Associação Propagadora Esdeva
Centro Universitário Academia – UniAcademia
Curso de Engenharia de Software
Projeto de Pesquisa – Artigo

Integração de Software em Ambiente Corporativo

Ana Carolina Machado Vasconcellos Oliveira[1]
Centro Universitário Academia, Juiz de Fora, MG
Lucas Pablo de Paula Silva [2]
Centro Universitário Academia, Juiz de Fora, MG
Lucas Piccinini Gonzaga [3]
Centro Universitário Academia, Juiz de Fora, MG
Marilia Roberto da Cruz Carvalho [4]
Centro Universitário Academia, Juiz de Fora, MG
Renzo Faedda Panza [5]
Centro Universitário Academia, Juiz de Fora, MG
Vagner Gonçalves Vieira [6]
Centro Universitário Academia, Juiz de Fora, MG
Jacimar Fernandes Tavares [2]
Centro Universitário Academia, Juiz de Fora, MG

Linha de Pesquisa: Engenharia de *Software*

RESUMO

Sabe-se que ao longo do tempo grandes projetos de software tem ganhado espaço de mercado ao propor cada vez mais novas possibilidades para seus usuários, como, por exemplo, a conexão de diferentes componentes de software fazendo com a nova configuração traga novidades, em termos de recursos e dados. A isso, dá-se o nome de integração de software. A função do responsável pelas integrações não está limitada a apenas o ato de integração de processos, mas sim na participação de tudo que permeia o processo de desenvolvimento das novas tecnologias.
Este projeto de pesquisa tem por objetivo viabilizar novas integrações entre plataformas já existentes, permitindo assim que novos recursos sem explorados nas tecnologias integradas.
Palavras-chave: Integração de Software, Ambiente Corporativo.

[1, 2, 3, 4, 5, 6] Discentes do Curso de Bacharelado em Engenharia de Software do Centro Universitário Academia –
[27] Docente do Curso de Bacharelado em Engenharia de Software do Centro Universitário Academia. Orientador.

1. INTRODUÇÃO

A integração de software veio da necessidade de manter dados necessários a sistemas diferentes, reunidos em um mesmo lugar, resolvendo o problema de ter dados isolados. Quando os sistemas começam a conversar entre si, torna-se mais fácil gerenciar os dados que precisam ser usados [1].

Integrações de software tornam-se ainda relevantes quando relacionadas a empresas e diversos ambientes do mercado de trabalho. Focando na relação das integrações com as aplicações internas das empresas (ambientes corporativo), pode-se chegar a um ecossistema de aplicações que conversem entre si, gerando informações que não seriam conhecidas, caso não houvesse a integração [2].

Ao analisar a função das integrações, é perceptível que facilidades são trazidas ao ambiente de trabalho. A exemplo, se caso precise comunicar com dados que estão em outro sistema, com a ajuda de uma integração, será possível a criação de informações que antes não eram conhecidas, o que pode trazer benefícios à corporação.

O objetivo deste projeto de pesquisa é desenvolver, juntamente com os discentes e a empresa parceira, conhecimentos necessários para que possam ser conduzidas, dentro da empresa parceira, projetos reais de integração entre seus produtos corporativos que são parte dos objetivos da empresa. Para isso, o grupo discente selecionado pôde passar por treinamentos, estudos e aprendizados de grupo e individuais para se adequarem ao objetivo.

Este artigo se divide nas seguintes seções: a seção 2 apresenta o referencial teórico base para o entendimento de tudo que foi estudado e implementado ao longo do projeto de pesquisa. Nela são apresentados conceitos como integração de software, middleware, APIs, armazenamento em nuvem e etc. Na seção 3 é apresentada em detalhes a metodologia usada durante todo o projeto de pesquisa, passando pela definição do escopo do projeto, a seleção de pessoas e a execução das atividades programadas. A seção 4 apresenta os pontos principais dos resultados e discussões gerados ao longo de todo o projeto de pesquisa, com ilustrações reais dos estudos de casos conduzidos. Por fim, na seção 5 são apresentadas as considerações finais deste projeto e os trabalhos futuros (que serão desenvolvidos como continuidade deste mesmo projeto).

