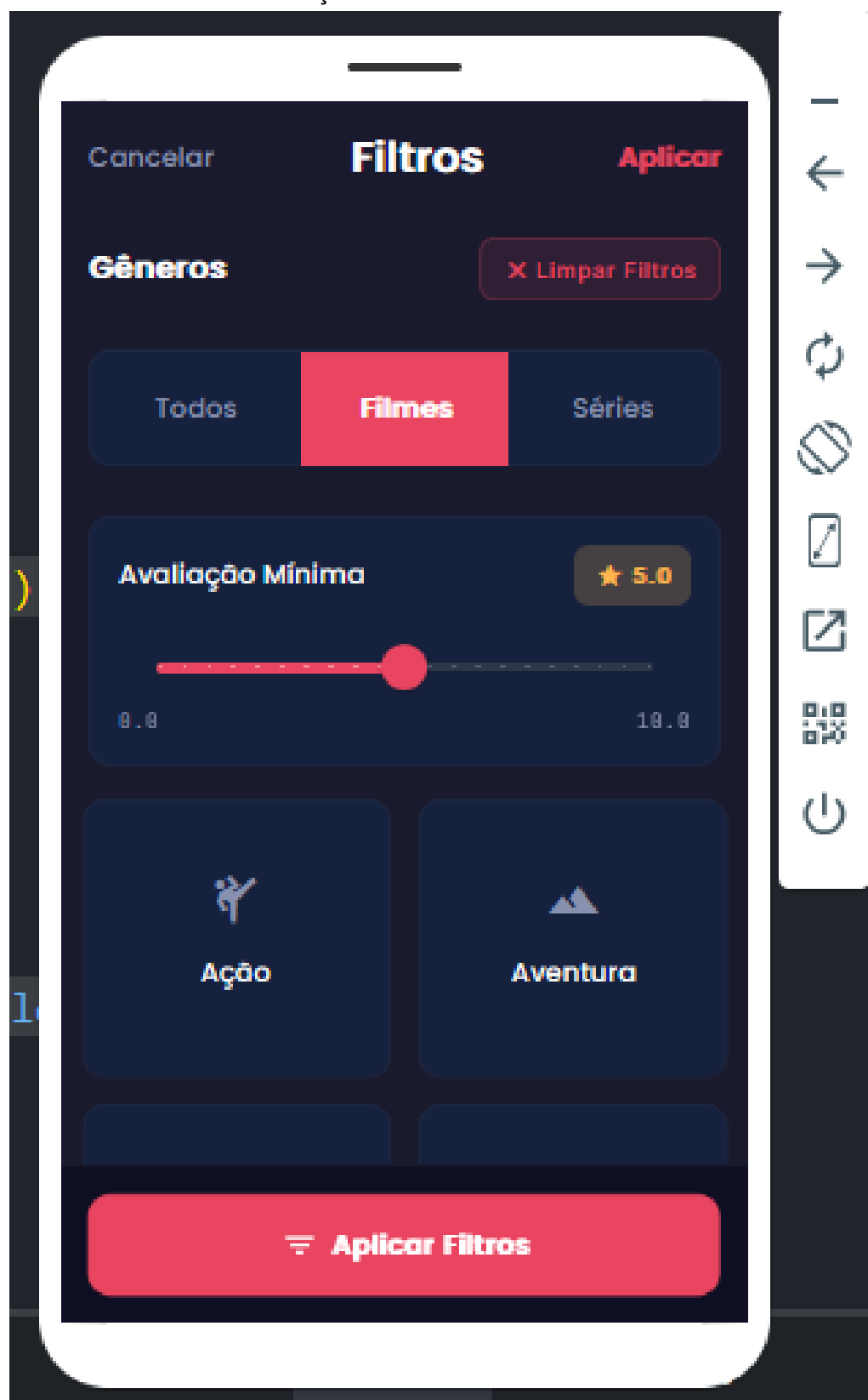
Página relatando lista vazia apesar de existirem itens favoritados (como pode-se ver no terminal):



