

**SMART
LEAF**

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



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

ArquiteturaMVC

Interface Responsiva

Backup Restore

SuporteMultiplataforma

Conformidade LGPD

DIAGRAMA DE CASO DE USO

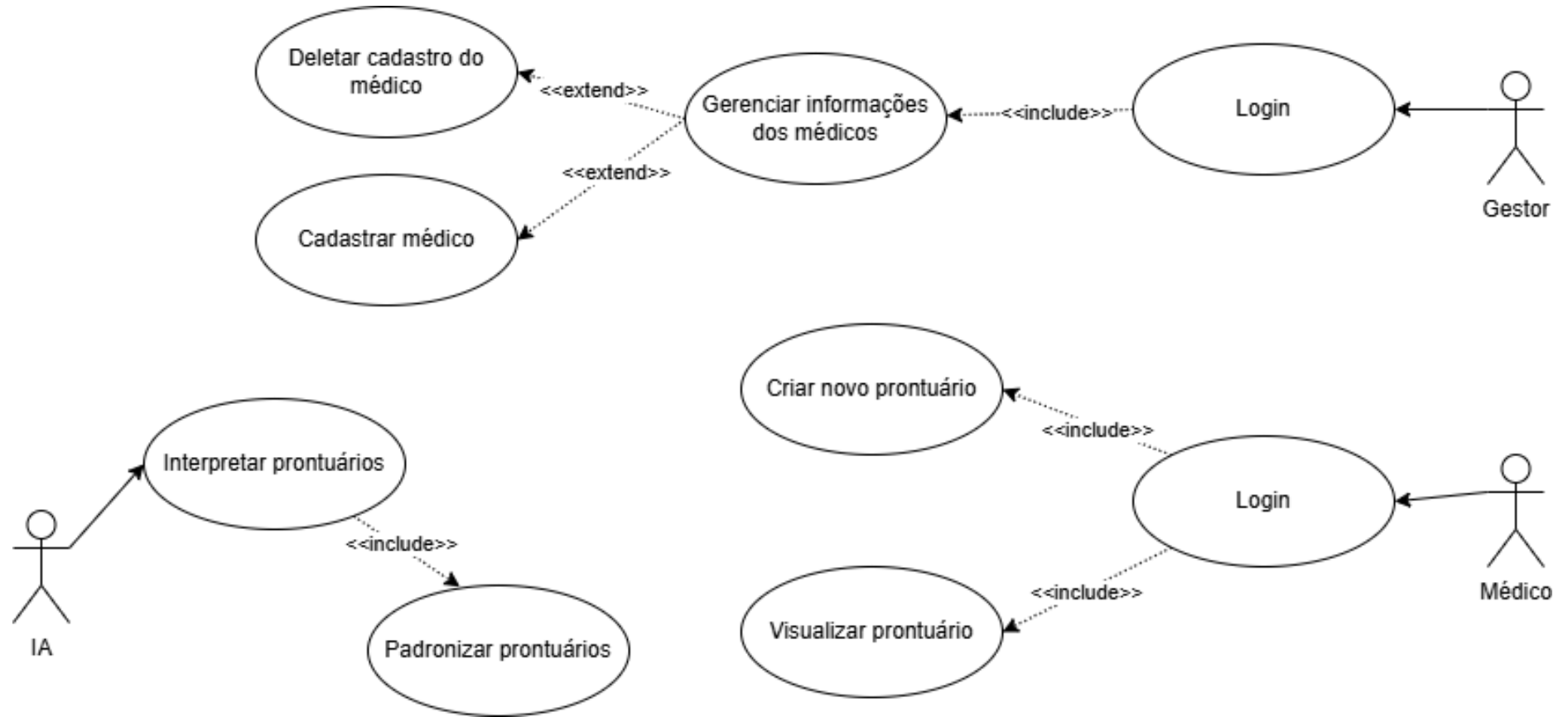
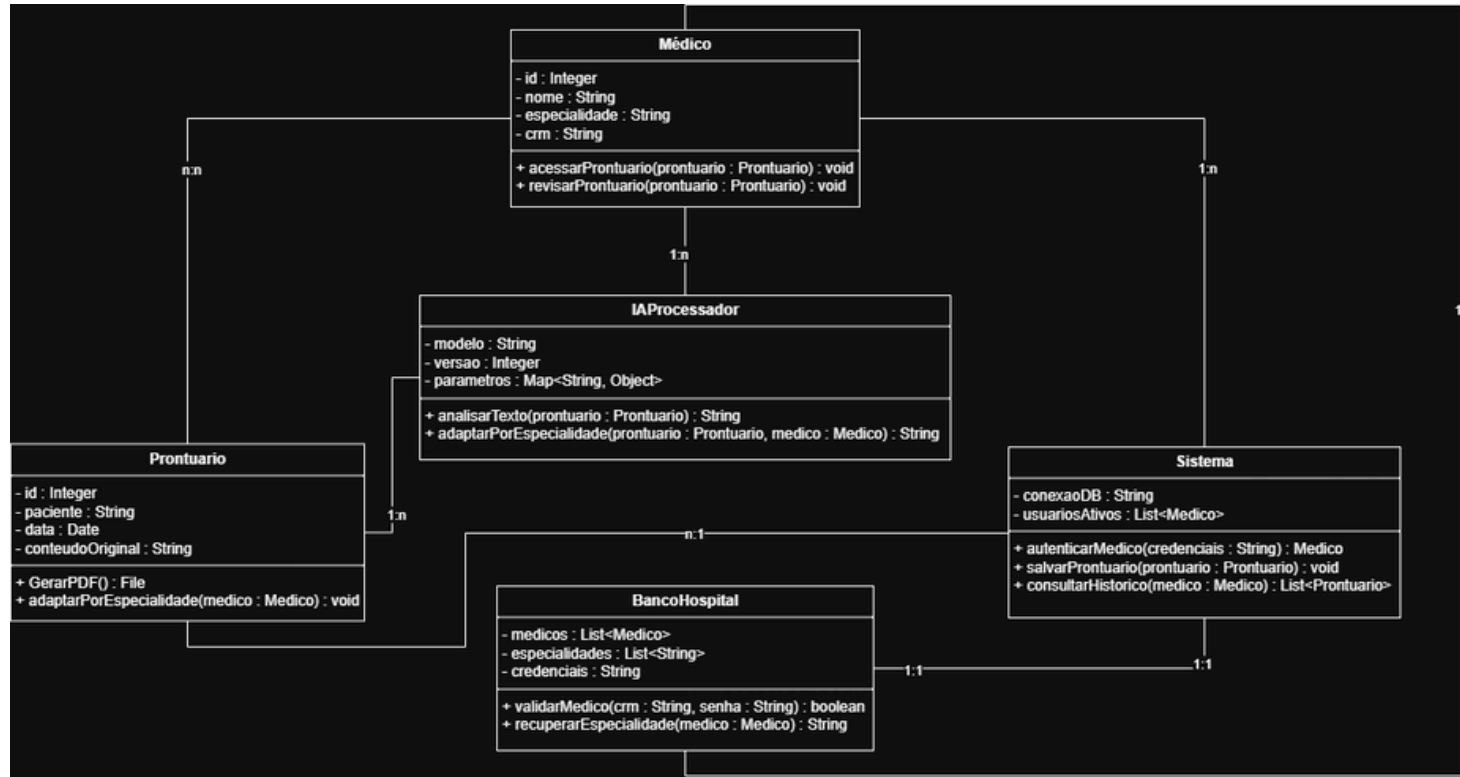


