TRABALHO COMPLETO
FULL PAPER
XXVII SEPE
Simpósio de Ensino, Pesquisa e Extensão
Teaching, Research and Extension Symposium
24 A 26 DE OUTUBRO 2023
OCTOBER 24th to 26th, 2023
ISSN: 2316-9745
CIÊNCIA E DESENVOLVIMENTO SUSTENTÁVEL
SCIENCE AND SUSTAINABLE DEVELOPMENT
UFN
Universidade Franciscana
https://doi.org/10.48195/sepe2023.27350

SISTEMA WEB PARA GESTÃO DE PERFIS EM AMBIENTES AUTOMATIZADOS

Marcelo de Siqueira[1]; Ana Paula Canal[2]; Alessandro Mainardi de Oliveira[3]; Guido Ruviário[4]; Alexandre de Oliveira Zamberlan[5]

RESUMO

Este trabalho visa complementar o sistema projetado e implementado por Aloísio Kneipp dos Santos, nominado de Sistemas Pervasivos Integrados por Agentes Inteligentes em JASON e Raspberry Pi, via o projeto e a implementação de um sistema Web para gestão de usuários e seus perfis em ambientes, como cômodos em uma casa, no que se refere a preferências de iluminação e temperatura. O sistema Web integra dados de preferências e de usuários na simulação criada por meio de um *Web Service*. Para isso, as tecnologias Python, Django e Bootstrap são utilizadas para a construção dessa plataforma. Os resultados gerados são modelagem funcional e estrutural do sistema, além de alguns protótipos de tela para gestão de usuários, cômodos, perfis, entre outros.

Palavras-chave: Sistema de Informação; Python; Django.

ABSTRACT

This research aims to complement the system designed and implemented by Aloísio Kneipp dos Santos, called Pervasive Systems Integrated by Intelligent Agents in JASON and Raspberry Pi. Therefore, we intend to design and implement a Web system for managing users and their profiles in environments, such as rooms in a house, with regard to lighting and temperature preferences. The Web system integrates preference and user data into the simulation created through a Web Service. To achieve this, Python, Django and Bootstrap technologies are used to build this platform. The results generated are functional and structural modeling of the system, in addition to some screen prototypes for managing users, rooms, profiles, among others.

Keywords: Information Systems; Python; Django.

Eixo Temático: Tecnologia, Inovação e Desenvolvimento Sustentável (TIDS).

[1] Autor/Apresentador - UFN - Curso de Sistemas de Informação - marcelo.siqueira@ufn.edu.br
[2] Professor - UFN - Cursos Ciência da Computação e Sistemas de Informação - apc@ufn.edu.br
[3] Professor - UFN - Cursos Ciência da Computação e Sistemas de Informação - alessandroandre@ufn.edu.br
[4] Aluno - UFN - Curso de Direito - ruviario.guido@ufn.edu.br
[5] Orientador - UFN - Cursos Ciência da Computação e Sistemas de Informação - alexz@ufn.edu.br

ISSN: 2316-9745
CIÊNCIA E DESENVOLVIMENTO SUSTENTÁVEL
SCIENCE AND SUSTAINABLE DEVELOPMENT
UFN Universidade Franciscana Franciscan University

1. INTRODUÇÃO

A simulação criada por Santos e Zamberlan (2021) tenta avaliar a relação de usuários e suas preferências (iluminação e temperatura) em cômodos de uma casa, por exemplo. No trabalho, foi realizada a integração da teoria BDI (*Belief, Desire, Intention*) de agentes inteligentes com os fundamentos da Computação Pervasiva, no contexto da Internet das Coisas (do inglês, *Internet of Things* - IoT), em que dispositivos pudessem comunicar-se entre si de forma inteligente, autônoma, proativa e flexível. O trabalho foi composto de uma simulação de solução computacional que forneceu integração entre dispositivos Raspberry Pi e a tecnologia de Sistemas Multiagentes implementados em JASON. Contudo, a pesquisa feita trouxe como trabalhos futuros, a responsabilidade de gerenciar usuários, ambientes e perfis específicos dos usuários em ambientes (SANTOS; ZAMBERLAN, 2021). Nesse sistema, deveria ser possível associar usuário a uma residência ou a um ambiente comercial e subdividi-lo em salas ou unidades. Cada sala, o usuário poderia especificar um perfil de climatização e de iluminação, tendo como parâmetros intensidade da luz, temperatura, entre outros.