2. REFERENCIAL TEÓRICO

Nesta seção são apresentados os conceitos que são relevantes para o entendimento de tudo que foi desenvolvido ao longo deste projeto de pesquisa. Neste sentido, são apresentados ao leitor todas as teorias com as quais se trabalhou no sentido de viabilizar a implementações feitas.

2.1. Integração de software

Integração se refere à incorporação de um elemento num conjunto, no caso de *softwares*, a integração refere-se ao ato de fazer dois ou mais sistemas ou softwares comunicarem entre si e serem capazes de trocarem informações uns com os outros [3].

Para se realizar integrações, pode-se utilizar a assistência de um *middleware*, que é um *software* que se encontra entre o sistema operacional e os aplicativos executados nele. *Middleware* funciona como uma camada de tradução, que permite a comunicação e o gerenciamento de dados para aplicativos [4].

Exemplos comuns de uso de integrações que envolvem *middleware* incluem o uso do mesmo em banco de dados, em servidor de aplicativos, entre outros. Embora todos os tipos de *middleware* executam funções de comunicação, o tipo utilizado dependerá do tipo de serviço sendo utilizado e do tipo de informação que deve ser comunicado.

2.2. Camada Middleware

Middleware é um *software* que fornece recursos para aplicações. Com a ajuda de um *middleware*, os desenvolvedores são capazes de criar aplicações com mais facilidade e eficiência, pois estes possuem o papel de conectar aplicações, dados e usuários [7].

O termo *middleware* surgiu após uma conferência de Engenharia de Software da OTAN de 1968 na Alemanha [8]. *Middleware* engloba várias coisas, desde servidores webs a sistemas de autenticação e ferramentas de mensagem. Alguns

dos casos comuns do uso de *middleware*: (i) o desenvolvimento de novas aplicações, (ii) otimização das aplicações existentes, (iii) integração abrangente, (iv) interfaces de programação de aplicações (APIs), (v) transmissão de dados e (vi) automação corporativa inteligente [8]. A presença de *middlewares* na transmissão de dados costuma auxiliar no compartilhamento de informações entre aplicações de forma assíncrona. Os dados são replicados em um repositório intermediário, onde eles podem ser compartilhados entre várias aplicações.

2.3. APIs e as API usadas

API, ou *Application Programming Interface,* é uma estrutura responsável por permitir a comunicação entre *softwares* seguindo protocolos e definições previamente definidas. Assim, entendendo mais o termo, podemos dizer que a*pplication,* ou aplicação, faz referência a qualquer *software* com funções diferentes. E *interface*, ou interface, faz referência ao contrato de serviço entre as aplicações, isso é as normas e regras para troca de informações e comunicação entre os *softwares* através de solicitações e respostas [3].

Dessa forma, podemos dizer que a aplicação que envia a solicitação é chamada de cliente e a aplicação que envia a resposta é chamada de servidor. Existem quatro modos diferentes que uma API pode funcionar: API SOAP, API RPC, API WebSocket e API REST. Para o propósito deste trabalho estaremos sempre nos referindo a API REST [3].

Segundo a AWS, REST, em português, significa transferência representacional de estado e sua principal característica é ter ausência de estado. Isso significa que nas requisições e transferências de dados não há a coleta e armazenamento de dados do cliente entre as solicitações. Esta API pode ser definida ainda, por sua estrutura já que segue as normas do protocolo HTTP e as trocas de dados são realizadas por funções como GET, PUT e DELETE.

Entre os principais benefícios da API REST estão a inovação, com a possibilidade de mudança e otimização de setores inteiros. Expansão, com a possibilidade de aumentar o alcance de seus serviços. Facilidade de manutenção. E, o que nos levou a usá-la neste projeto, integração, possibilitando o aproveitamento de código já existente [3].

2.4. Padrão Arquitetural MVC.

O MVC é um padrão de arquitetura de software. Sua sigla significa Model (Modelo) View (Visão), Controller (Controle) [4].

MVC contribui na otimização da velocidade entre as requisições. Ela é dividida nos três componentes que formam seu nome, Modelo, Visão e Controle. Ela é bastante popular na criação de aplicações para web. Seu grande objetivo é isolar ao máximo a camada de apresentação [5].

