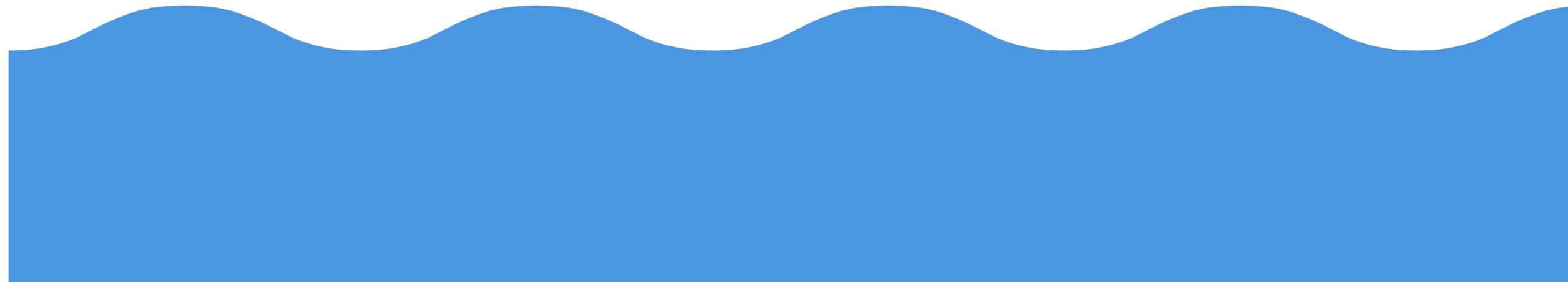


SAÚDE+



SOBRE O PROJETO

Desenvolver um software de padronização e interpretação de prontuários médicos entre diferentes especialidades

- Criação e armazenamento de prontuários médicos digitalizados
- Interpretação automatizada de dados clínicos via API de IA (StackpotIA)
- Padronização de registros médicos
- Geração de relatórios e exportação em PDF
- Interface intuitiva e segura, desenvolvida com Java Spring Boot, MongoDB e HTML/CSS/JS

ODS



9 INDÚSTRIA,
INovação e
INFRAESTRUTURA



10 REDUÇÃO DAS
DESIGUALDADES



PROBLEMÁTICA



- Prontuários médicos sem padronização, com registros inconsistentes entre profissionais e especialidades.
- Dificuldade na interpretação e análise dos dados clínicos, gerando retrabalho e risco de erro.
- Tempo excessivo gasto em documentação manual, reduzindo a produtividade dos profissionais de saúde.
- Ausência de ferramentas inteligentes para transformar dados clínicos em informações úteis para médicos.

