

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

PUC Minas Virtual

Pós-graduação *Lato Sensu* em Arquitetura de *Software* Distribuído

Projeto Integrado

Relatório Técnico

Inside Nutri

Michael Tadeu Alves de Oliveira

Belo Horizonte
<mês e ano>.

Projeto Integrado – Arquitetura de Software Distribuído

Sumário

Projeto Integrado – Arquitetura de Software Distribuído	2
1. Introdução	4
2. Cronograma do Trabalho	6
3. Especificação Arquitetural da solução	8
3.1 Restrições Arquiteturais	8
3.2 Requisitos Funcionais	8
3.3 Requisitos Não-funcionais	10
3.4 Mecanismos Arquiteturais	11
4. Modelagem Arquitetural	13
4.1 Diagrama de Contexto	13
4.2 Diagrama de Container	14
4.3 Diagrama de Componentes	16
5. Prova de Conceito (PoC)	20
5.1 Integrações entre Componentes	21
5.2 Código da Aplicação	25
Etapa 3 - Pendente	29
Referências	30

Lista de Figuras

Figura 1 - Visão Geral da Solução Inside Nutri. Fonte: Do Autor.....	13
Figura 2 – Diagrama de Container. Fonte: Do Autor.	15
Figura 3 – Diagrama de Componentes. Fonte: Do Autor.	17
Figura 4 - Tela de Cadastro de Usuários. Fonte: Do Autor.	21
Figura 5 - Tela de Login. Fonte: Do Autor.....	22
Figura 6 - Tela Inicial (Dashboard). Fonte: Do Autor.	23
Figura 7 - Tela de Cadastro de Nova Dieta do Usuário. Fonte: Do Autor.....	24
Figura 8 - Tela de Realizar Cálculos de Dieta. Fonte: Do Autor.....	25
Figura 9 - Código do fluxo para realizar Cálculos. Fonte: Do Autor.....	26
Figura 10 - Código do fluxo para realizar Cadastro de Pacientes. Fonte: Do Autor.....	27
Figura 11- Código do fluxo para realizar Cadastro de Dietas. Fonte: Do Autor.....	28

1. Introdução

Uma das maiores influências negativas sobre as taxas de morbidade e mortalidade por diferentes causas em pacientes internados é o comprometimento do estado nutricional (LOPES et al.; 2009). A desnutrição eleva a chance de o paciente apresentar complicações durante o período de internação hospitalar e ainda pode determinar um maior tempo de recuperação e reabilitação e, conseqüentemente, a elevação do custo do sistema de saúde, além da piora da qualidade de vida do paciente (OLIVEIRA et al.; 2010).

A desnutrição pode acometer rapidamente o doente hospitalizado principalmente devido ao estado hipercatabólico que acompanha as patologias, a presença de traumatismos e infecções em resposta ao estresse metabólico que ocorre nestas condições, em especial quando a ingestão nutricional é insuficiente (MINISTÉRIO DA SAÚDE, 2016).

O Inquérito Brasileiro de Avaliação Nutricional - Ibranutri, realizado em pacientes internados em hospitais da rede pública, observou uma prevalência de desnutrição em 48,1% dos 4 mil pacientes internados e submetidos à avaliação nutricional. E em uma amostra de 709 pacientes, a incidência de complicações nos desnutridos foi de 27,0%, ocasionando um aumento nos custos hospitalares de 60,5% para desnutridos e 3 vezes mais mortalidade nesse grupo quando comparados aos pacientes bem nutridos (WAITZBERG; CAIAFFA; CORREIA, 1999).

Mesmo com os avanços em terapia nutricional nas últimas décadas, a desnutrição continua sendo frequente em pacientes hospitalizados, com prevalência variando entre 30 e 65% (LEITE; CARVALHO; MENESES, 2005). Desta forma, o cuidado nutricional adequado, possui efeitos benéficos na recuperação dos pacientes e melhora da sua qualidade de vida (GARCIA; PADILHA; SANCHES, 2012).

Para que quadros de desnutrição hospitalar sejam raros, os pacientes hospitalizados necessitam de uma ingestão nutricional adequada que forneça as necessidades básicas de manutenção do organismo e de recuperação da saúde. Assim, o papel do nutricionista clínico inclui realizar o atendimento dietoterápico com base no quadro clínico do paciente, avaliar fatores de risco relacionados, analisar o estado nutricional através de medidas antropométricas e exames laboratoriais para elaborar objetivos e metas para o cuidado nutricional, utilizando fórmulas matemáticas para o cálculo adequado de calorias necessárias, de acordo com as diretrizes mais atuais para cada patologia (CFN, 2018).

Logo, para que o trabalho do nutricionista clínico hospitalar seja otimizado, reduzindo a probabilidade de erros nos cálculos, o objetivo deste trabalho é apresentar a descrição do projeto arquitetural da aplicação Inside Nutri para realizar cálculos precisos e relevantes para os profissionais, sendo a partir da entrada de dados específicos, o valor calórico total diário para o paciente de acordo com a patologia, a quantidade de proteína, o volume total de dieta enteral a ser administrada e velocidade de infusão, fazendo com que a Terapia Nutricional seja efetiva, sendo uma ferramenta de auxílio para a redução da desnutrição hospitalar. Para alcançar o objetivo proposto, os seguintes objetivos específicos são necessários:

- Realizar levantamento na literatura das fórmulas matemáticas para os cálculos necessários;
- Realizar levantamento das recomendações nutricionais para cada patologia;
- Mapeamento das dietas enterais para cada patologia.

2. Cronograma do Trabalho

A seguir é apresentado o cronograma proposto para as etapas deste trabalho.

Datas		Atividade / Tarefa	Produto / Resultado
De	Até		
19 / 06 / 2022	19 / 06 / 2022	1. Cronograma de Trabalho	Tabela com o cronograma de trabalho
20 / 06 / 2022	22 / 06 / 2022	2. Brainstorming	Proposta do projeto a ser desenvolvido
23 / 06 / 2022	25 / 06 / 2022	3. Contextualização do Trabalho	Construção da Introdução deste trabalho, contendo o problema, motivação e objetivos
26 / 06 / 2022	30 / 06 / 2022	4. Elicitação das restrições Arquiteturais	Lista com as restrições arquiteturais identificadas
01 / 07 / 2022	05 / 07 / 2022	5. Elicitação dos Requisitos Funcionais	Lista com os requisitos funcionais identificados
06 / 07 / 2022	10 / 07 / 2022	6. Elicitação dos Requisitos Não-Funcionais	Lista com os requisitos não-funcionais identificados
11 / 07 / 2022	15 / 07 / 2022	7. Elicitação dos Mecanismos Arquiteturais	Lista com os mecanismos arquiteturais identificados
16 / 07 / 2022	20 / 07 / 2022	8. Elaboração do Diagrama de Contexto – Modelo C4	Diagrama de contexto do projeto proposto
21 / 07 / 2022	28 / 07 / 2022	9. Revisão do documento da Etapa 1	Documento revisado da Etapa 1 do projeto proposto
29 / 07 / 2022	05 / 08 / 2022	10. Elaborar apresentação da Etapa 1	Apresentação do projeto proposto
06 / 08 / 2022	10 / 08 / 2022	11. Gravar vídeo da Etapa 1	Vídeo gravado da Etapa 1 do projeto proposto
11 / 08 / 2022	14 / 08 / 2022	12. Publicar Etapa 1 no repositório do GitHub	Arquivos da etapa 1 disponibilizados no repositório do GitHub
15 / 08 / 2022	15 / 08 / 2022	13. Entrega da Etapa 1 no canvas da disciplina Projeto Aplicado	Etapa 1 concluída e enviada para avaliação
16 / 08 / 2022	29 / 08 / 2022	14. Construção do Diagrama de Container	Diagrama de container do projeto proposto
30 / 08 / 2022	10 / 09 / 2022	15. Construção do Diagrama de Componentes	Diagrama de componentes do projeto proposto
11 / 09 / 2022	23 / 09 / 2022	16. Desenho do Wireframe da POC	Protótipo de telas do projeto proposto
24 / 09 / 2022	04 / 10 / 2022	17. Código da aplicação	Aplicação com três requisitos implementados

