Parte 3: Questionário de estudo
Responda às seguintes perguntas com base na sua experiência prática nas Partes

1 e 2. O objetivo é entender o "porquê" de cada ferramenta.
1. Definição e Propósito:

- **Qual é a principal função do SonarCloud no seu processo de desenvolvimento?**

O SonarCloud é uma plataforma de análise estática no servidor que:
analisa o repositório inteiro (todas as branches configuradas);
mede a qualidade geral do projeto (bugs, vulnerabilidades, code smells, cobertura de testes, duplicações);
aplica um Quality Gate com regras compartilhadas pelo time;
guarda o histórico de qualidade ao longo do tempo (para cada commit/PR).
Em resumo: ele garante a qualidade do código em nível de projeto/repositório e ajuda a impedir que código ruim seja integrado na main.

- **Qual é a principal função do SonarLint?**

O SonarLint é um analisador estático dentro da IDE que:

faz a análise em tempo real, enquanto você edita;
sublinha problemas diretamente no código (bugs, vulnerabilidades, code smells);
orienta a correção imediatamente, antes mesmo do commit.
Em resumo: ele atua como um “linter inteligente” local, focado em dar feedback rápido ao desenvolvedor.

2. Momento do Feedback (Timing):

- **Em que momento você recebeu o feedback do SonarLint?**

O feedback do SonarLint veio:

enquanto eu digitava ou logo após salvar o arquivo na IDE.
Assim que eu abria ou alterava um arquivo, os problemas já apareciam sublinhados, sem precisar rodar pipeline nem fazer push.

- **Em que momento você recebeu o feedback do SonarCloud?**

Depois que eu subi o código para o GitHub (push) e a análise foi executada automaticamente (ou via GitHub Actions/CI).
Ou seja, o retorno veio após o commit/push, no contexto do projeto inteiro.

3. Escopo da Análise:

- **Quando o SonarLint analisa seu código, qual é o escopo dele? (Ex: O projeto inteiro, a pasta, o arquivo?)**

No meu uso, o SonarLint:
Analisa principalmente o arquivo aberto/que estou editando;
Dependendo da IDE e da configuração, pode analisar outros arquivos do projeto, mas o foco é sempre no código que está sendo visto/editado naquele momento.
Então, o escopo é basicamente local/por arquivo, dentro do projeto aberto na IDE.

- **Qual é o escopo da análise do SonarCloud?**

O SonarCloud:
analisa o projeto inteiro do repositório (o código de todas as pastas/arquivos incluídos na análise);
calcula métricas globais (cobertura de testes do projeto, duplicação geral, tendências de bugs, etc.).
Ou seja, o escopo é global, em nível de repositório/projeto.

4. O "Quality Gate":

- **No painel do SonarCloud, você viu um indicador "Passed" (Verde) ou "Failed" (Vermelho). O que é o "Quality Gate" (Portão de Qualidade) e por que ele é importante para um time de desenvolvimento?**

Um Quality Gate é um conjunto de critérios de qualidade definidos para o projeto, por exemplo:

zero vulnerabilidades de severidade crítica;
menos de X% de bugs em código novo;
cobertura de testes mínima (por ex., 80%) em código novo;
limite de code smells em código novo.
O SonarCloud usa esses critérios para dizer se o projeto (ou a mudança recente) está:

“Passed” (Verde) – atende aos requisitos de qualidade;
“Failed” (Vermelho) – não atende e precisa de correções.

5. A Sinergia das Ferramentas:

- **Por que é útil ter ambas as ferramentas (SonarCloud e SonarLint)? Por que não usar apenas o SonarLint na IDE, já que ele mostra os erros na hora?**

Se eu usasse apenas o SonarLint:

eu teria feedback local, mas:
não teria Quality Gate;
não teria métricas globais (cobertura, duplicação, tendências);
cada desenvolvedor poderia estar usando regras diferentes, sem padronização;
problemas que só aparecem no contexto de todo o projeto ou da integração poderiam passar despercebidos.
Portanto, as duas ferramentas se complementam:

SonarLint ajuda a não “criar problemas”,
SonarCloud garante que, mesmo assim, o projeto como um todo permanece dentro dos padrões.