DIAGRAMA DE CLASSE



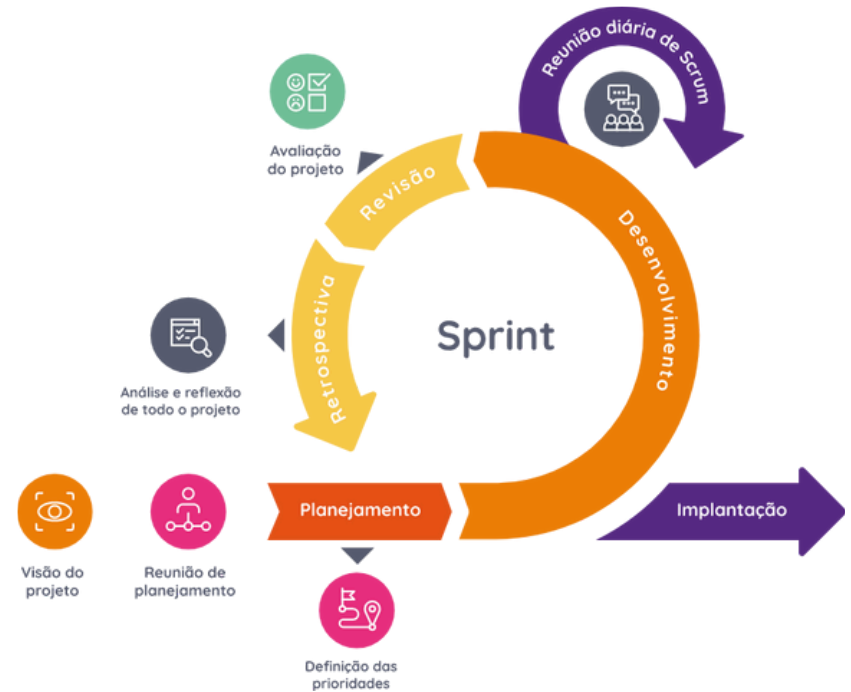
MÉTODO DE DESENVOLVIMENTO

Kanban



@balsamiq

Scrum



PLATAFORMA TRELLO

Projeto_Integrador

EA MC BC FF GG +1

+ Compartilhar

+ Adicionar outra lista