Projeto Integrado – Engenharia de *Software* - PMV

Datas		Atividade / Tarefa	Produto / Resultado
De	Até		
05 / 10 / 2022	12 / 10 / 2022	18. Revisão do documento da Etapa 2	Documento revisado da Etapa 2 do projeto proposto
13 / 10 / 2022	14 / 10 / 2022	19. Publicar Etapa 2 no repositório do GitHub	Arquivos da etapa 2 disponibilizados no repositório do GitHub
15 / 10 / 2022	15 / 10 / 2022	20. Entrega da Etapa 2 no canvas da disciplina Projeto Aplicado	Etapa 2 concluída e enviada para avaliação
16 / 10 / 2022	29 / 10 / 2022	21. Análise das abordagens arquiteturais	Seção do documento produzido
30 / 10 / 2022	10 / 11 / 2022	22. Construção dos Cenários	Seção do documento produzido
11 / 11 / 2022	23 / 11 / 2022	23. Evidências da avaliação	Seção do documento produzido
24 / 11 / 2022	04 / 12 / 2022	24. Resultados obtidos	Seção do documento produzido
05 / 12 / 2022	10 / 12 / 2022	25. Avaliação crítica dos resultados	Seção do documento produzido
11 / 12 / 2022	13 / 12 / 2022	26. Conclusão	
14 / 12 / 2022	14 / 12 / 2022	27. Revisão do documento da Etapa 3	Documento revisado da Etapa 3 do projeto proposto
15 / 12 / 2022	15 / 12 / 2022	28. Produção do vídeo 2 na Etapa 3, apresentando o trabalho de forma completa	Vídeo gravado da Etapa 3 do projeto proposto
15 / 12 / 2022	15 / 12 / 2022	29. Publicar Etapa 2 no repositório do GitHub	Arquivos da etapa 3 disponibilizados no repositório do GitHub
15 / 12 / 2022	15 / 12 / 2022	30. Entrega da Etapa 2 no canvas da disciplina Projeto Aplicado	Etapa 3 concluída e enviada para avaliação

3. Especificação Arquitetural da solução

Esta seção apresenta a especificação básica da arquitetura da solução a ser desenvolvida, incluindo diagramas, restrições e requisitos definidos pelo autor, tal que permitem visualizar a macroarquitetura da solução.

3.1 Restrições Arquiteturais

A qualidade do projeto arquitetural e das descrições arquiteturais são fortemente influenciadas por restrições arquiteturais. Seu impacto na qualidade da arquitetura engloba propriedades relativas à qualidade do sistema de software em construção, bem como a processo de design, incluindo sua eficiência e suas perspectivas. Assim, para esse projeto foi proposto restrições arquiteturais que devem ser satisfeitas, sendo:

ID	Descrição
RA01	Deve ser usado tecnologias <i>Open source</i> para o desenvolvimento de toda a aplicação Inside Nutri.
RA02	Deve ser usado o serviço de nuvem da <i>Amazon Web Services</i> (AWS) ou <i>Microsoft Azure</i> como provedora da infraestrutura necessária para a aplicação Inside Nutri.
RA03	Deve ser usado o serviço <i>OAuth 2.0</i> do Google bem como a possibilidade de criação de conta diretamente na aplicação Inside Nutri, para o gerenciamento de autenticação dos usuários.
RA04	Deve ser implementado a aplicação Inside Nutri utilizando uma solução móvel (<i>Apps Mobile</i>) que deverá suportar os sistemas operacionais móveis mais populares, a saber Android e IOS.
RA05	Deve ser implementado uma <i>API RESTful</i> para prover todos os dados e comunicação com os clientes de forma agnóstica e desacoplada com o backend, facilitando uma possível mudança tecnológica no frontend e a utilização de microserviços na aplicação Inside Nutri.
RA06	Implementar <i>RabbitMQ</i> como <i>event broker</i> , sendo necessário a comunicação orientada a eventos na aplicação Inside Nutri.
RA07	A comunicação na aplicação Inside Nutri de origem externa ao ambiente deve passar por um <i>API Gateway</i> .
RA08	O registro de log's deve ser feito utilizando a ferramenta slack ELK para CI/ CD da aplicação Inside Nutri.

3.2 Requisitos Funcionais

Os requisitos de um sistema de software correspondem às descrições do que esse sistema deve fazer, aos serviços que ele oferece e às restrições em sua funcionalidade. Tais requisitos representam a demanda dos clientes desse sistema (SOMMERVILLE, 2019; CHITCHYAN et al., 2005). Em geral, os requisitos são classificados como (SOMMERVILLE, 2019): i) Requisitos Funcionais (RF) e (ii) Requisitos Não-Funcionais.

Os Requisitos Funcionais descrevem funções que o sistema deve fornecer, como ele deve reagir às entradas específicas e como ele deve se comportar em determinadas situações. A seguir, são apresentados os Requisitos Funcionais da aplicação Inside Nutri.

ID	Descrição Resumida	Dificuldade (B/M/A)*	Prioridade (B/M/A)*
RF01	A aplicação Inside Nutri deve permitir o auto cadastramento do usuário nutricionista.	B	A
RF02	A aplicação Inside Nutri deve permitir o auto cadastramento do usuário nutricionista através de uma conta do Google.	B	A
RF03	A aplicação Inside Nutri deve atender as orientações estabelecidas pela LGPD.	A	A
RF04	A aplicação Inside Nutri deve listar as funcionalidades na tela principal.	B	A
RF05	A aplicação Inside Nutri deve permitir que o nutricionista possa realizar uma avaliação nutricional.	M	A
RF06	A aplicação Inside Nutri deve permitir que o nutricionista calcule as seguintes métricas nutricionais, sendo Índice de Massa Corporal, Peso Ideal, Peso Ajustado, Peso Corrigido, Peso Estimado, Adequação de Peso, Altura Estimada e Gasto Energético Total	M	A
RF07	A aplicação Inside Nutri deve permitir que a partir dos dados obtidos no RF06 o nutricionista possa calcular o volume total de dieta enteral, hidratação e velocidade de infusão (Bomba de Infusão Contínua ou Infusão Intermitente) para cada patologia.	M	A
RF08	A aplicação Inside Nutri deve permitir que o nutricionista gere um relatório com todas as informações obtidas através dos cálculos.	M	M
RF09	A aplicação Inside Nutri deve permitir o nutricionista cadastrar a dieta enteral utilizada pela unidade de hospitalar, caso não encontre cadastrada na aplicação.	A	A
RF10	A aplicação Inside Nutri deve permitir a importação de dados de dietas enterais de fornecedores regionais.	M	M
RF11	A aplicação Inside Nutri deve permitir que o nutricionista cadastre o paciente, com os dados de nome, data de nascimento, sexo e raça/ cor.	B	A
RF12	A aplicação Inside Nutri deve permitir que o nutricionista possa trocar de senha.	B	B

ID	Descrição Resumida	Dificuldade (B/M/A)*	Prioridade (B/M/A)*
RF13	A aplicação Inside Nutri deve permitir que o nutricionista visualize uma listagem dos pacientes cadastrados.	B	M
RF14	A aplicação Inside Nutri deve permitir que o nutricionista altere alguma informação dos pacientes cadastrados.	B	B
RF15	A aplicação Inside Nutri deve calcular a idade dos pacientes a partir da data de nascimento.	M	A
RF16	A aplicação Inside Nutri deve realizar os cálculos com base nas informações fornecidas de cada paciente, por exemplo, caso seja de uma certa idade e raça/ cor irá calcular o Peso Estimado de acordo com uma formula específica.	M	A
RF17	A aplicação Inside Nutri deve permitir que o nutricionista visualize a porcentagem de energia fornecida através da dieta enteral em comparação ao gasto energético total.	M	A
RF18	A aplicação Inside Nutri deve permitir que o nutricionista visualize a porcentagem de proteína fornecida através da dieta enteral em comparação às proteínas totais diárias.	M	A
RF19	A aplicação Inside Nutri deve permitir que o nutricionista visualize a porcentagem de água fornecida em comparação à hidratação diária necessária.	M	A
RF20	A aplicação Inside Nutri deve permitir que o nutricionista envie os cálculos por e-mail.	B	B
RF21	A aplicação Inside Nutri deve permitir que o administrador possa realizar todas as funcionalidades e ainda descredenciar nutricionistas, caso esteja divulgando informações dos pacientes.	B	M

*B=Baixa, M=Média, A=Alta.

3.3 *Requisitos Não-funcionais*

Os Requisitos Não-Funcionais (RNF), representam restrições impostas às funções oferecidas pelo sistema ou ao processo de desenvolvimento. A seguir, são apresentados os Requisitos Não-Funcionais da aplicação Inside Nutri.