- **Model:** é onde ficam os dados do aplicativo, nele não ficam nenhuma lógica da aplicação, não é nele que descreve como os dados armazenados serão apresentados para o usuário. Ela basicamente tem a função de gerenciar e controlar o comportamento dos dados.
- **View:** Responsável pela apresentação das informações ao usuário, ela é o frontEnd da aplicação. O View é responsável com a comunicação com o usuário e transmite suas requisições ao controller, e é responsável por apresentar as respostas que Controller recebeu da camada Model.
- **Controller:** Camada responsável por intermediar as requisições vindas de View com as respostas do Controller. Ele recebe os eventos que foram disparados pela camada View, e executa uma resposta para a requisição. Geralmente é executar um método na camada Model. A camada Controller pode ser vista como uma “ponte” que conecta a camada View e a Camada Controller.

O padrão MVC ajuda a quebrar o código de front-end e back-end em componentes separados. Dessa forma, torna-se mais intuitivo gerenciar e fazer alterações em qualquer um dos lados sem que interfiram entre si. O MVC torna a gestão mais simples, mas não perfeita, especialmente quando vários desenvolvedores precisam atualizar, modificar ou depurar um aplicativo completo ao mesmo tempo.

Em MVC, o ponto chave é: a separação de interesses. Os aplicativos da web modernos são muito complexos e, às vezes, fazer uma mudança pode ser um grande problema — quanto maior o navio, mais difícil manobrar. Gerenciar o front-end e o back-end em componentes menores e separados permite que o aplicativo seja escalonável, sustentável e fácil de expandir. E com muito menos dor

de cabeça. Alguns benefícios são Aumento de produtividade; Uniformidade na estrutura do software; Redução de complexidade no código; As aplicações ficam mais fáceis de manter; Facilita a documentação; Estabelece um vocabulário comum de projeto entre desenvolvedores; Permite a reutilização de módulos do sistema em outros sistemas; É considerada uma boa prática utilizar um conjunto de padrões para resolver problemas maiores que, sozinhos, não conseguiriam; Ajuda a construir softwares confiáveis com arquiteturas testadas; Reduz o tempo de desenvolvimento de um projeto [6].

Graças as facilidades para o desenvolvimento de aplicações o padrão MVC passou a ser adotado por diversos frameworks e aplicados em diversos tipos de projetos, onde no desenvolvimento web esse se tornou mais popular. Como o MVC não é um conceito de linguagem mas sim de arquitetura ele pode ser utilizado em qualquer linguagem de programação [9].

É importante que todo desenvolvedor tenha conhecimento sobre o MVC, pois ele é amplamente utilizado e difundido pelo mercado. Também é interessante conhecer outros patterns baseados no MVC e que são utilizados com frequência no mercado, como o MVM e o MVP.

3 METODOLOGIA

A metodologia usada pelo grupo do projeto de pesquisa passa por várias etapas que devem ser apresentadas na ordem com a qual foram planejadas e, posteriormente, executadas.

1. Definição do escopo do projeto;
2. Definição dos critérios para escolha dos discentes envolvidos;
3. Condução do processo seletivo dos discentes;
4. Apresentação do cronograma de atividades;
5. Condução de treinamentos para alinhamento da equipe;
6. Estudos dirigidos às tecnologias necessárias para a área de integração de software;
7. Implementação de estudos de casos (softwares com as mesmas tecnologias usadas na empresa parceira) visando a integração;
8. Implementação das integrações reais demandadas pela empresa parceira;

Para a definição do escopo do projeto, tópico 1, foi conduzida uma reunião entre o discente responsável pelo projeto e o profissional da empresa parceira, responsável pela interação com o grupo de pesquisa. Nesse sentido, ficou alinhado que o projeto envolveria a escolha de discentes para participar do projeto, bem como suas atribuições individuais e coletivas. Posteriormente, no tópico 2, foram criados critérios objetivos que visavam pautar a seleção dos discentes, contemplando o perfil desejado para as tarefas. Vários candidatos qualificados se apresentaram e, dentre eles, formou-se o grupo de trabalho.