Dessa forma, o objetivo geral deste trabalho é projetar, desenvolver e implantar um sistema Web para gerenciar usuários, ambientes e perfis específicos dos usuários. Para que o objetivo geral seja alcançado, identificaram-se alguns objetivos específicos: entender e testar a simulação criada por Santos e Zamberlan (2021), bem como todos os processos modelados; investigar como integrar usuários e seus perfis no sistema Web no ambiente de simulação criado (*Web Service*); entender e testar o desenvolvimento de sistemas Web via tecnologia Python-Django-Bootstrap; mapear e compilar trabalhos relacionados.

2. REVISÃO BIBLIOGRÁFICA

Esta seção trata dos principais fundamentos e processos trabalhados no texto. Inicialmente, apresenta os temas foco da pesquisa e na sequência as tecnologias necessárias para a construção da solução.

2.1 COMPUTAÇÃO EM NUVEM, INTERNET DAS COISAS, AUTOMAÇÃO E WEB SERVICE

É fato que muitos sistemas já executam via a Internet, como sistemas (*online*) distribuídos geograficamente. Dessa forma, esses sistemas estão em nuvem, pois executam em navegadores ou em aplicativos móveis. Esse conceito de Computação em Nuvem, ou *Cloud Computing*, refere-se ao processamento computacional e armazenamento de dados que são executados fora da infraestrutura local (COSTA; ZAMBERLAN, 2021).

Do inglês, *Internet of Things* (IoT), trata-se de um ecossistema ou uma rede de objetos (coisas ou dispositivos), que coleta, processa e compartilha dados via protocolos de comunicação da Internet (modelo TCP/IP) (GODOI; ARAUJO, 2019). Isso se dá por meio de sensores e atuadores instalados nos ambientes, que monitoram dados e executam ações, respectivamente. A Internet das Coisas pode ser utilizada para uso empresarial, doméstico, industrial, nos processos de tomada de decisão, entre outros.

Web Service é considerado um padrão de comunicação e integração de sistemas, permitindo que diferentes aplicações independente de plataforma compartilhem e troquem dados pela Internet (COSTA; ZAMBERLAN, 2021). Portanto, é uma combinação de hardware e software, que possibilita que aplicações (homogêneas e/ou heterogêneas) possam trocar dados entre si, de maneira natural e transparente, sem a necessidade de protocolos complexos de comunicação. Há diferentes formas de se construir *Web Service*, mas em geral Extensible Markup Language (XML), JavaScript Object Notation (JSON) e Comma-Separated Values (CSV) são as mais utilizadas.

2.3 FRAMEWORKS, PYTHON, DJANGO E BOOTSTRAP

Um *framework* é uma ferramenta que funciona como facilitador ou agilizador no momento de desenvolver um software. Ele é formado por um conjunto de pacotes, classes e métodos implementados fornecendo recursos prontos para que não seja necessário desenvolver uma funcionalidade ou uma estrutura do zero. Dessa forma, diminuindo tarefas repetitivas utilizando de códigos disponíveis e já testados (consolidados) (PRESSMAN; MAXIM, 2016).

Django é um *framework* gratuito para desenvolvimento rápido e seguro de aplicações na Web escrito em Python. Fornece inúmeros modelos de *design*, *drivers* de conexão com banco de dados, de processos de validação (como *login* e senhas), de aplicativos com CRUD (*Create, Retrieve, Update, Delete*) completo. Trabalha no modelo MVT (*Model-View-Template*). Model: responsável pelo mapeamento do banco de dados via classes e objetos do sistema; Template: interface (página HTML - visual) que o usuário vai interagir com a aplicação; View: parte lógica do sistema, onde se especifica a interação da parte visual com o modelo de dados, ou seja, as regras do negócio. Além dessas características, Django inclui vários recursos e funcionalidades prontas que são facilmente incorporadas à aplicação, reduzindo consideravelmente o tempo de desenvolvimento e as linhas de código (DJANGO, 2023).