ID	Descrição	Prioridade B/M/A
RNF01	A aplicação Inside Nutri deve habilitar a autenticação baseado no modelo <i>OAuth2</i> do <i>Google</i> e diretamente no sistema.	A
RNF02	O sistema deve operar em tempo integral em 24h x 7d x 365, tendo disponibilidade mínima de 90%, e para atender essa necessidade deverá usar a hospedagem AWS ou <i>Microsoft Azure</i> com mecanismo de escala automática para responder ao aumento de demanda.	A
RNF03	A comunicação entre o sistema Back-End, Front-End e Mobile da aplicação Inside Nutri deve ser implementado através do padrão de serviços REST.	A
RNF04	O sistema deverá utilizar para persistência em um Banco de Dados NoSQL para atender a necessidade de alta performance na resposta em alta demanda, aceitando em contrapartida uma integridade eventual. Porém, pode ser necessário a utilização de um Banco de Dados Relacional em determinados Microserviços.	A
RNF05	As notificações enviadas da aplicação Inside Nutri por e-mail ou <i>push</i> devem operar por meio de filas de mensagens por não haver a necessidade de serem em tempo real e atenderem a grande demanda.	A
RNF06	A versão Web da aplicação Inside Nutri deve suportar os navegadores modernos, tais como Google Chrome, Microsoft Edge e Mozilla Firefox.	M
RNF07	Deve ser registrado uma trilha de auditoria para todas as alterações que ocorrerem no cadastro de clientes, cadastro de refeições e cadastro de dietas.	M
RNF08	O deploy em produção da aplicação Inside Nutri e a infraestrutura devem ser automatizadas usando pipelines de CI/CD.	M
RNF09	As API's devem possuir um <i>throughput</i> de no mínimo 300 tps.	A
RNF10	O processamento de grandes quantidades de dados deve ser feito preferencialmente entre às 0hs e 4hs.	M

3.4 Mecanismos Arquiteturais

Esta seção deve apresentar uma visão geral dos mecanismos que compõem a arquitetura do software, baseando-se em três estados: (1) análise, (2) design e (3) implementação. Em termos de Análise devem ser listados os aspectos gerais que compõem a arquitetura do software, como: persistência, integração com sistemas legados, geração de logs do sistema, ambiente de front end, tratamento de exceções, formato dos testes, formato de distribuição/implantação (deploy), dentre outros. Em Design deve-se identificar o padrão tecnológico a seguir para cada mecanismo identificado na análise. Em Implementação deve-se identificar o produto a ser utilizado na solução.

Os Mecanismos arquiteturais são soluções comuns para problemas comuns que podem ser usados durante o desenvolvimento para minimizar a complexidade. Eles representam os principais conceitos técnicos que serão padronizados em toda a solução. Facilitam a evolução de aspectos arquiteturais significativos do sistema. Eles permitem que a equipe mantenha uma arquitetura coesa enquanto permitem que os detalhes da implementação sejam adiados até que eles realmente precisem ser feitos.

Assim, os Mecanismos Arquiteturais são usados para satisfazer requisitos arquitetonicamente significativos. Normalmente, esses são requisitos não funcionais, como problemas de desempenho e segurança. Quando totalmente descritos, os Mecanismos Arquiteturais mostram padrões de estrutura e comportamento no software. Eles formam a base do software comum que será aplicado de forma consistente em todo o produto que está sendo desenvolvido. Eles também formam a base para padronizar a maneira como o software funciona, portanto, eles são um elemento importante da arquitetura geral do software. A definição de mecanismos de arquitetura também permite decisões sobre se os componentes de software existentes podem ser aproveitados para fornecer o comportamento necessário ou se um novo software deve ser comprado ou construído. Com isso, segue os mecanismos arquiteturais da aplicação Inside Nutri.

Análise	<i>Design</i>	Implementação
Persistência	ORM	NHibernate ou Hibernate
Persistência	Banco de Dados NoSql	MongoDB ou Amazon Dynamo DB
Front-End	Single Page Application	Angular
Front-End	Mobile	Kotlin
Back-End	Arquitetura em Camadas	.Net Core ou Spring Boot
Gateway	API Gateway	AWS API Gateway
Integração	API RestFull	APIs externas a aplicação Inside Nutri
Log do sistema	Gestão de Logs da aplicação	Slack ELK
Teste de Software	Testes de Unidade	JUnit ou XUnit
Versionamento	Repositório contendo o código fonte	GitHub
Deploy	Integração e Entrega continua (CI/CD)	AWS Code Pipelines
Mensagens	Mensageria e/ou notificação de push	Amazon Pinpoint e/ou RabbitMQ
API	Documentação das APIs da aplicação Inside Nutri	Swagger
Autenticação	OAuth 2.0	Open Authorization 2.0

4. Modelagem Arquitetural

Esta seção apresenta a modelagem arquitetural da solução proposta, de forma a permitir seu completo entendimento visando à implementação da prova de conceito (PoC) da aplicação Inside Nutri, descrito na seção 5 deste trabalho.

Para esta modelagem arquitetural optou-se por utilizar o modelo C4 para documentação de arquitetura de software. Mais informações a respeito podem ser encontradas aqui: <https://c4model.com/> e aqui: <https://www.infoq.com/br/articles/C4-architecture-model/>. Dos quatro nível que compõem o modelo C4 três serão apresentados aqui e somente o Código será apresentado na próxima seção (5).

4.1 Diagrama de Contexto

O diagrama de contexto modelo C4 é usada para descrever e definir arquiteturas de forma abstrata e simples. O modelo C4 é uma maneira de apresentar a macroarquitetura da solução que aborda o desenvolvimento de software, apresentando em sua modelagem a concentração em quatro c's: (i) contexto (pessoas); (ii) contêineres; (iii) componentes; e (iv) código. (BROWN, 2020). Para a aplicação proposta temos o seguinte diagrama de contexto:

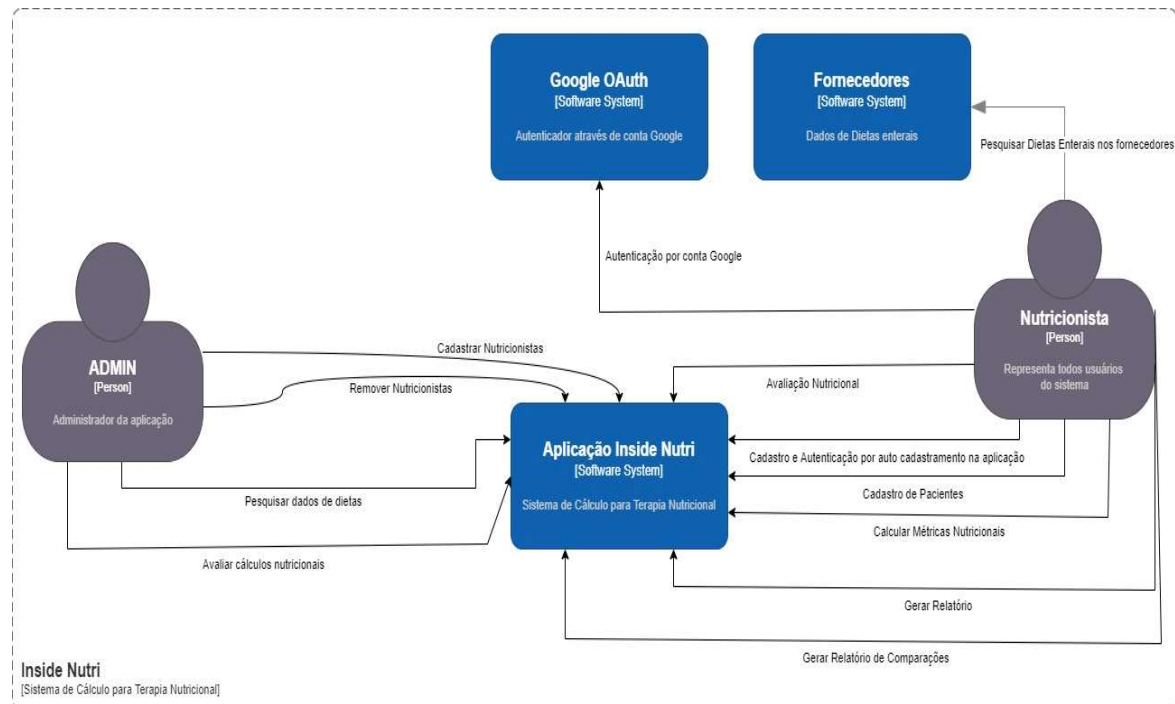


Figura 1 - Visão Geral da Solução Inside Nutri. Fonte: Do Autor.