A condução do processo seletivo, tópico 3, se deu com entrevistas onde se conheceu os candidatos, apresentando a eles o escopo do projeto. Na sequência, o calendário com o cronograma de atividades foi apresentado (tópico 4), separando entre os envolvidos as atividades que cada um iria desempenhar. Nos dias que se sucederam, treinamentos específicos foram conduzidos com a equipe (tópicos 5 e 6). Isso envolveu apresentações de resultados voltados para as tecnologias que seriam necessárias durante todo o projeto de pesquisa. Dentre elas pode-se destacar: (i) Orientação a Objetos; (ii) Desenvolvimento C#; (iii) Implementação de bases de dados SQL Server; (iv) desenvolvimento com APIs e (v) Frameworks de persistência.

As implementações de estudos de caso (tópico 7) visava basicamente a criação de aplicações de software, simuladas de contextos reais, que permitiam com que a equipe pudesse conduzir, em ambiente de desenvolvimento, experimentos de implementação de aplicações de software que pudessem ser integradas. Neste momento, duas aplicações foram planejadas para serem executadas: (i) um sistema web que simulava (em partes) implementações e telas da plataforma iFood e (ii) um sistema de propósito semelhante, com implementações e telas do Robin Food.

Por fim, no tópico 8, tem-se por meta o desenvolvimento de integrações entre as aplicações criadas (iFood e Robin Food). Para tal, foram usadas APIs diferentes para manipulação e leitura das bases de dados, bem como uma camada de middleware. Este tópico será detalhado na seção resultados e discussões.

4 RESULTADOS E DISCUSSÕES

Ao longo deste projeto de pesquisa, diferentes tipos de resultados foram obtidos. O primeiro deles foi descrito na seção que tratou da metodologia. Lá foi dito

que após o processo seletivo dos discentes, treinamentos e atividades de alinhamento foram realizados. Particularmente, este tipo de ação gera como resultado uma melhor capacitação dos envolvidos. Com isso, foi possível alçar desafios maiores, no que tange a implementações consideradas complexas para o perfil dos discentes, na sua maioria, de períodos iniciais da graduação, que não tiveram contato com o conhecimento necessário neste ponto, em alguns casos.

Na sequência, pode-se aqui descrever resultados e discussões gerados no âmbito das integrações realizadas. A primeira a ser destacada foi a discussão em torno de quais tipos de aplicação deveriam ser implementadas. A decisão levou a implementação de duas aplicações simuladas: Robin Food e iFood. A Figura 1 mostra as telas de cadastro de dados das aplicações, bem semelhante à tela de cadastro do Robin Food e do iFood.