2.4 NGINX, GUNICORN E SQLITE

Uma arquitetura em nuvem é formada por múltiplos servidores. Entre os vários conceitos que abrangem a computação em nuvem destacam-se balanceamento de carga, banco de dados aglomerados e escalonamento de armazenamento. Nesse quesito de balanceamento de carga, surge então o serviço NGINX, que faz a distribuição de requisições de maneira equitativa sobre os nós de um ambiente distribuído (SANTOS *et al.*, 2016). Em síntese, o NGINX é em uma aplicação Web o responsável por gerenciar a disponibilidade da aplicação ao usuário. Além disso, esse serviço oferece certificado digital SSL (*Secure Sockets Layer*), permitindo que o tráfego entre o cliente final e o servidor seja utilizando conexão segura através do protocolo HTTPS (*Hypertext Transfer Protocol Secure*). O NGINX também foi projetado para servir e fazer cacheamento de arquivos estáticos como imagens, CSS (*Cascading Style Sheets*) e Javascript, além de servir o conteúdo dinâmico encaminhado pelo GUNICORN. Por sua vez, GUNICORN é considerado um servidor Python para páginas e/ou sistemas via protocolo HTTP para sistemas UNIX (Linux). Ou seja, ele resolve de forma estática e dinâmica as solicitações HTTP de clientes via seus navegadores. Esse serviço é compatível com várias estruturas Web, com uso de recursos otimizados (SANTOS *et al.*, 2016).

De acordo com o site de referência (SQLITE, 2023), SQLite é um Sistema de Gerenciamento de Banco de Dados (SGBD) relacional, porém simples, portável e leve. Além disso, não exige uma configuração complexa ou um servidor separado para sua operação. É o SGBD embutido no *framework* Django, muito útil para o processo de desenvolvimento.

2.5 SCRUM

De acordo com Sutherland (2016), Scrum é uma estrutura ágil de gerenciamento de projetos criada no início dos anos 90, considerada um *framework* flexível e funcional, não apenas para desenvolvimento de software, como também útil para qualquer trabalho em equipe que necessite ser gerido desta forma. Na prática, o Scrum funciona da seguinte forma: mantém-se uma programação regular de pequenas reuniões (*sprints*), com a finalidade de inspecionar o projeto em andamento, descobrir possíveis falhas ou erros, a fim de corrigi-los com agilidade e fazer entregas ao cliente sempre no prazo combinado.

3. METODOLOGIA E PROPOSTA DE TRABALHO

Este trabalho é uma continuação de um estudo de caso, com desenvolvimento de um sistema Web e um sistema de integração (*Web Service*), tendo como referência a pesquisa realizada por Santos e Zamberlan (2021). Já no projeto e desenvolvimento do Sistema Web, são utilizados a metodologia Scrum (SUTHERLAND, 2016) com a técnica Kanban para gestão de atividades, prazos e responsáveis. As ferramentas utilizadas são: ambiente de aplicação da técnica Kanban: Trello; ambiente de diagramação *Unified Language Model* (UML): Astah; linguagem de programação Python e seus frameworks Bootstrap e Django para o sistema Web; serviços GUNICORN e NGINX em sistema operacional Linux; sistema gerenciador de banco de dados: SQLite; ambiente de versionamento de código Github; ambiente de desenvolvimento: Visual Studio Code e suas extensões Python-Django.

Um diagrama importante em relação aos aspectos funcionais, é o diagrama de atividades, que ilustra o fluxo de trabalho pretendido com o sistema proposto (Figura 1). Nesse diagrama, foi apresentado a forma como a simulação faz a relação

ou comunicação com o sistema proposto. Destaca-se que a simulação vai continuar independente do sistema. A Figura 2 ilustra os atores e as principais funcionalidades do sistema. O diagrama apresenta três pacotes: sistema pervasivo proposto por Santos e Zamberlan (2021), que é a simulação projetada, implementada e executada; o ambiente de simulação construído na ferramenta JASON; sistema proposto, em que vale ressaltar que o componente *Web Service* presente no outro pacote, faz parte deste trabalho. Buscou-se ilustrar as funcionalidades para gestão de usuários, cômodos e perfis, sendo que ao final, os dados cadastrados estarão disponíveis ao simulador via o *Web Service*. A ideia inicial é gerar conteúdo ao *Web Service* sempre que uma simulação for iniciada.

Uma vez definidas e especificadas as funcionalidades, segue a especificação de aspectos estruturais. Um diagrama importante para isso é o de Classe mas com uma característica de componentes, ilustrando componentes e/ou classes necessários (Figura 3). No diagrama de pacotes, é possível destacar as classes. Usuário: representando os usuários de um cômodo; Cômodo: com dados para descrever e identificar a peça em uma casa ou escritório; Equipamento: representando um equipamento como ar condicionado ou como sistema de iluminação; UsuárioCômodo: representa a relação do usuário com um cômodo, ou seja, qual o nível de importância que um determinado tem sobre um cômodo. Isso é importante, pois quando mais de um usuário entrarem no cômodo, o simulador precisa entender que tem prioridade de escolha; Perfil: representa os desejos de um usuário sobre um cômodo, separado por estação do ano, temperaturas para os três turnos e intensidade de iluminação, também para os três turnos; Rele: são os dados do relé de acionamento que recebe ações do simulador para disparar algum equipamento.