A Figura 1 - **Visão Geral da Solução Inside Nutri**. Fonte: **Do Autor** mostra a especificação o diagrama geral da solução proposta, com todos seus principais módulos e suas interfaces, tais como a integração da aplicação Inside Nutri com a autenticação do OAuth2.0 e com sistemas de fornecedores via API. Podemos verificar os dois atores da aplicação, sendo o administrador e o nutricionista. É possível verificar as funcionalidades que cada ator poderá realizar na aplicação.

4.2 Diagrama de Container

O diagrama *Container* mostra a forma de alto nível da arquitetura de software e como as responsabilidades são distribuídas por ela. Ele também mostra as principais opções de tecnologia e como os *containers* se comunicam entre si. É um diagrama simples e focado em tecnologia de alto nível que é útil tanto para desenvolvedores de software quanto para equipes de suporte/operações. Para a aplicação proposta temos o seguinte Diagrama de *Container*:

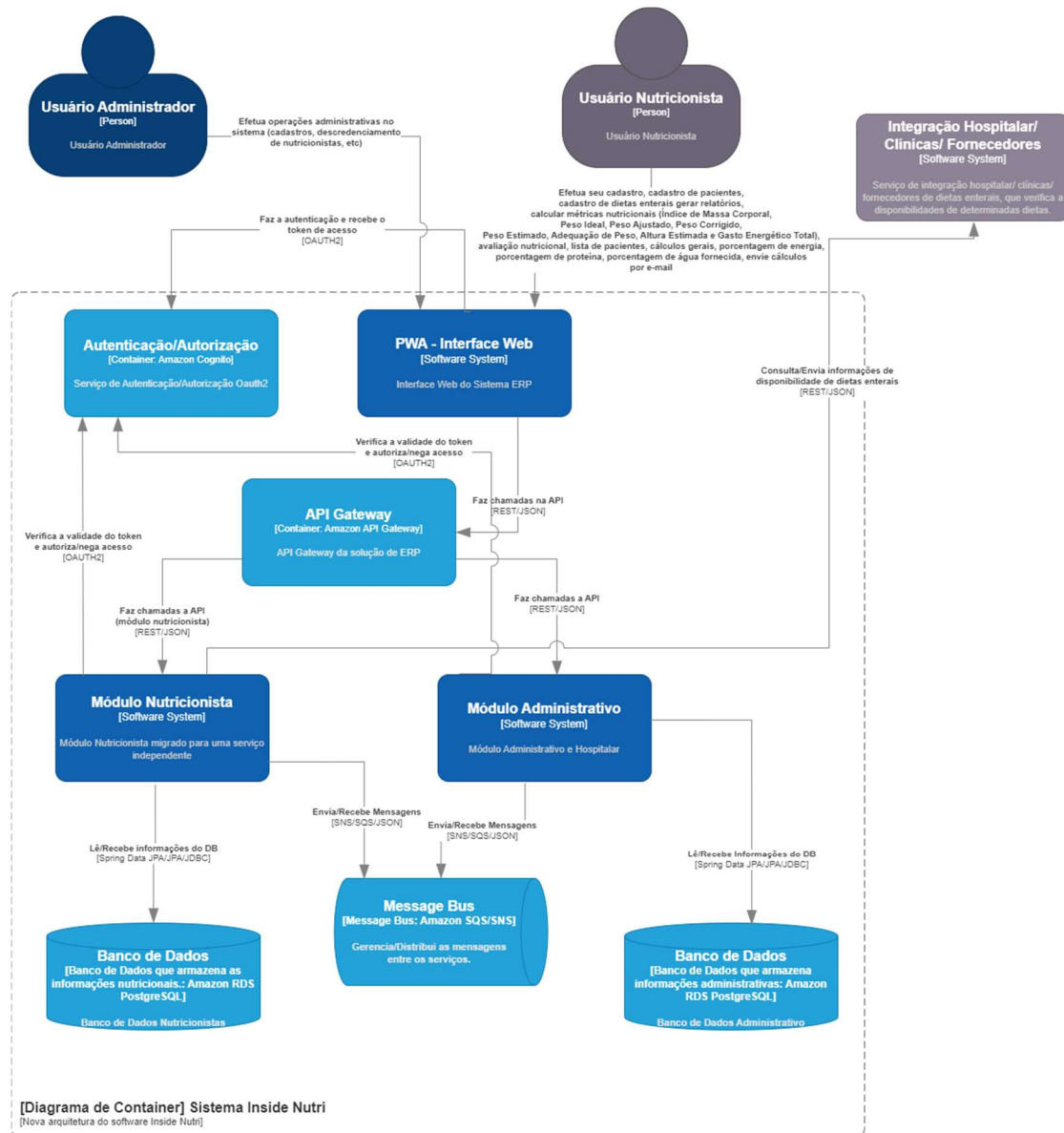


Figura 2 – Diagrama de Container. Fonte: Do Autor.

A Figura 2 – **Diagrama de Container. Fonte: Do Autor** apresenta os containers da aplicação Inside Nutri de forma amplificada os containers que compõem os módulos Nutricionista e Administrativo, onde em cada *container* é evidenciado sua natureza, bem como a tecnologia que é usada e também os protocolos de integração com os quais eles interagem entre si. A página web é onde são servidos os conteúdos estáticos (HTML, CSS e JavaScript) fornecido pelo NGINX que compõem a aplicação web feita em Angular que por sua vez será renderizada aos usuários via navegador web.

Existem 2 papéis, para utilização do sistema, o usuário do Administrador e o usuário do módulo Nutricionista. No dia a dia pode acontecer do mesmo usuário assumir ambos os papéis, mas para fins de explicação, foi escolhido detalhar como figuras distintas.

O usuário faz o login na aplicação frontend PWA. A aplicação frontend, em sua tela de login, requisita um *token* de acesso para o serviço o OAuth2, que é fornecido pelo Amazon Cognito. Com o *token* de acesso disponível, a aplicação web é autorizada a fazer requisições.

As requisições são feitas através do API Gateway, utilizando a tecnologia Amazon API Gateway. Dependendo da requisição, o API Gateway redireciona as requisições para o módulo nutricionista ou módulo administrativo.

Tanto o módulo nutricionista quanto o módulo administrativo possuem bancos de dados independentes. A comunicação entre os módulos será feita via mensageria, funcionando assim de forma desacoplada e utilizando as tecnologias Amazon SQS (Fila) e Amazon SNS (tópicos). E para requisições web dos usuários, as aplicações, de forma independente, fazem a checagem do *token* de acesso com o servidor OAuth2 (Amazon Cognito).

Toda e qualquer autenticação e autorização será feita diretamente no API Gateway, devido características internas e tecnológica. O módulo nutricionista comunica-se via interface REST com o Integração Hospitalar/ Clínicas/ Fornecedores, que é um serviço externo de uma empresa parceira. O Integração Hospitalar/ Clínicas/ Fornecedores, possui uma interface hospitalar, clínicas ou fornecedores de dietas enterais, que verifica a disponibilidades de determinadas dietas na região.