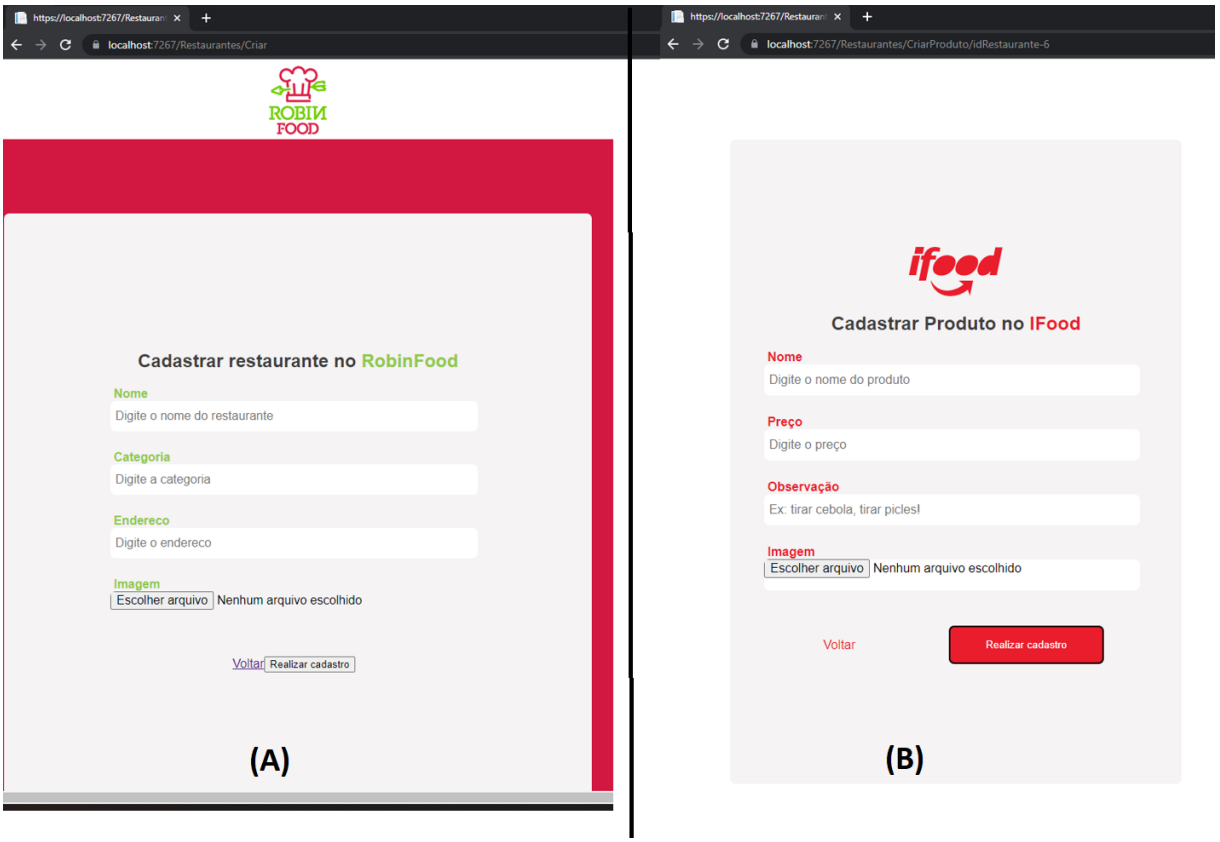

Figura 1. Telas de cadastro de restaurante criadas pelo grupo, simuladas das aplicações reais.

Com os dados cadastrados nas diferentes aplicações (Robin Food - A e iFood - B) foi possível integrar diferentes dados. Isso se deu graças as implementações de

APIs em cada uma das aplicações. Ambas são responsáveis por gravar dados de restaurantes e pedidos, via API. em bases de dados na nuvem.

Após ter-se as duas aplicações simuladas implementadas, com seus devidos mecanismos de persistência a manipulação de APIs, partiu-se para o desenvolvimento de uma aplicação que é responsável por fazer a camada Middleware. A Figura 2 ilustra os comandos básicos dessa camada.