6. Modo Conectado (Connected Mode):

- **Qual é a principal vantagem de configurar o "Modo Conectado" (vincular o SonarLint ao SonarCloud), como feito no passo 5 da Parte 2? (Dica: Pense sobre "regras" e "consistência").**

A principal vantagem do Modo Conectado é:
o SonarLint passa a usar as mesmas regras e perfis de qualidade do SonarCloud para aquele projeto.
O Modo Conectado garante que IDE e servidor falem a mesma língua de qualidade.

7. Exploração de Problemas:

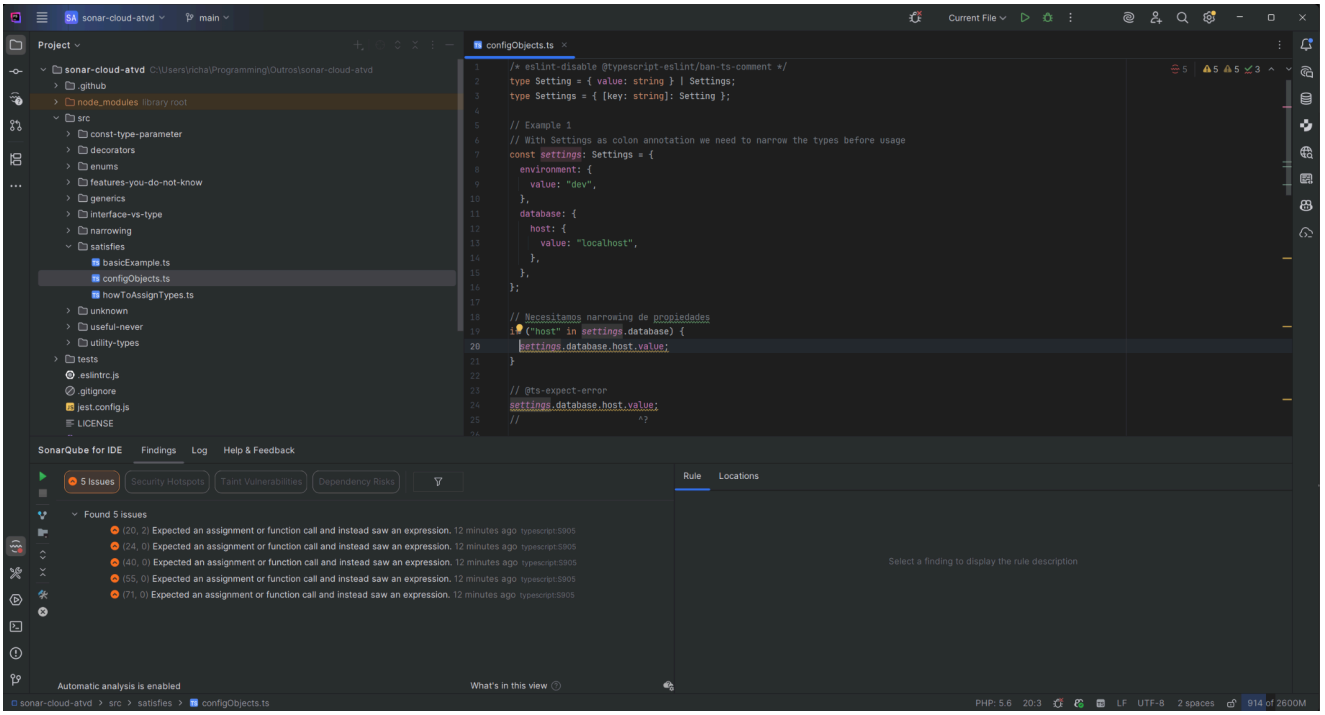
- Escolha um "Code Smell" que o SonarLint ou o SonarCloud encontrou no seu projeto. Explique por que a ferramenta marcou aquilo como um problema e qual seria a forma correta de escrever aquele código.

“Unexpected empty class.”
export class EndpointPayload {}

A classe EndpointPayload está se nenhum corpo, logo ela não será usada para nenhuma finalidade, o que representa codigo desnecessário na aplicação dificultando a manutenibilidade.

Entregáveis:

Screenshot CodeSmell:



Disponível na pasta “Respostas Atividade/SonarQube for IDE.png”

Link Dashboard Publico SonarCloud:
https://sonarcloud.io/project/overview?id=richardgss_sonar-cloud-atvd

Link Repositório Github:
https://github.com/richardgss/sonar-cloud-atvd