Lean Canvas



CANVAS

REQUISITOS FUNCIONAIS

Anotações SOAP

Exportação Prontuários

Login Seguro

Gestão de Médicos

Especialidade IA

Histórico de Versões

Edição de Prontuários

Comparação Diff

Colaboração Médica

Integração FHIR HL7

App Mobile Offline

Relatórios de Estatísticas

Recuperação de Senha

REQUISITOS NÃO FUNCIONAIS

Desempenho de 5s

Disponibilidade de 99%

Autenticação Segura

Escalabilidade Container

Arquitetura MVC

Interface Responsiva

Backup Restore

Suporte Multiplataforma

Conformidade LGPD

DIAGRAMA DE CASO DE USO

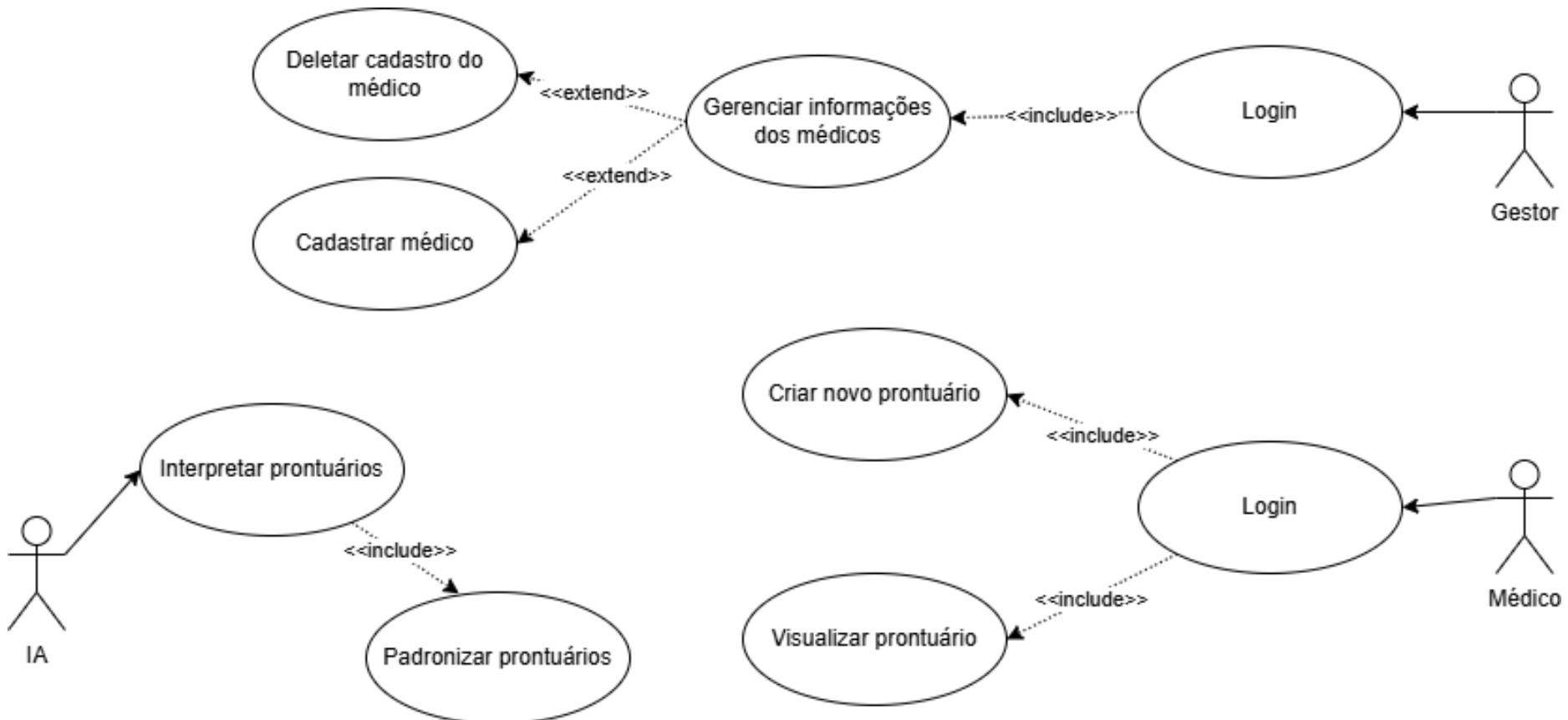
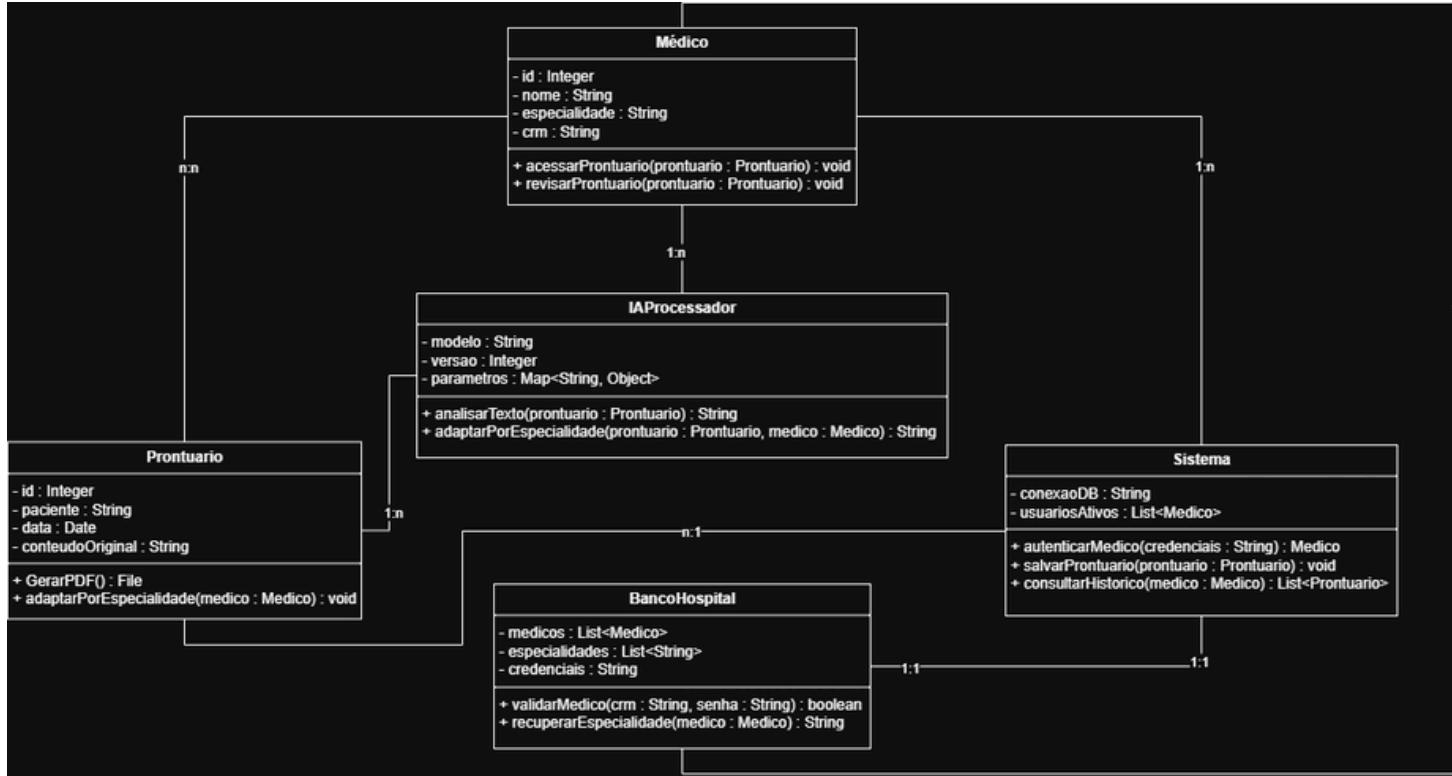
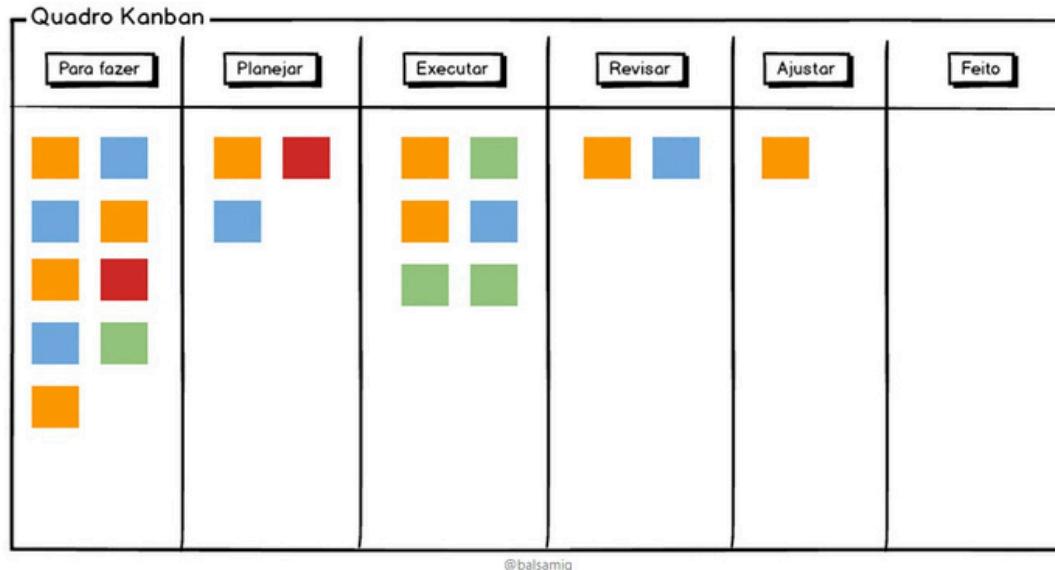