Por exemplo, um usuário, tem preferências de temperatura e iluminação para a sua sala de jantar, nas quatro estações, além de cada turno do dia, com temperatura específica e intensidade de iluminação para cada turno.

Figura 2: Diagrama de Atividade.

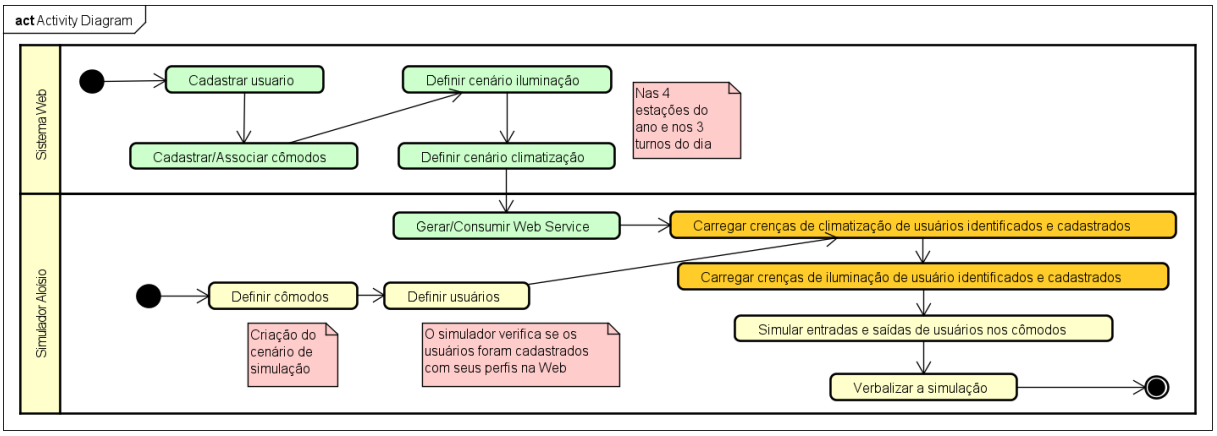


Figura 2: Diagrama de Casos de Uso ilustrando tanto as funcionalidades, quanto a ideia geral da arquitetura da integração do simulador com o sistema Web.

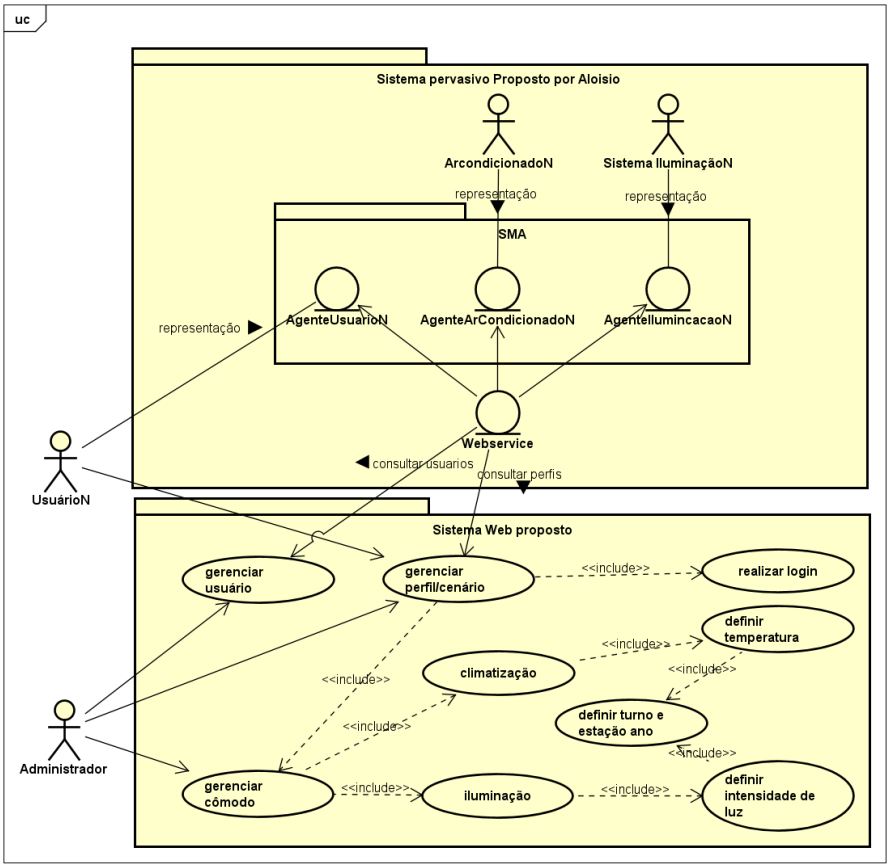
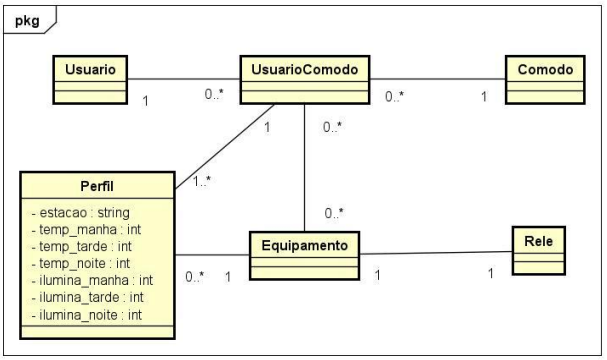