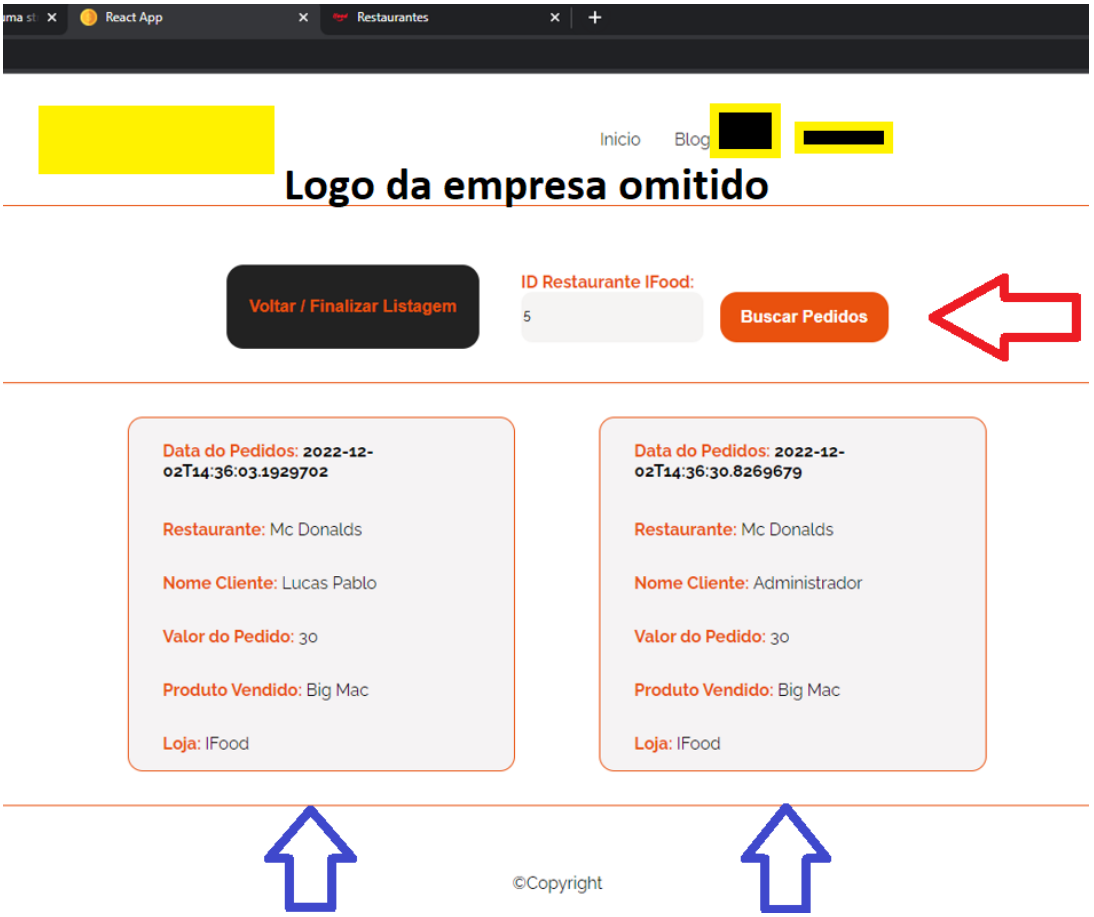

Figura 2. Uma das telas da aplicação Middleware, que implementa as diferentes APIs.

Na Figura 2 é possível ver uma das telas da camada de Middleware, que, mediante a escolha da plataforma integrada (Robin Food ou IFood), invoca-se a API responsável (ou do Robin Food ou do iFood) para buscar dados na sua respectiva base de dados, na nuvem (Azure). Além de buscar dados via API específica de cada plataforma integrada, pode-se ainda invocar uma terceira API que, por sua vez, invocará as APIs específicas das plataformas, combinando consigo os dados gerados. Essa integração é possível via a comunicação da camada Middleware

(Figura 2) com as APIs de cada plataforma integrada (Robin Food ou IFood), gerando informação combinada e nova. A Figura 3 mostra uma tela com dados que foram obtidos através da integração entre as APIs que compõem a camada middleware. Cada quadrado cinza representa um conjunto de dados cadastrado numa base de dados diferente. A seta azul indica dados de pedido que vieram, via API, da base de dados dos produtos da plataforma simulada iFood. Já o sinalizado com seta vermelha, por exemplo, indica dados de pedido que vieram, via API, da base de dados dos produtos da plataforma simulada Robin Food.