4.3 Diagrama de Componentes

Um diagrama de componentes, também conhecido como diagrama de componentes UML, descreve a organização e a fiação dos componentes físicos em um sistema. Os diagramas de componentes geralmente são desenhados para ajudar a modelar os detalhes da implementação e verificar novamente se todos os aspectos das funções exigidas do sistema são cobertos pelo desenvolvimento planejado. Para a aplicação proposta temos o seguinte Diagrama de Componentes:

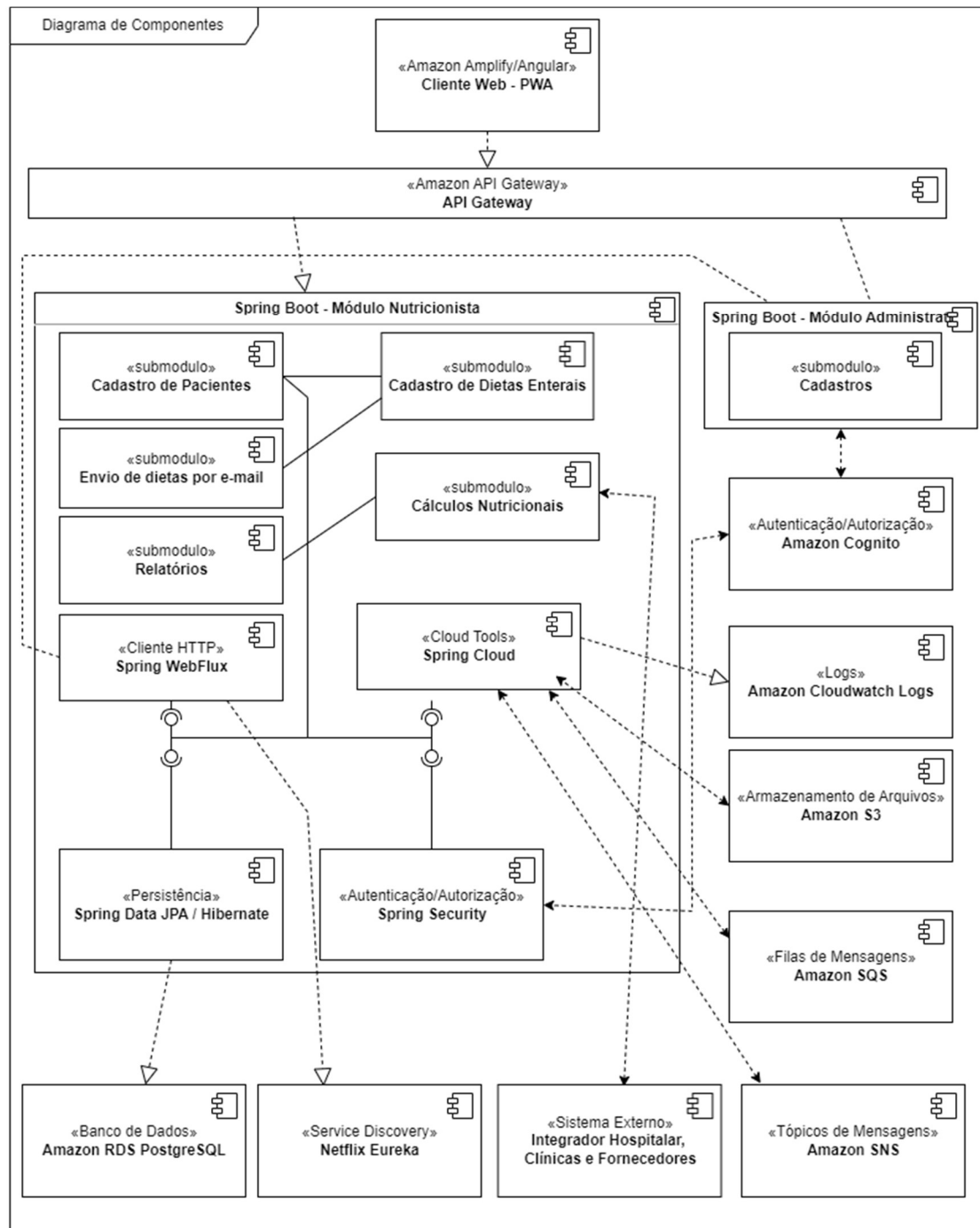


Figura 3 – Diagrama de Componentes. Fonte: Do Autor.

A **Figura 3 – Diagrama de Componentes. Fonte: Do Autor.** mostra o Diagrama de Componentes do sistema de software Inside Nutri, seguindo o padrão UML. Descrevendo suas dependências com mais detalhes da base tecnológica da aplicação. Os componentes da solução:

- **Cliente Web/PWA:** É uma aplicação desenvolvida em Angular, utilizando o pacote de componentes angular/pwa e também a biblioteca AWS

Amplify para a integração com o serviço Amazon Cognito (servidor OAuth2).

- **API Gateway:** É um serviço Amazon API Gateway como solução de API Gateway, que é utilizado para unificar a API dos serviços do Módulo Nutricionista e Módulo Administrativo em um único *endpoint*;
- **Integração Hospitalar/ Clínicas/ Fornecedores:** serviço externo de uma empresa parceira responsável pela integração de dietas enterais. A comunicação é realizada via disponibilização de uma API REST para fazer essa integração;
- **Amazon RDS PostgreSQL:** É o servidor de banco de dados do módulo nutricionista. Assim, a utilização do serviço de bancos de dados é gerenciada pela Amazon, o Amazon RDS. É utilizado a engine de banco de dados PostgreSQL;
- **Netflix Eureka:** Serviço de Service Discovery é um dos principais princípios da arquitetura baseada em microservices. Tentando configurar a mão para cada cliente ou alguma forma de convenção. O Eureka é o Service Discovery do Netflix que é usado no Server e no Cliente;
- **Amazon Cognito:** É um servidor OAuth2, responsável por realizar as autenticações e autorizações dos usuários do sistema de software Inside Nutri;
- **Amazon S3:** É um serviço de armazenamento de objetos (arquivos) da Amazon;
- **Amazon SQS:** É um *Amazon Simple Queue Service* (SQS) é um serviço de filas de mensagens gerenciado;
- **Amazon SNS:** É um serviço de notificação da Amazon. Com sua utilização é possível criar tópicos de mensagens. É uma solução para padronizar as mensagens em pub/sub de mensagens;
- **Amazon Cloudwatch Logs:** É utilizado para o armazenamento de logs dos serviços;

- **Módulo Administrativo:** É um módulo que tem funcionalidades específicas para manter o sistema de software Inside Nutri executando com alta disponibilidade e o mínimo de manutenção;
- **Módulo Nutricionista:** É um módulo principal do sistema de software Inside Nutri que é a solução proposta nesse trabalho. Utiliza como base o *framework* Spring Boot. Tendo como principais funcionalidades:
 - **Cadastro de Pacientes:** É um submódulo dentro do Módulo Nutricionista, responsável pelo cadastro dos pacientes que cada nutricionista está atendendo e acompanhando;
 - **Cadastro de Dietas Enterais:** É um submódulo dentro do Módulo Nutricionista, responsável pelo cadastro das dietas enterais para cada paciente atendido e acompanhado pelos nutricionistas;
 - **Envio de Dietas por E-mail:** É um submódulo dentro do Módulo Nutricionista, responsável pelo envio das dietas enterais por e-mail para cada paciente cadastrado;
 - **Relatórios:** É um submódulo dentro do Módulo Nutricionista, responsável por gerar relatórios qualitativos e quantitativos dos atendimentos e/ ou pacientes, bem como das dietas enterais;
 - **Cálculos Nutricionais:** É um submódulo dentro do Módulo Nutricionista, responsável pela parte mais importante do sistema de software Inside Nutri. Este realiza todos os cálculos necessários para os nutricionistas definir quais e quanto de dieta enteral cada paciente necessita e por quanto tempo;
 - **Spring Web Flux:** é uma biblioteca do Spring Boot responsável por consultas REST;
 - **Spring Cloud:** É uma biblioteca para o Spring Boot, utilizada para auxiliar no gerenciamento de serviços cloud, tais como, Amazon Web Services. Nessa solução, servirá para realizar a comunicação com o Amazon SQS e SNS. E também será possível comunicar com o serviço Amazon S3. Além disso, será utilizado para

disponibilizar o serviço de Service Discovery utilizando o Netflix Eureka para o Spring Web Flux;

- **Spring Data JPA/Hibernate:** O Spring Data JPA é um facilitador para a criação de repositórios JPA. O JPA, sigla para Java Persistence API, este é uma API padrão para Java que faz a interface com o banco de dados. Assim, é utilizado a biblioteca Hibernate como implementação da API JPA.

5. Prova de Conceito (PoC)

Uma prova de conceito (PoC) é o primeiro passo no processo de desenvolvimento de software após o desenvolvimento da ideia geral do produto. Uma PoC visa validar a viabilidade do projeto e verificar a viabilidade geral da ideia. O objetivo do PoC, do ponto de vista do desenvolvimento de software, é mostrar a viabilidade tanto da ideia do produto quanto do plano de negócios. Assim, uma prova de conceito devidamente apresentada ajuda a evitar erros dispendiosos e facilita a atração de investidores.