Figura 3: Diagrama de Classe.



O processo de simulação criado (SANTOS; ZAMBERLAN, 2021), gera uma dinâmica (via JASON) em que usuários entram e saem de um ambiente (sala ou quarto, por exemplo) e que esses ambientes percebem (via sensores) esses usuários, capturando seus perfis, com preferências para climatização e/ou iluminação, atuando (via atuadores) nos sistemas de iluminação e climatização, configurando assim temperatura e intensidade de iluminação. Diferente do que ocorria na simulação, em que os usuários e seus perfis eram previamente cadastrados na simulação (no ambiente do JASON), agora, os usuários e seus perfis estão cadastrados no sistema Web proposto e são compartilhados ou consumidos (seus dados) com a simulação via um *Web Service*. A ideia do fornecimento dos dados dos usuários ao simulador é via *Web Service*, que é gerado de tempos em tempos pelo sistema Web e consumido pelo simulador.

Os resultados iniciais do trabalho referem-se à modelagem dos aspectos funcionais e estruturais do sistema proposto. Os aspectos funcionais podem ser visualizados em Diagramas de Atividades e de Casos de Uso. Quando questões funcionais são modeladas ou mapeadas, destacam-se os atores que fazem relação com o sistema, as funcionalidades ou serviços que o sistema deve atender e o fluxo de funcionamento ou de interação do sistema com esses atores. Também foi modelado um aspecto estrutural, via diagrama de componentes (classe), em que foi possível identificar as classes, logo tabelas, que o sistema Web deve tratar. Como o sistema está sendo construído via a linguagem Python e o *framework* Django, cada uma dessas classes deve ser um aplicativo (app) do sistema, contendo as três camadas do modelo MVT: model (classes e futuramente tabelas); view (todas as regras do negócio, em especial os métodos Create, Retrieve, Update, Delete -

CRUD); template (arquivos HTML e CSS de comunicação com o usuário). Também é possível registrar como resultado inicial, algumas telas do sistema. A Figura 4 apresenta a tela de *login* combinada com uma *landpage*, contendo informações do projeto. Na Figura 5, é possível visualizar a página principal do sistema, bem como o menu de serviço, com a possibilidade de gestão de usuários, cômodos, equipamentos, perfis e relés.

Figura 4: Login e Landpage do sistema.