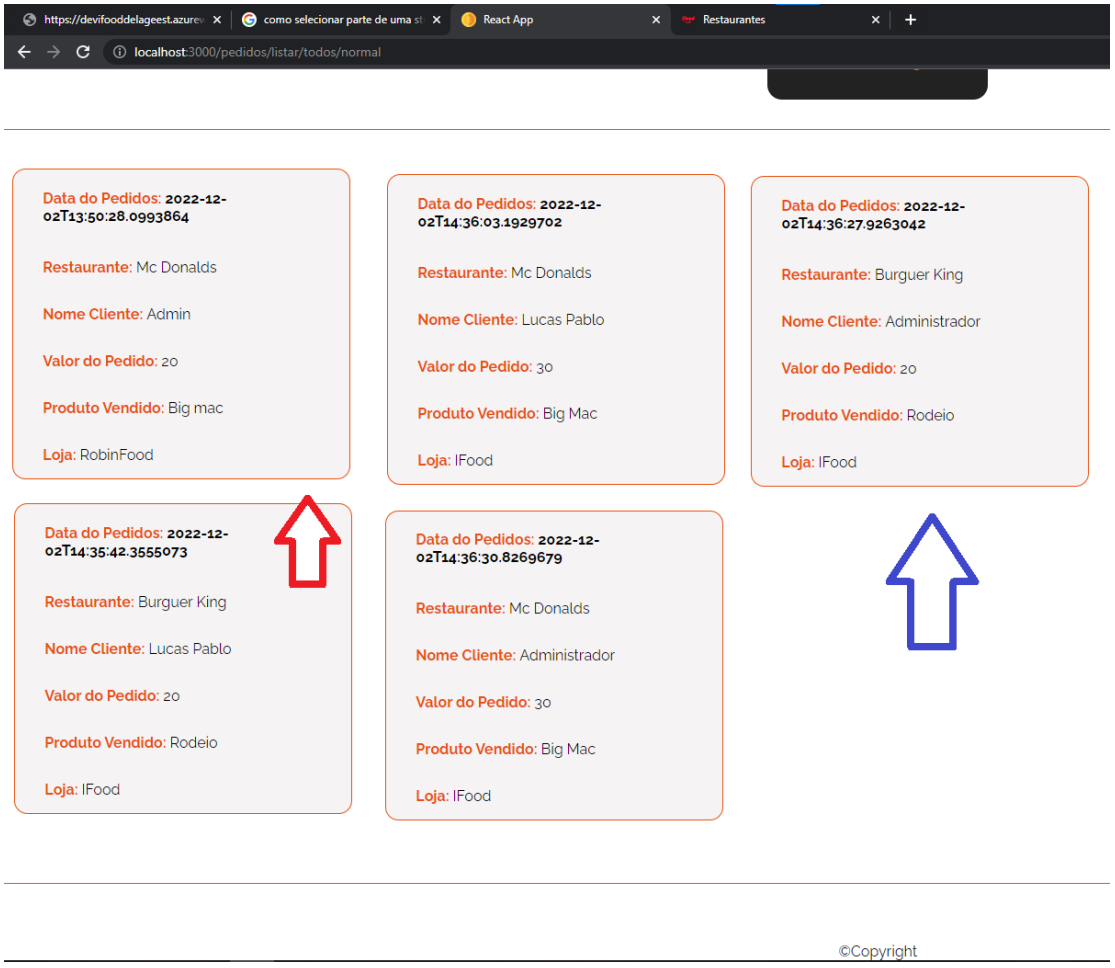

Figura 3. Tela que mostra os dados recuperados na comunicação das 3 APIs, no middleware.

Para finalizar, a Figura 4 mostra uma visão geral de como se deu a arquitetura completa desenvolvida até este momento, no projeto de pesquisa.

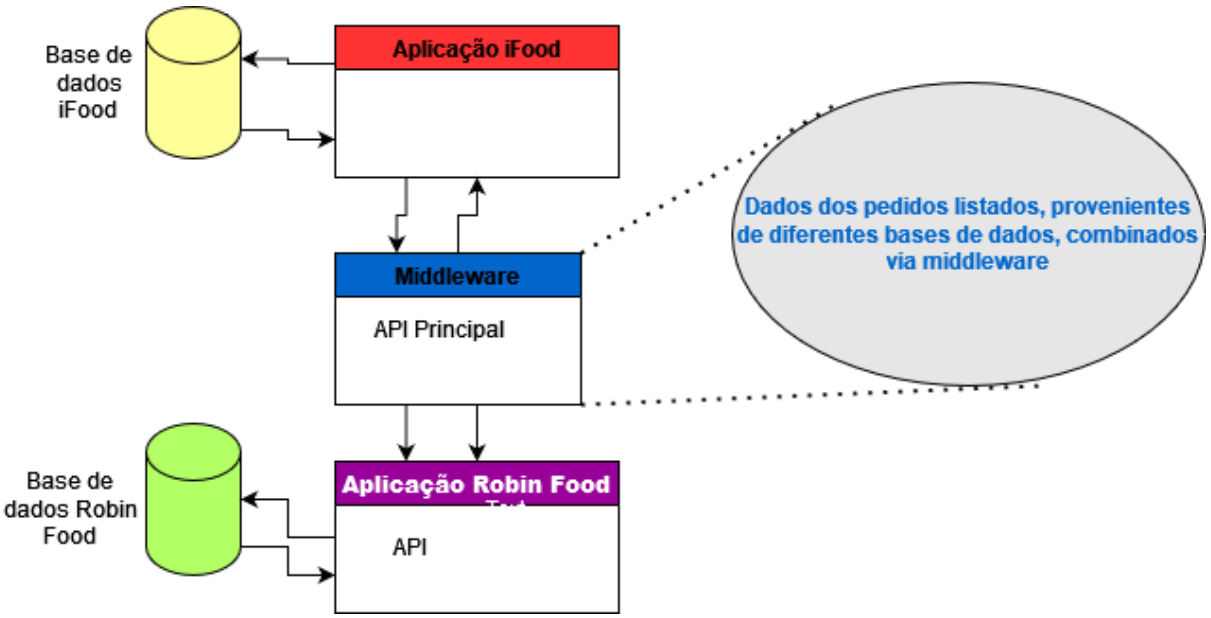

Figura 4. Arquitetura geral das integrações realizadas até o momento.