Para a aplicação proposta temos aspectos importantes para a arquitetura e que foram contemplados, sendo:

- **Persistência de dados:** O objetivo dessa PoC foi avaliar o o Spring Data JPA/Hibernate a fim de entender sua implementação e funcionamento, buscando comprovar sua performance que é considerada uma vantagem em relação a outras bibliotecas semelhantes. Espera-se que com a utilização da biblioteca seja possível obter maior controle na performance de queries;
- **Design *pattern* para API:** O objetivo dessa PoC foi comprovar que a arquitetura em camadas alinhada a conceitos de DDD (Domain-Driven Design) pode organizar bem o código diante da grande quantidade de regras de negócio que compõem a aplicação. Espera-se que seja possível incluir, retirar ou modificar regras de negócio de forma simples e rápida;

- **Angular:** O objetivo dessa POC foi estudar a utilização do framework Angular para desenvolvimento do front-end. Espera-se que com a utilização desse framework seja possível desenvolver um sistema responsivo, performático e que não dependa de muitos componentes de terceiros para seu funcionamento.

5.1 Integrações entre Componentes

A PoC foi desenvolvida a partir de requisitos prioritários coerentes a especificação proposta e implementada de maneira eficaz afim de permitir a validação das integrações entre os componentes.

Cadastro de Usuários:

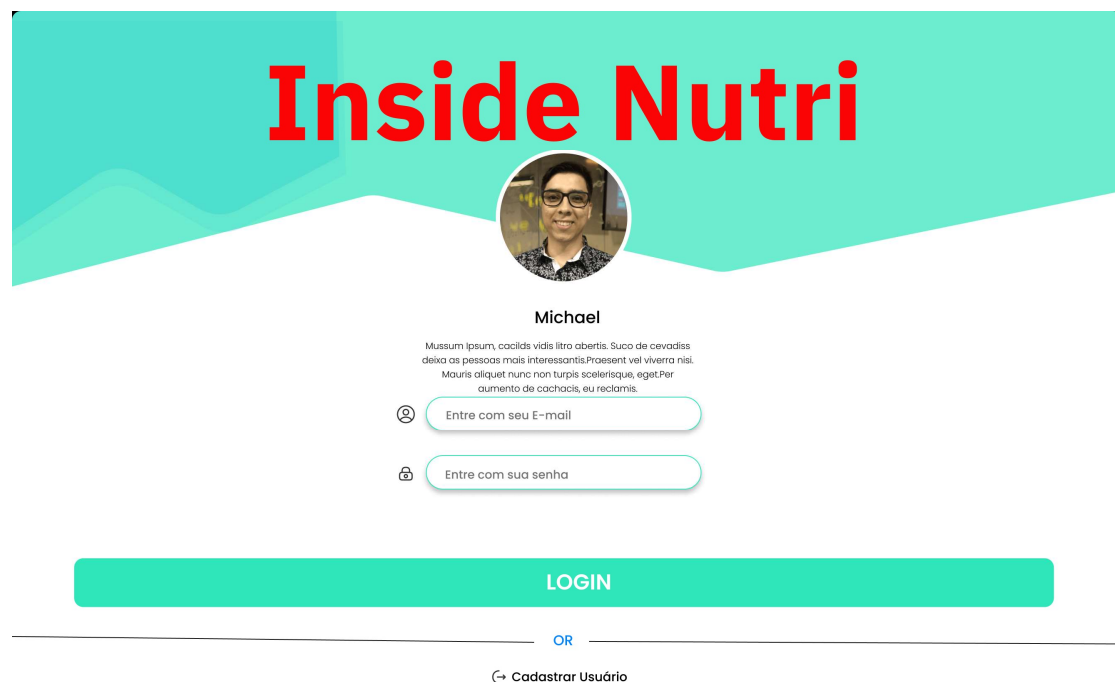
- **Componentes e Middlewares Envolvidos:** Portal Web (PWA) e API de autenticação;
- **Integrações e Protocolos de Comunicação:** Comunicação com a API RESTful de usuários (JSON);
- **Requisitos Não-Funcionais:** Segurança e Usabilidade.



Figura 4 - Tela de Cadastro de Usuários. Fonte: Do Autor.

Acesso (Login) de Usuários:

- **Componentes e Middlewares Envolvidos:** Portal Web (PWA);
- **Integrações e Protocolos de Comunicação:** Comunicação via OAuth2 e Comunicação com Banco de Dados via TCP/JDBC;
- **Requisitos Não-Funcionais:** Segurança, Usabilidade, Compatibilidade.



Inside Nutri

Michael

Mussum ipsum, cacilds vidis litro abertis. Suco de cevadiss
deixa as pessoas mais interessantes! Prosent vel viverra nisi.
Mauris aliquet nunc non turpis scelerisque, eget. Per
aumento de cachacis, eu reclamis.

Entre com seu E-mail

Entre com sua senha

LOGIN

OR

← Cadastrar Usuário

Figura 5 - Tela de Login. Fonte: Do Autor.

Tela Inicial (Dashboard):

- **Componentes e Middlewares Envolvidos:** Portal Web (PWA);
- **Integrações e Protocolos de Comunicação:** Comunicação com a API de paciente, atendimentos e dietas via RESTful (JSON);
- **Requisitos Não-Funcionais:** Segurança, Usabilidade, Compatibilidade.

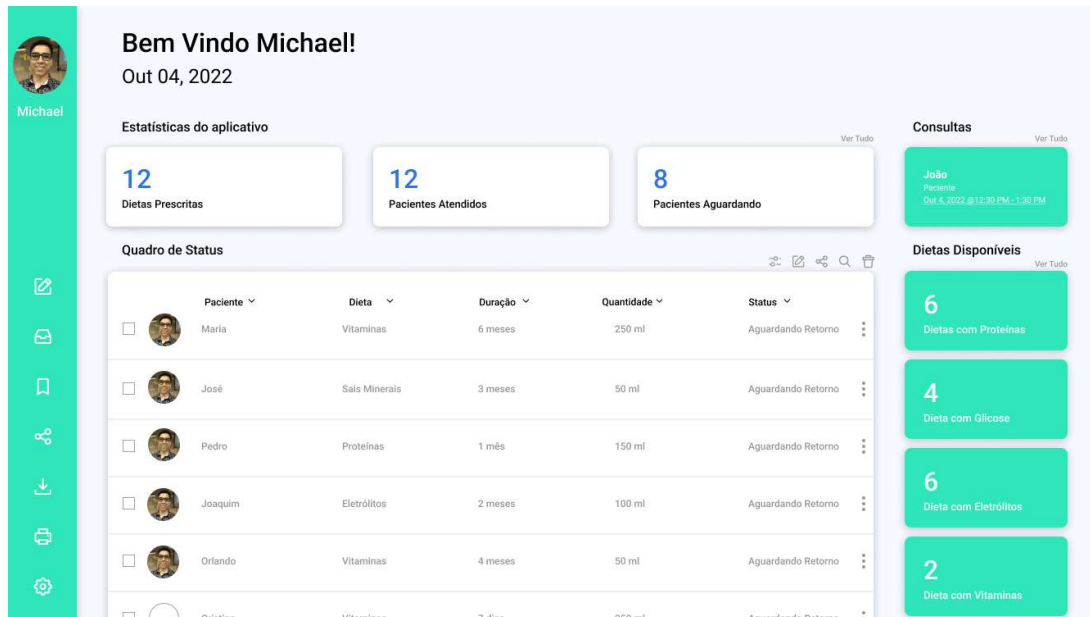


Figura 6 - Tela Inicial (Dashboard). Fonte: Do Autor.

Tela de Cadastro de Nova Dieta do Usuário:

- **Componentes e Middlewares Envolvidos:** Portal Web (PWA);
- **Integrações e Protocolos de Comunicação:** Comunicação com a API de paciente via RESTful (JSON), Comunicação com a API de dieta via RESTful (JSON) e Comunicação com Banco de Dados via TCP/JDBC;
- **Requisitos Não-Funcionais:** Segurança, Usabilidade, Compatibilidade.

Figura 7 - Tela de Cadastro de Nova Dieta do Usuário. Fonte: Do Autor.

Tela de Realizar Cálculos de Dieta:

- **Componentes e Middlewares Envolvidos:** Portal Web (PWA);
- **Integrações e Protocolos de Comunicação:** Comunicação com a API de paciente via RESTful (JSON), Comunicação com a API de dieta via RESTful (JSON), Comunicação com a API de cálculos via RESTful (JSON) e Comunicação com Banco de Dados via TCP/JDBC;
- **Requisitos Não-Funcionais:** Segurança, Usabilidade, Compatibilidade.