DIAGRAMA DE CLASSE



MÉTODO DE DESENVOLVIMENTO

Kanban



Scrum



PLATAFORMA TRELLO

Projeto_Integrador 000

EA MC BC FF GG +1 ⌂ Compartilhar ...

Sprint 1 - 18/08 - 08/09

- ✓ Reunião Inicial
- ✓ SMART
- ✓ EAP
- ✓ Canvas
- ✓ BPMN
- ✓ Caso de Uso
- ✓ Diagrama de Classe
- ✓ Treino da IA Generativa - Fase 1: Criação
- ✓ Documentação Parte 1
- 
- + Adicionar um cartão

Sprint 2 - 08/09 - 29/09

- ✓ Reunião Inicial
- ✓ WireFrame
- ✓ Desenvolvimento do Leitor de PDF/Arquivo
- ✓ Alterações no Design do Projeto
- ✓ Protótipo - Telas
- ✓ Desenvolvimento do FrontEnd
- ✓ Conexão com o Banco de Dados (MongoDB)
- ✓ Treino da IA Generativa - Fase 2: Implementação
- ✓ Treino da IA Generativa - Fase 2: Machine Learning
- ✓ EAP Atualizado
- ✓ Documentação Parte 2
- + Adicionar um cartão

Sprint 3 - 06/10 - 27/10

- ✓ Reunião Inicial
- Alterações no Design do Projeto - Figma e Front End
- ✓ WireFrame
- Documentação Parte 1 - Final
- Documentação Parte 2 - Final
- 
- Apresentação 13/10
- ✓ Reunião Final
- + Adicionar um cartão

Sprint 4 - 03/11 - 01/12

- Reunião Inicial
- Artigo Científico - Entrega
- Correções de falhas
- Apresentação Final
- Versão final do projeto
- + Adicionar um cartão

+ Adicionar outra lista

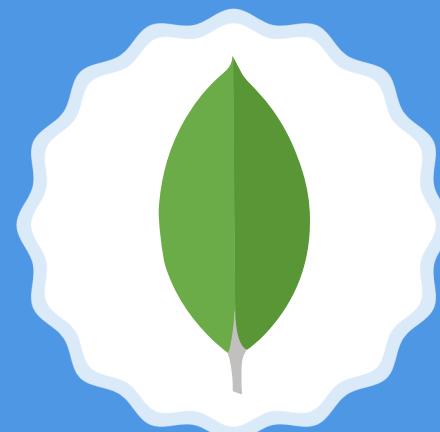
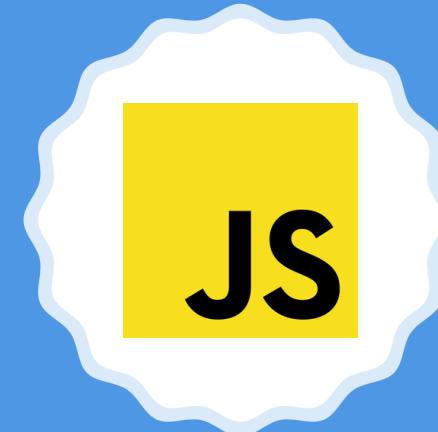
+ Adicionar um cartão

Caixa de entrada Planejador Quadro Mudar de quadros

Jira



TECNOLOGIAS UTILIZADAS



The screenshot shows a code editor interface with a dark theme. On the left, there's a vertical toolbar with various icons: a file icon, a search icon, a refresh icon, a copy/paste icon, a magnifying glass, a gear, and a question mark. The main area displays a Python script named `saudade_mais.py`. The code implements an OAuth2 client to get an access token from a StackSpot identity provider and then uses it to make a POST request to a chat endpoint.

```
saude_mais.py x
src > main > resources > ia-integration > saude_mais.py
18     REALM = "stackspot-freemium"
19
20     CLIENT_ID = "f56b6b65-6488-4401-9a94-c4c211654497"
21     CLIENT_KEY = "y5H1vve2jo2ci23HdzhafYpYni04143BBY12MemAER5LqR5rdfy57ZA5g76CF93o"
22
23     AGENT_URL = "https://genai-inference-app.stackspot.com/v1/agent/01K3GT29GMFY5HMCEM7F8HSNKN/chat"
24
25
26     def get_jwt():
27
28         url = f"https://idm.stackspot.com/{REALM}/oidc/oauth/token"
29
30         payload = {
31             "grant_type": "client_credentials",
32             "client_id": CLIENT_ID,
33             "client_secret": CLIENT_KEY
34         }
35
36         headers = {"Content-Type": "application/x-www-form-urlencoded"}
37
38         response = requests.post(url, data=payload, headers=headers)
39
40         if response.status_code != 200:
41
42             raise HTTPException(status_code=401, detail="Erro na autenticação")
43
44         return response.json().get("access_token")
45
46
47     @app.post("/chat")
48
49     async def chat(request: Request):
50
51         import datetime
52
53         body raw = await request.body()
54
55         response = {"status": "ok", "message": "Chat received at " + str(datetime.datetime.now())}
```

App controller (model)

The screenshot shows a Java code editor with the file `appController.java` open. The code defines a controller class with three methods: `home()`, `login()`, and `gestor()`. The `home()` and `login()` methods map to the root URL and the login URL respectively. The `gestor()` method maps to the URL `/gestor`. The code uses annotations `@Controller` and `@RequestMapping` from the `org.springframework.web.bind.annotation` package.

```
src > main > java > com > br > iasaude > saudemais > Controller > appController.java > gestor()

8 @Controller
9
10 public class appController {
11     // Rota para a página inicial
12     @GetMapping("/")
13     public String home() {
14         return "index";
15     }
16
17     // Rota para a página de login
18     @GetMapping("/login")
19     public String login() {
20         return "login";
21     }
22
23     // Rota para a página do gestor
24     @GetMapping("/gestor")
25     public String gestor() {
26         return "gestor";
27     }
28 }
```