Vale ressaltar que todas as 3 aplicações (as simuladas iFood e Robin Food + a aplicação middleware) foram desenvolvidas usando as tecnologias React, Microsoft C#, Azure e APIs de comunicação.

5 CONSIDERAÇÕES FINAIS

Neste momento, o projeto de pesquisa avança nas etapas de integração das tecnologias desenvolvidas e no entendimento das tecnologias usadas pela empresa parceira. Na última apresentação de relatório do andamento dos trabalhos do grupo, esperava-se, ao final do prazo previsto para o projeto, a integração de aplicações usando as nuvens Azure, Middlewares desenvolvidos pela empresa parceira e demais tecnologias. Agora, as etapas descritas acima já foram vencidas, como pode ser visto na seção resultados e discussões. Neste sentido, como trabalhos futuros, espera-se desenvolver diferentes abordagens de integração entre os diferentes tipos de aplicações reais do contexto da empresa parceira. Isso envolve diferentes tipos de aplicações comerciais que necessitam ter seus dados integrados, para gerar inteligência ao negócio.

Até este momento, este grupo de pesquisa foi capaz de se preparar para os desafios dos cenários reais, ao usar as mesmas tecnologias que a empresa parceira utiliza. Como trabalhos futuros, espera-se, nas próximas etapas de renovação do

projeto de pesquisa, desenvolver integrações de cenários reais, com dados reais do negócio, diferentemente dos experimentos e simulações estabelecidos em ambiente de desenvolvimento.

Agradecimento à Uniacademia, por formar parcerias que viabilizem aos alunos experiências profissionais mesmo estando ainda eles na graduação. Agradecimento à empresa parceira, empresa de TI de JF, pela cooperação e financiamento dos envolvidos no projeto.

ABSTRACT

It is known that over time large software projects have gained market space by proposing more and more new possibilities for their users, such as, for example, the connection of different software components making the new configuration bring news, in terms of of resources and data. This is called software integration. The function of the person responsible for integrations is not limited to just the act of integrating processes, but rather the participation in everything that permeates the process of developing new technologies. This research project aims to enable new integrations between existing platforms, thus allowing new features unexplored in integrated technologies

Keywords: Software Integration, Corporate Environment.

REFERÊNCIAS

[1] CIn UFPE. https://www.cin.ufpe.br/~gta/rup-vc/core.base_rup/guidances/concepts/software_integration_2F85C9B0.html. Acessado em: 20/10/2022.

[2] Tavares, Juliana da Silva Zafalão. Elaboração de um Framework para Integração de Software. Marília, SP, 2012.

[3] O que é uma API?. **AWS,** 2022. Disponível em: <https://aws.amazon.com/pt/what-is/api/>. Acessado em 02 de dez. de 2022.

[4] O que é o padrão MVC? Disponível em : https://www.lewagon.com/pt-BR/blog/o-que-e-padrao-mvc. Acesso em 02 de dez. de 2022.

[5]O que é o MVC? Disponível em: https://tecnoblog.net/responde/o-que-e-mvc/. Acessado em 02 de dezembro 2022

[6] MVC. Disponível em: https://www.treinaweb.com.br/blog/o-que-e-mvc. Acessado em 02 de dezembro de 2022

[7] Whats is Middleware? : Disponível em: https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-middleware/. Acessado em 02 de dezembro de 2022.

[8] Whats is Middleware? Disponível em: https://www.redhat.com/en/topics/middleware/what-is-middleware. Acessado em 02 de dezembro de 2022.

[9] Josué Luciano, Wallison Joel Barberá Alves. PADRÃO DE ARQUITETURA MVC: MODEL-VIEW-CONTROLLER.. Bebedouro, São Paulo (2011).