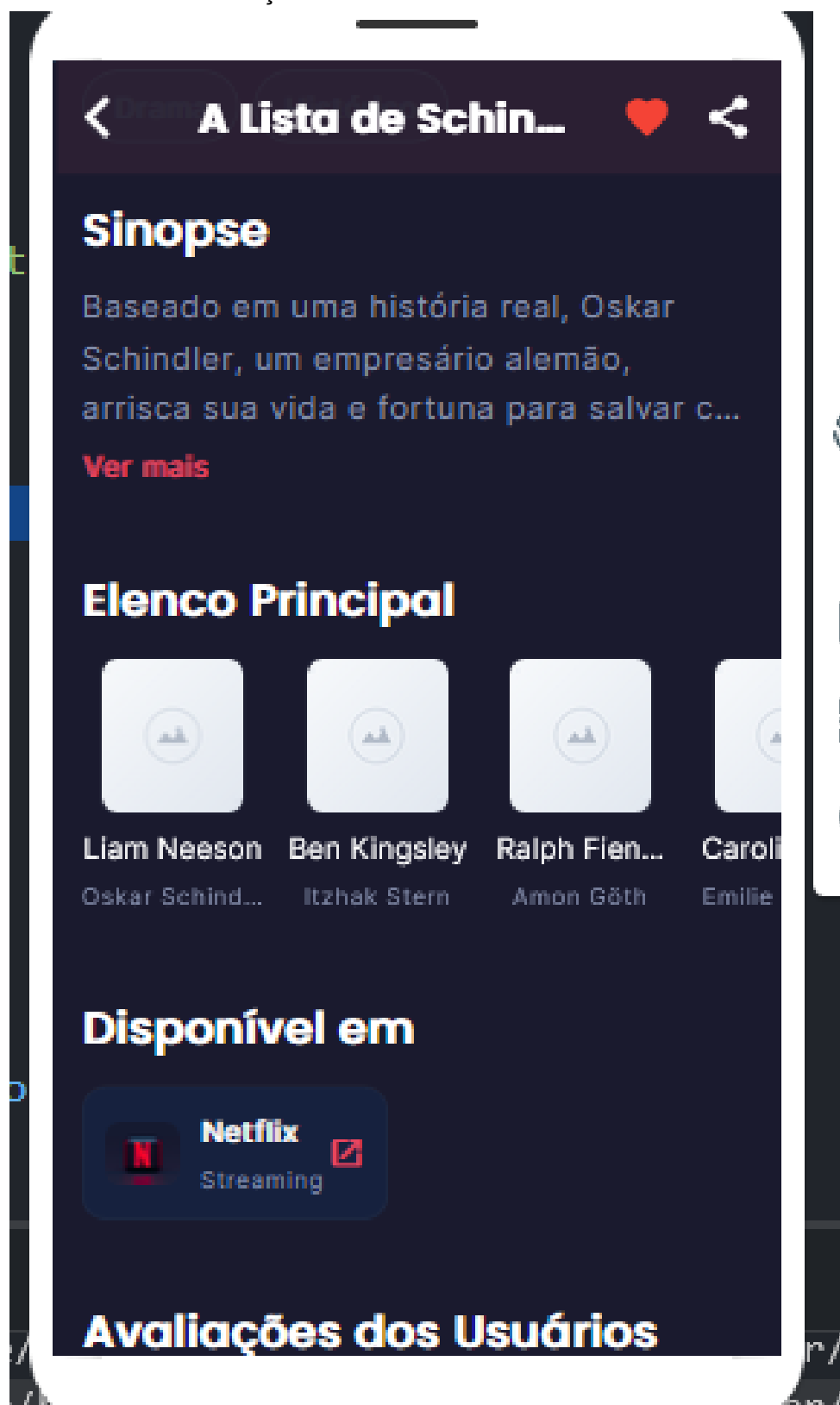
Após mudanças:



Inconsistência da Padronização da Escala de Filtro:



Inconsistência de Exibição do Elenco:



Inconsistência da Funcionalidade de Compartilhar :