Medicorest controller

The screenshot shows a Java code editor interface with a dark theme. On the left, there's a vertical toolbar with icons for file operations, search, and navigation. The main area displays a Java file named `MedicoRestController.java`. The code implements a REST controller for a medical application:

```
1 package com.br.iasaude.saudemais.Controller;
2
3 import org.springframework.web.bind.annotation.GetMapping;
4 import org.springframework.web.bind.annotation.RequestMapping;
5 import org.springframework.web.bind.annotation.RestController;
6 import org.springframework.security.access.prepost.PreAuthorize;
7
8 @RestController
9 @RequestMapping("/api")
10 public class MedicoRestController {
11     @GetMapping("/medico")
12     public String medico() {
13         return "Acesso autorizado ao endpoint REST /api/medico";
14     }
15 }
16
```

The code uses Spring annotations to map URLs to methods and apply security checks. The bottom status bar indicates the file is ready for Java compilation.

The screenshot shows a Java IDE interface with the following details:

- Title Bar:** MedicoRestController.java 1 < PdfController.java <
- Project Structure:** src > main > java > com > br > iasaude > saudemais > Controller > PdfController.java > {} com.br.iasaude.saudemais.Controller
- Code Editor:** The PdfController.java file is open. The code handles PDF interpretation and sends requests to an AI API.

```
24  public class PdfController {  
30      public ResponseEntity<Map<String, Object>> interpretarPdf(@RequestParam("file") MultipartFile file) throws IOException {  
46          // Se não houver texto extraído, retorna erro  
47          if (text == null || text.trim().isEmpty()) {  
48              // (Removido bloco duplicado e corrigido)  
49              logger.warn("Texto extraído do PDF está vazio ou nulo. Não será enviada requisição para IA.");  
50              result.put(key: "erroIA", value: "Texto extraído do PDF está vazio ou nulo.");  
51              return ResponseEntity.ok(result);  
52          }  
53          // Chamada para a API da IA (FastAPI) usando RestTemplate  
54          String iaResponse = null;  
55          try {  
56              ObjectMapper mapper = new ObjectMapper();  
57              Map<String, String> iaPayload = new HashMap<>();  
58              iaPayload.put(key: "user_prompt", text);  
59              String requestBody = mapper.writeValueAsString(iaPayload);  
60              logger.info("Enviando para IA: {}", requestBody);  
61  
62              HttpHeaders headers = new HttpHeaders();  
63              headers.setContentType(MediaType.APPLICATION_JSON);  
64              HttpEntity<String> entity = new HttpEntity<>(requestBody, headers);
```

- Toolbars and Status Bar:** Includes icons for file operations, search, and navigation. The status bar at the bottom shows: main, 0, 3, Java: Ready, edutbx (1 semana atrás), Ln 2, Col 35, Espaços: 4, UTF-8, CRLF, Java, Go Live, and a refresh icon.

Profile controller

The screenshot shows a Java IDE interface with the following details:

- Title Bar:** The title bar displays "Profile controller".
- File List:** The left sidebar shows files: "MedicoRestController.java 1" and "ProfileController.java X".
- Code Editor:** The main area contains the code for "ProfileController.java".
- Code Content:** The code defines a REST controller for profiles. It imports necessary packages, defines a constructor with an Environment parameter, and implements a method to get a profile map.

```
src > main > java > com > br > iasaude > saudemais > Controller > ProfileController.java > {} com.br.iasaude.saudemais.Controller  
4 import org.springframework.web.bind.annotation.GetMapping;  
5 import org.springframework.web.bind.annotation.RestController;  
6 import java.util.Arrays;  
7 import java.util.Map;  
8  
9 @RestController  
10 public class ProfileController {  
11     private final Environment env;  
12  
13     public ProfileController(Environment env) {  
14         this.env = env;  
15     }  
16  
17     @GetMapping("/api/profile")  
18     public Map<String, Object> getProfile() {  
19         String[] profiles = env.getActiveProfiles();  
20         return Map.of(  
21             k1: "activeProfiles", profiles,  
22             k2: "dev", Arrays.asList(profiles).contains(o: "dev")  
23         );  
24     }  
25 }
```

- Bottom Status Bar:** The status bar includes icons for file operations, Java readiness, and a Go Live button.
- Bottom Right:** A footer bar with icons for search, file operations, and Java/CRLF settings.

OBRIGADO