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

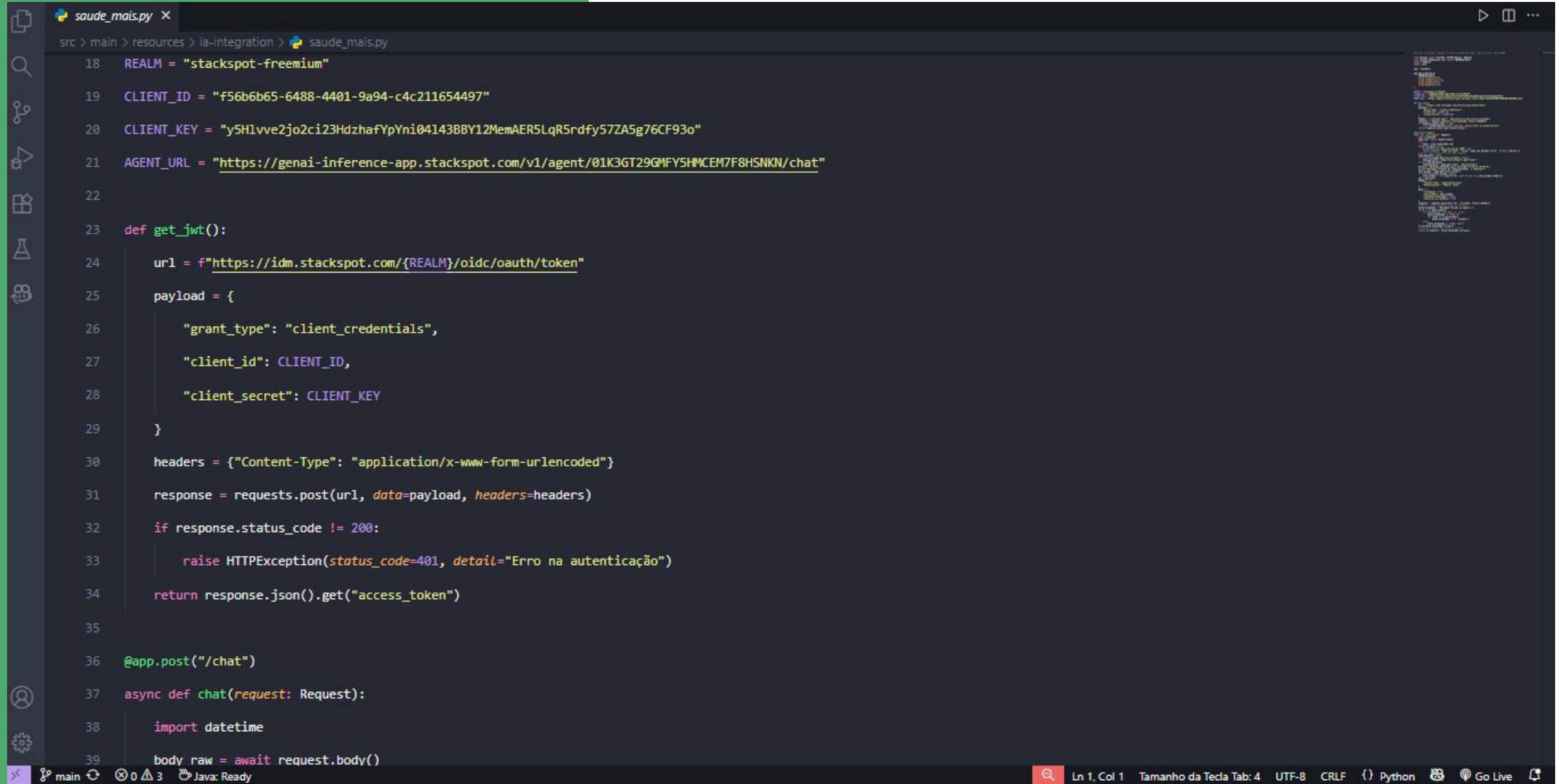
- Reunião Inicial
- Artigo Científico - Entrega
- Correções de falhas
- Apresentação Final
- Versão final do projeto
- + Adicionar um cartão