Realizar Cálculos

Paciente

Obama

Tipo de Cálculo

Peso estimado

Cor/ Raça

Branco

Altura

1,81 mts

Resultado

Resultado do Peso Estimado

Calcular Salvar

Figura 8 - Tela de Realizar Cálculos de Dieta. Fonte: Do Autor.

5.2 *Código da Aplicação*

Nessa sessão será explicado a nível de código o funcionamento dos requisitos escolhidos. O código fonte completo da aplicação pode ser acessado no endereço: <https://github.com/michaelTadeu/projeto-aplicado>:

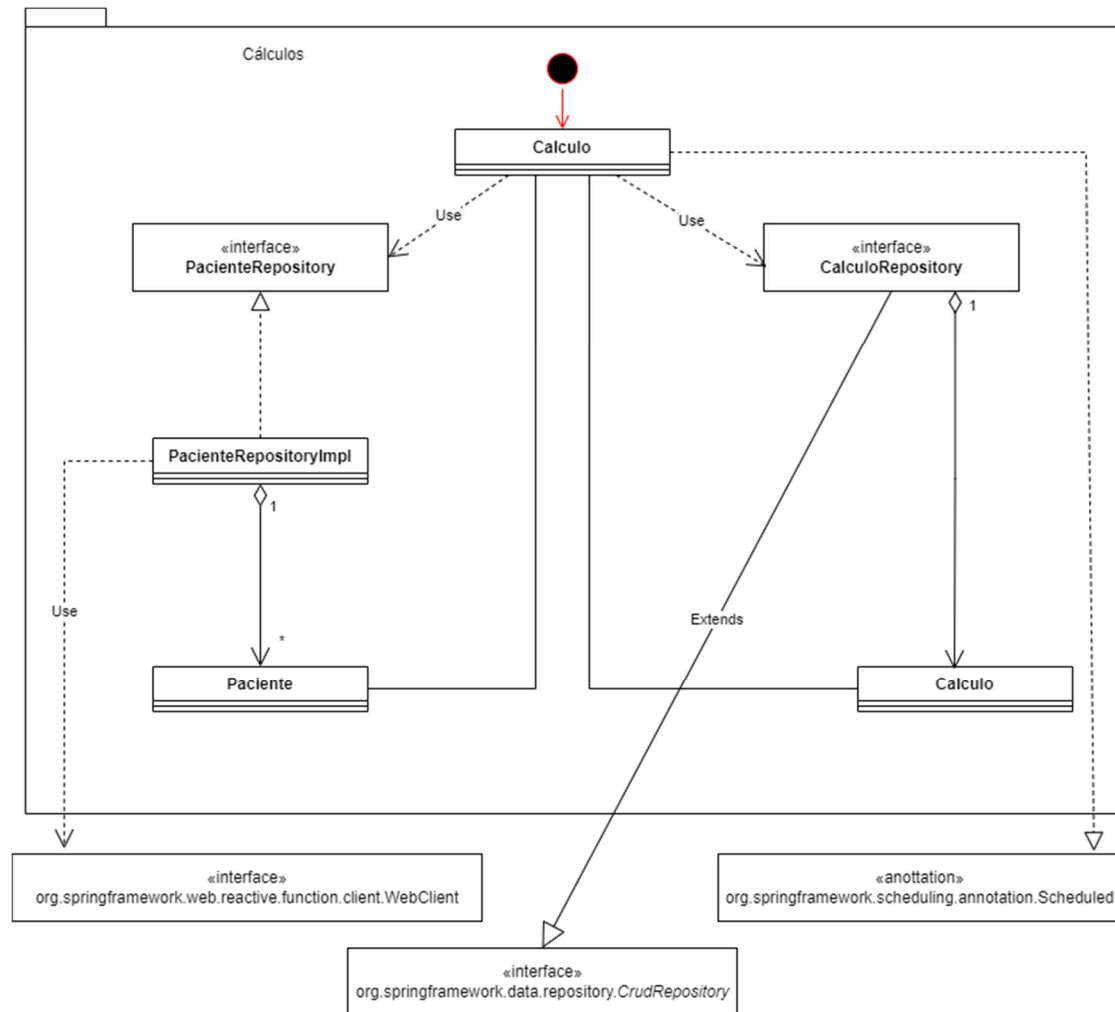


Figura 9 - Código do fluxo para realizar Cálculos. Fonte: Do Autor.

Na Figura 9 - Código do fluxo para realizar Cálculos. Fonte: Do Autor. é detalhado o funcionamento dos Cálculos. A classe **Calculo** é uma *task* agendada do Spring Boot, configurada com a anotação **Scheduled**. São consultadas no banco de dados todas os cálculos possíveis de serem feitos através da interface **CalculoRepository**, que utiliza o **CrudRepository** do Spring Data JPA para gerar uma implementação de um **Repository** em tempo de execução. Após a consulta dos dados, são enviados os cálculos para o respectivo paciente via classe **PacienteRepository**. Assim, os cálculos realizados para cada paciente são armazenados e ficam disponíveis para o nutricionista consultar posteriormente.

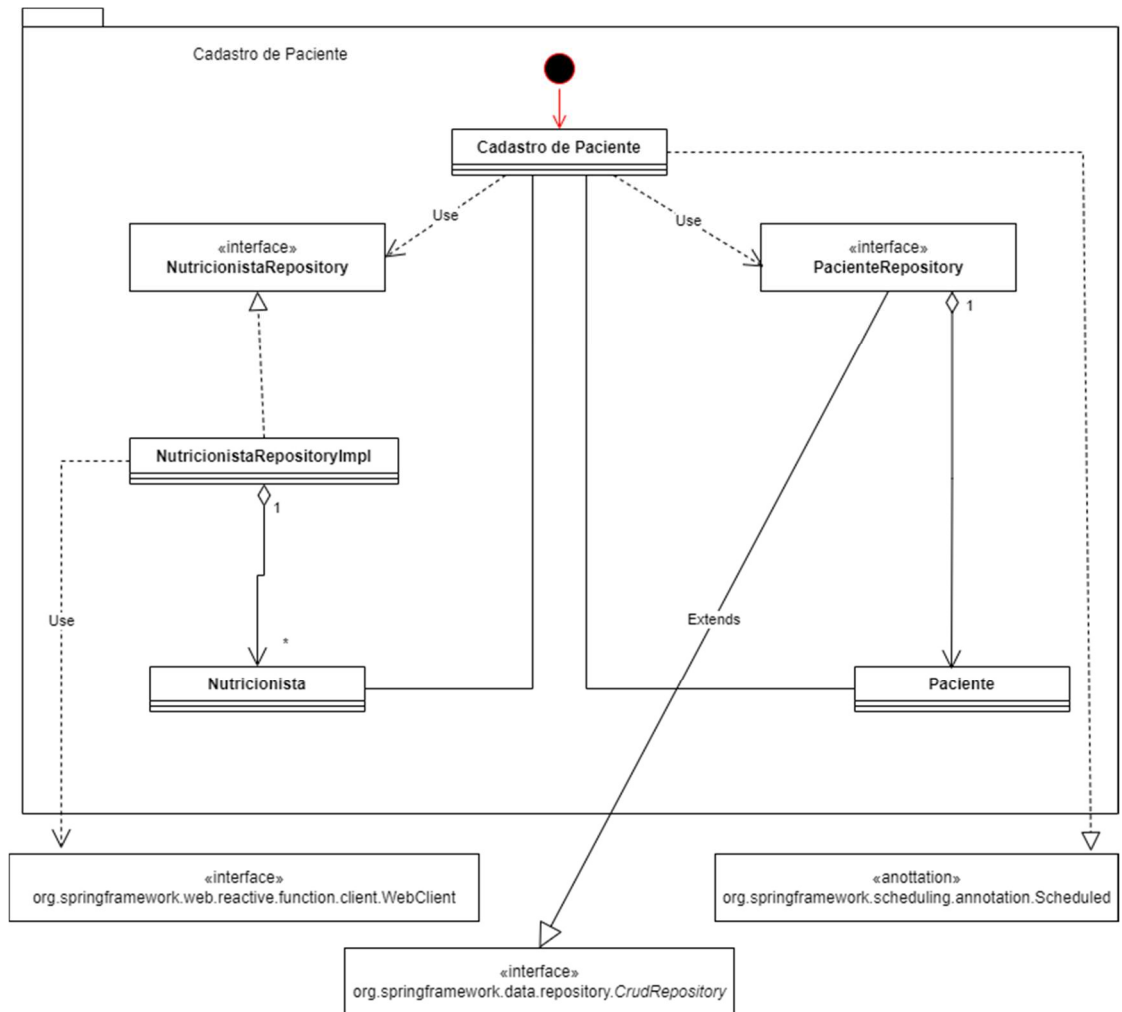


Figura 10 - Código do fluxo para realizar Cadastro de Pacientes. Fonte: Do Autor.

