Relatório de Produção Individual - Hugo Alves dos Santos (Parte 2)

Período de Referência: Foco na elaboração e implementação de Testes Unitários para os módulos de Cadastro e Agendamento do back-end.

---

1. Objetivo da Atividade

A principal atividade desta etapa foi a elaboração de uma suíte completa de testes unitários para os microsserviços de **Cadastro** e **Agendamento** do sistema SimpleHealth. O objetivo é garantir a robustez, a funcionalidade e a aderência a requisitos de segurança e de negócio em todas as camadas da aplicação (Domain, Application e Web).

2. Módulo de Cadastro (SimpleHealth Cadastro)

Foi desenvolvida uma cobertura abrangente com **123 testes unitários** no total, focando na camada de back-end.

Camada	Escopo de Teste	Detalhes da Cobertura
Entidades de Domínio	Classes: `Pessoa`, `Paciente`, `Medico`, `Usuario`, `Convenio` e `EPerfilUsuario` (Enum)	Validação da lógica de domínio e comportamento das entidades.
Serviços (Business)	`PacienteService`, `MedicoService`, `ConvenioService`, `UsuarioService`	Testes completos de CRUD (Create, Read, Update, Delete). Cobertura de validações de negócio, como CPF duplicado e CRM duplicado. Teste de criptografia de senha e validação de login duplicado.
Use Cases	`CadastrarNovoPacienteUseCase`	Cobertura de cenários de sucesso e falha (ex.: CPF já cadastrado).

Controllers (API)	PacienteController, MedicoController, ConvenioController	Teste de todos os endpoints (POST, GET, PUT, DELETE) com simulação de autenticação (@WithMockUser) e validação de proteção CSRF, utilizando MockMvc.
Mappers e DTOs	EntityDtoMapper, DTOs de todas as entidades	Teste de conversão bidirecional entre entidades e DTOs, incluindo tratamento de valores nulos.

Tecnologias de Teste Utilizadas: JUnit 5, Mockito, Spring MockMvc, Spring Security Test e AssertJ.

3. Módulo de Agendamento (SimpleHealth Agendamento)

Foi desenvolvida uma cobertura abrangente com **198 testes unitários** no total, abrangendo 100% dos principais componentes.

Camada	Escopo de Teste	Detalhes da Cobertura
Web/Controllers	AgendamentoController, BloqueioAgendaController, EncaixeController	Cobertura completa de todos os controllers.
Services e Use Cases	AgendamentoService, ConsultaService, Casos de Uso (ex.: AgendarConsultaUseCase, CancelarAgendamentoUseCase, SolicitarEncaixeUseCase)	Lógica de negócio e fluxo de trabalho de ponta a ponta dos casos de uso.

Domain Entities	Agendamento, Consulta, BloqueioAgenda, Exame, Procedimento	100% das entidades de domínio testadas.
Infraestrutura	Componentes Redis (RedisEventPublisher, AgendamentoSubscriber)	Teste da comunicação e publicação/subscrição de eventos de domínio.
Exceções e Enums	Todas as exceções customizadas e enums (StatusAgendamentoEnum, ModalidadeEnum, TipoConsultaEnum)	Cobertura completa.

Boas Práticas de Teste Aplicadas:

- Adoção dos padrões Arrange-Act-Assert (AAA) e Given-When-Then.
- Alto nível de isolamento de testes, utilizando Mocking de dependências.
- Testes de integração para controllers com MockMVC.

4. Evidência de Commits

Os seguintes commits do Git contêm as implementações dos testes unitários para os módulos:

- Testes do Módulo de Cadastro: https://github.com/ps-es-2025-2/grupo4/commit/5cfdc357c39cb99269ca7fbbc1fe0327f4d47c9b
- Testes do Módulo de Agendamento: https://github.com/ps-es-2025-2/grupo4/commit/91d3aabf6ea2c793072b6f7832087d333a68b1da

---

Observação: O foco na cobertura de testes para os módulos de Cadastro e Agendamento reforça o compromisso com a qualidade do back-end, garantindo que as principais lógicas de negócio e as camadas da API estejam funcionando conforme especificado antes da entrega final, os testes unitários do módulo de estoque ainda não foram implementados pois não está pronto