Caixa de entrada | Planejador | Quadro | Mudar de quadros

Jira

TECNOLOGIAS UTILIZADAS



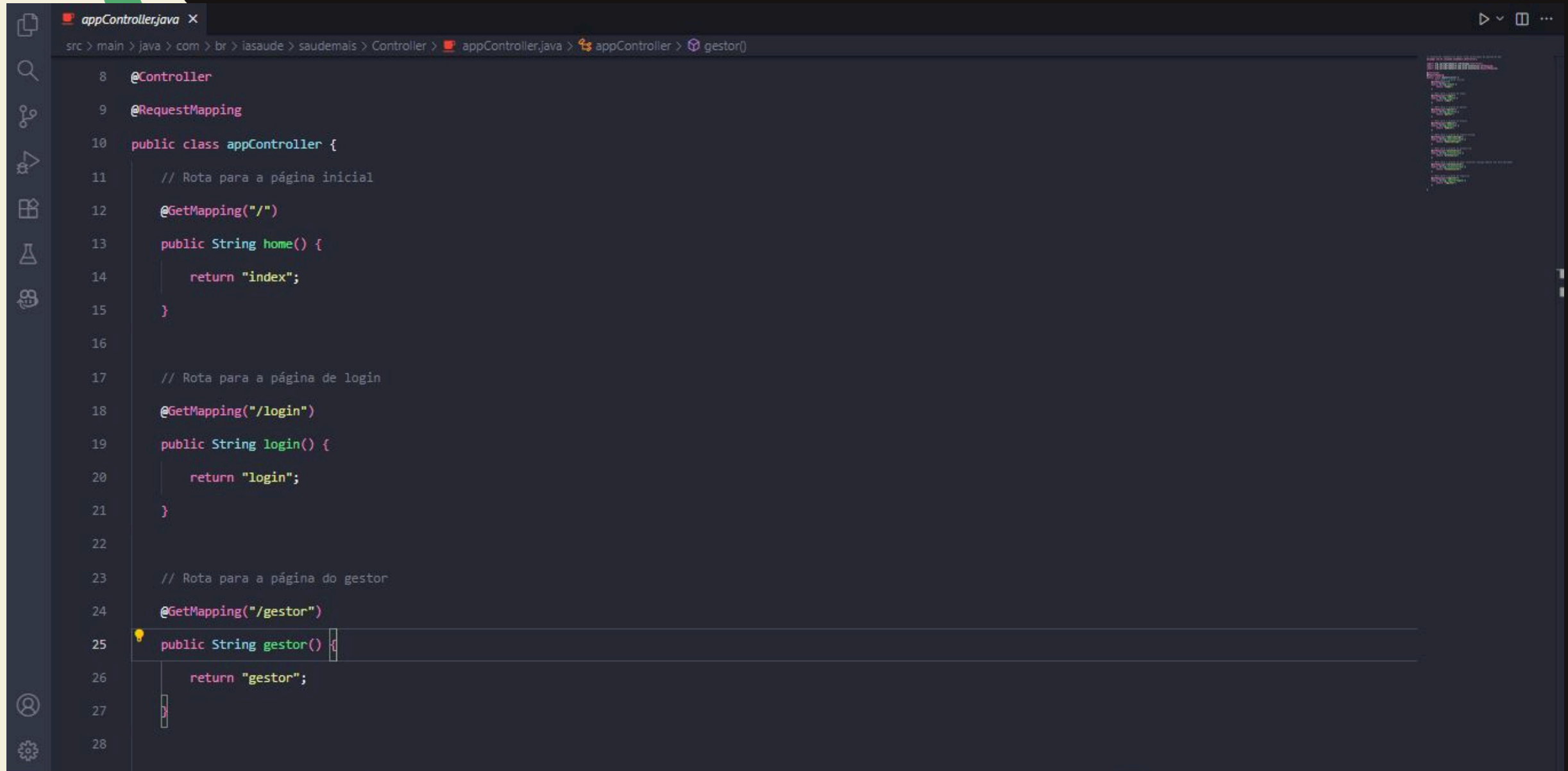


```
18 REALM = "stackspot-freemium"
19 CLIENT_ID = "f56b6b65-6488-4401-9a94-c4c211654497"
20 CLIENT_KEY = "y5Hlvve2jo2ci23HdzhafYpYni04143BBY12MemAER5LqR5rdfy57ZA5g76CF93o"
21 AGENT_URL = "https://genai-inference-app.stackspot.com/v1/agent/01K3GT29GMFY5HMC7F8HSNKN/chat"
22
23 def get_jwt():
24     url = f"https://idm.stackspot.com/{REALM}/oidc/oauth/token"
25     payload = {
26         "grant_type": "client_credentials",
27         "client_id": CLIENT_ID,
28         "client_secret": CLIENT_KEY
29     }
30     headers = {"Content-Type": "application/x-www-form-urlencoded"}
31     response = requests.post(url, data=payload, headers=headers)
32     if response.status_code != 200:
33         raise HTTPException(status_code=401, detail="Erro na autenticação")
34     return response.json().get("access_token")
35
36 @app.post("/chat")
37 async def chat(request: Request):
38     import datetime
39     body_raw = await request.body()
```

Ln 1, Col 1 Tamanho da Tecla Tab: 4 UTF-8 CRLF Python Go Live

Appcontroller

(model)