Na Figura 10 - Código do fluxo para realizar Cadastro de Pacientes. Fonte: Do Autor. é detalhado o funcionamento do Cadastro de Pacientes. A classe Cadastro de Pacientes consulta os dados do nutricionista no banco de dados através da interface **NutricionistaRepository**, que utiliza o **CrudRepository** do Spring Data JPA para gerar uma implementação de um Repository em tempo de execução. Após a consulta dos dados, são enviados os dados dos pacientes para a via classe **PacienteRepository**. Assim, os pacientes cadastrados são armazenados e ficam atrelados aos nutricionistas que efetuou o cadastro.

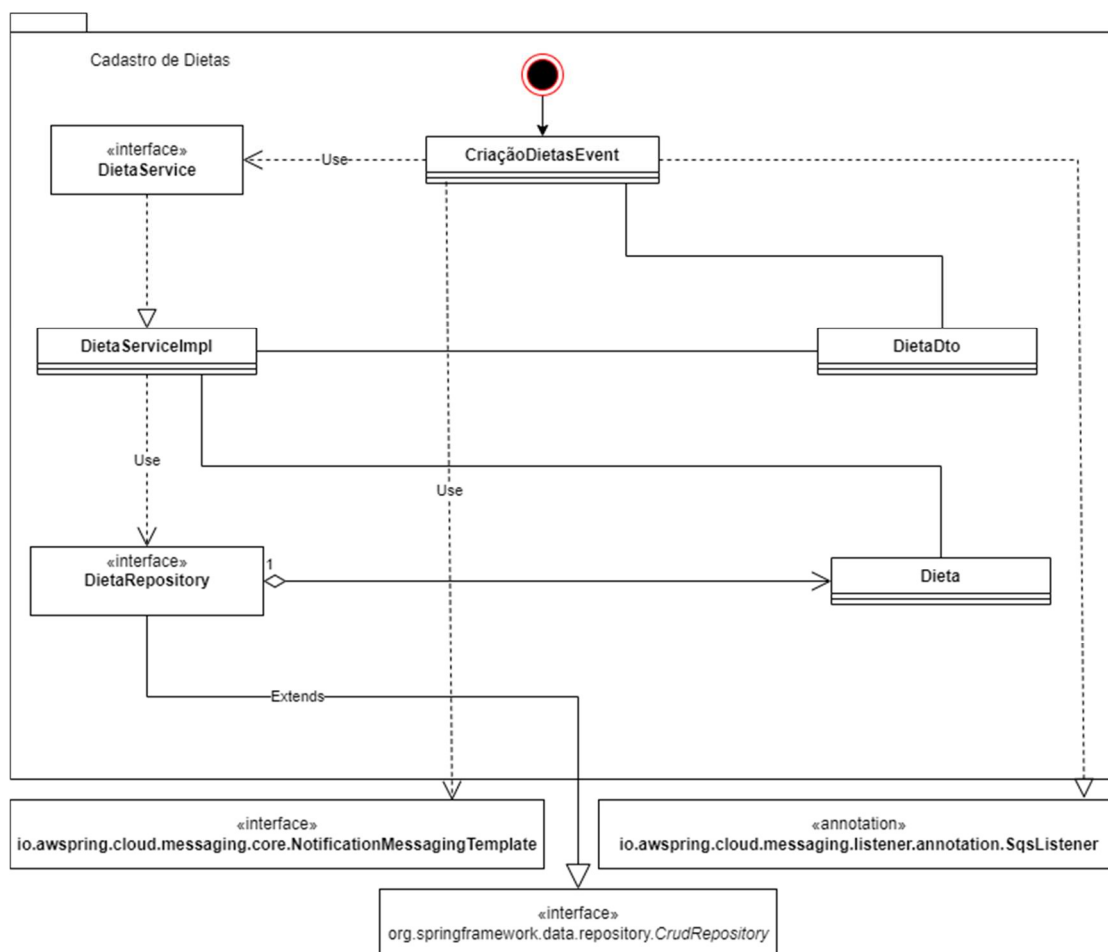


Figura 11- Código do fluxo para realizar Cadastro de Dietas. Fonte: Do Autor.

A classe **CriacaoDietasEvent** é um listener de uma fila SQS, e é anotada com a annotation **SqsListener** do Spring Cloud. Essa classe recebe a mensagem no formato JSON, que segue a estrutura do **DietaDto**. Com o **DietaDto** preenchido, é utilizada a interface **DietaService**, e sua implementação, a classe **DietaServiceImpl**, para iniciar o processo de salvar. O **DietaService** é a classe de negócios do Cadastro de Dietas. O **DietaDto** é convertido para a classe de módulo **Dieta**, e é persistida no banco de dados com a interface que **DietaRepository**, uma interface de implementação do Spring Data JPA. Após a persistência, o fluxo volta para a **CriacaoDietasEvent**, onde é enviada a resposta de sucesso ou falha via tópico do Amazon SNS utilizando a classe **NotificationMessagingTemplate** do Spring Cloud. Com isso é criado (ou não) o Cadastro da Dieta é notificado todos os serviços que são inscritos no tópico SNS correspondente.

Etapa 3 - Pendente

<Conteúdo a ser produzido – Data final 15 de Dezembro>

Referências

OLIVEIRA, L.B.; JUNIOR, P.B.R.; GUIMARÃES, N.M.; DIDONET, M.T. Variáveis relacionadas ao tempo de internação e complicações no pós operatório de pacientes submetidos à cirurgia do trato gastrointestinal. *Comunicação em Ciências da Saúde*. Vol. 21. Num. 4. 2010. p. 319-330.

MINISTÉRIO DA SAÚDE. Secretaria de Atenção à Saúde. Manual de Terapia Nutricional na Atenção Especializada Hospitalar. Brasília. Ministério da Saúde. 2016.

LOPES, D.A.M.; MANZATTI, F.; BRUSTOLIN, A.; PENTEADO, E.G.; FRANCO, S.; BARATTO, I. Perfil nutricional de pacientes em um hospital de Guarapuava-PR. *Anais da SIEPE. Semana de Integração Ensino, Pesquisa e Extensão*. 26 a 30 de outubro de 2009.

LEITE, H. P.; CARVALHO, W. B.; MENESES, J. F. S. Atuação da equipe multidisciplinar na terapia nutricional de pacientes sob cuidados intensivos. *Revista de Nutrição*, 2005.

GARCIA, R.W.D.; PADILHA, M.; SANCHES, M. Alimentação hospitalar: proposições para a qualificação do Serviço de Alimentação e Nutrição, avaliadas pela comunidade científica. *Revista Ciência & Saúde Coletiva*. Vol. 17. Num. 2. 2012. p. 473-480.

CFN - CONSELHO FEDERAL DE NUTRICIONISTAS. Resolução CFN nº 600, de 25 de fevereiro de 2018. Dispõe sobre a definição das áreas de atuação do nutricionista e suas atribuições, indica parâmetros numéricos mínimos de referência, por área de atuação, para a efetividade dos serviços prestados à sociedade e dá outras providências. *Diário Oficial da União*, p. 1-55, 2018.

WAITZBERG, Dan Linetzky; CAIAFFA, Waleska T.; CORREIA, M. I. T. D. Inquérito brasileiro de avaliação nutricional hospitalar (Ibranutri). *Revista Brasileira de Nutrição Clínica*, v. 14, p. 124-34, 1999.

SOMMERVILLE, I. Engenharia de Software. 10ª. edição. Pearson Universidades, 768 p., 2019.

CHITCHYAN, R.; RASHID, A.; SAWYER, P.; GARCIA, A.; ALARCON, M. P.; BAKKER, J.; TEKINERDOGAN, B.; CLARKE, S.; JACKSON, A. Report Synthesizing State-of-the-Art in Aspect-Oriented Requirements Engineering, Architectures and Design. AOSD-Europe Deliverable D, 11, 1-259. 2005.

RIEHLE, Dirk; ZÜLLIGHOVEN, Heinz. Understanding and using patterns in software development. *Tapos*, v. 2, n. 1, p. 3-13, 1996.

BROWN, Simon. The c4 model for visualising software architecture (2020). 2020.