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

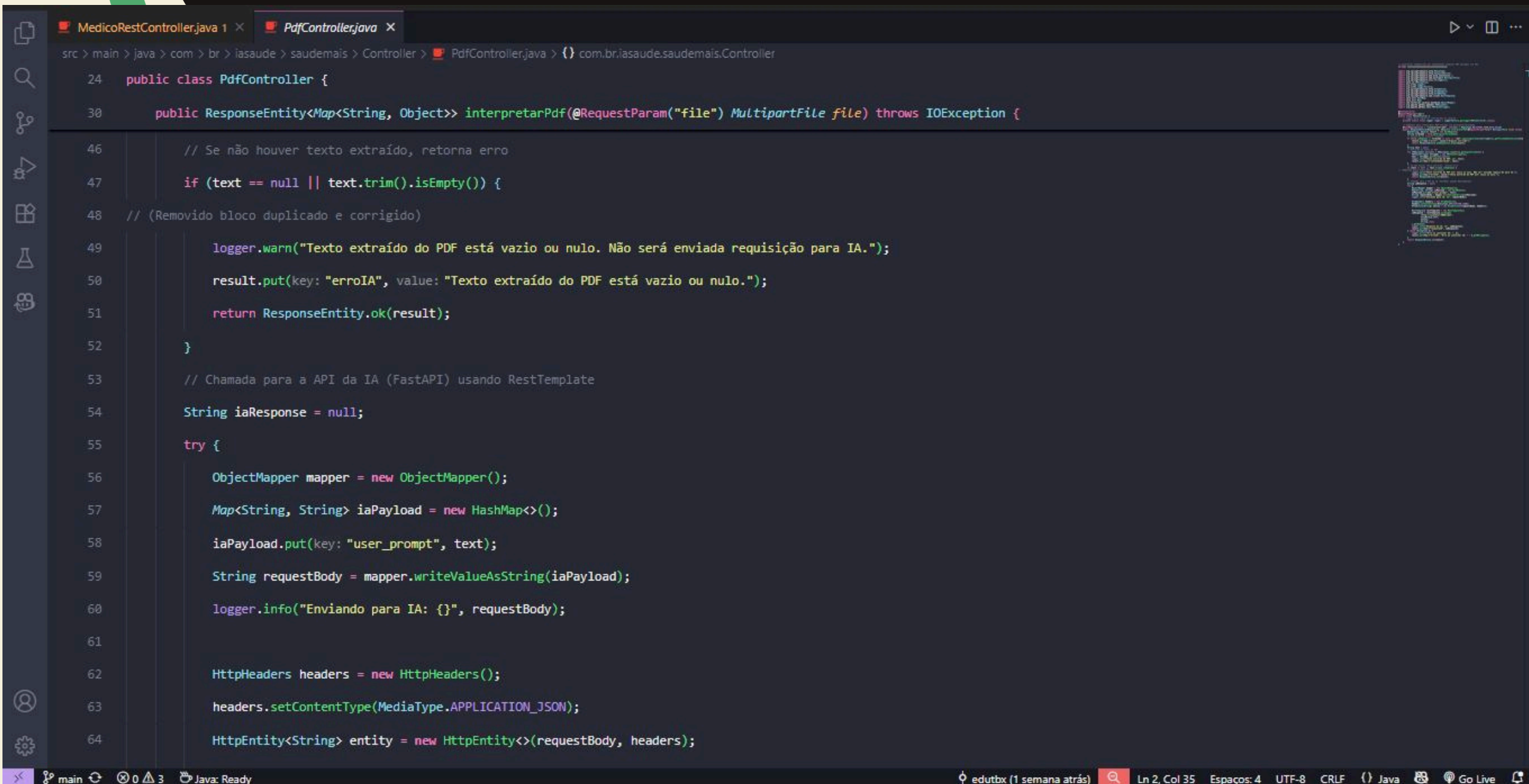
8  @Controller
9  @RequestMapping
10 public class appController {
11
12     // Rota para a página inicial
13     @GetMapping("/")
14     public String home() {
15         return "index";
16     }
17
18     // Rota para a página de login
19     @GetMapping("/login")
20     public String login() {
21         return "login";
22     }
23
24     // Rota para a página do gestor
25     @GetMapping("/gestor")
26     public String gestor() {
27         return "gestor";
28     }
29 }
```

MedicoRestController

MedicoRestController.java 1 X

src > main > java > com > br > iasaude > saudemais > Controller > MedicoRestController.java > ...

```
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
```

```
src > main > java > com > br > iasaude > saudemais > Controller > PdfController.java > {} com.br.iasaude.saudemais.Controller

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);
```

main 0 3 Java: Ready edutbx (1 semana atrás) Ln 2, Col 35 Espaços: 4 UTF-8 CRLF {} Java Go Live

Profilecontroller

```
MedicoRestController.java  ProfileController.java x
src > main > java > com > br > iasaude > saudemais > Controller > ProfileController.java > {} com.br.iasaude.saudemais.Controller
+ 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 }
```

OBRIGADO