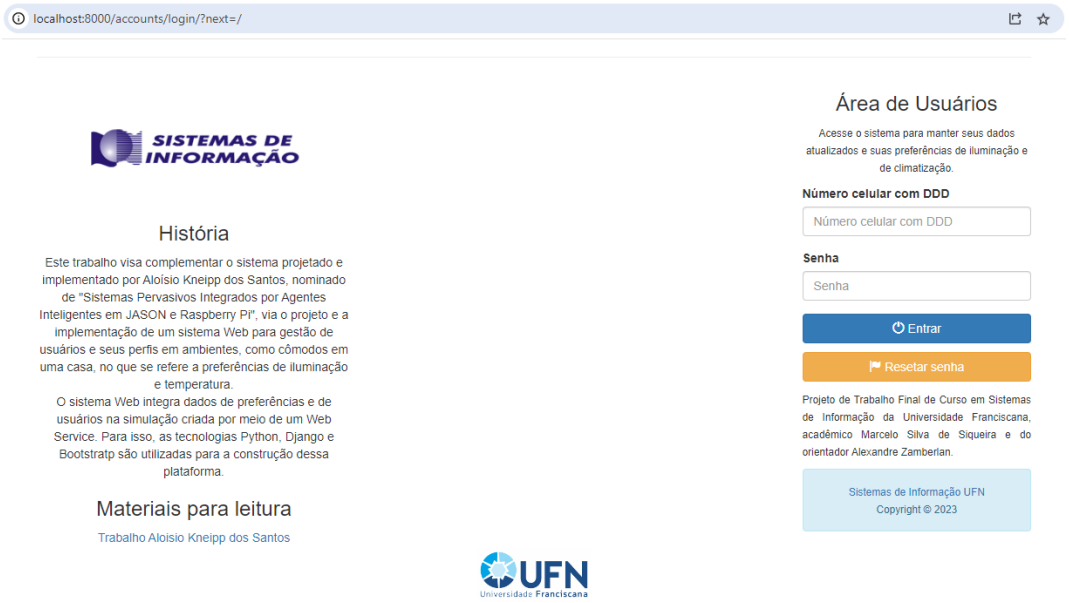
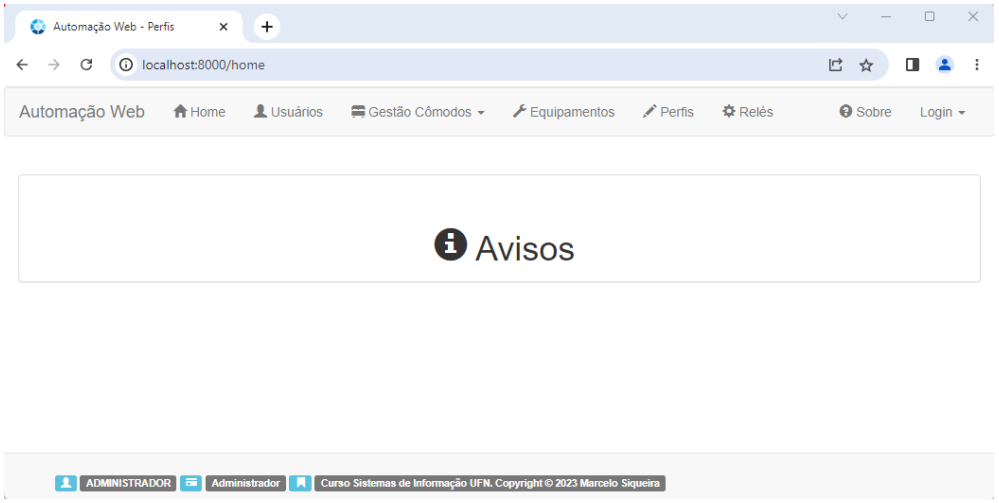


Figura 5: Página principal do sistema, após *login*.



4. CONSIDERAÇÕES

Ao longo do texto, foram discutidos os principais assuntos que norteiam este trabalho, como Computação em Nuvem, IoT, *Web Service*, \textit{Frameworks}, os serviços NGINX e GUNICORN, a metodologia Scrum e uma avaliação dos trabalhos relacionados. Na sequência, modelou-se o sistema para gestão de perfis em ambientes automatizados, tendo como base a simulação realizada por Santos e Zamberlan (2021). Dessa modelagem, destacam-se diagramas dos aspectos funcionais e estruturais, que dão suporte para o entendimento de como o sistema deve ser construído: quais seus atores, quais funcionalidades, quais componentes, como classes, etc.

REFERÊNCIAS

COSTA E. G.; ZAMBERLAN, A. Webservice para integração de dados entre a ferramenta de simulação MASPN e seu portal Web. Santa Maria, RS, Brasil. Disponível em https://tfgonline.lapinf.ufn.edu.br: Trabalho de Conclusão de Curso Sistemas De Informação - Universidade Franciscana (UFN), 2021.

DJANGO. Disponível em: https://www.djangoproject.com/. Acessado em junho de 2023

GODOI, M. G. D.; ARAÚJO, L. S. A internet das coisas: evolução, impactos e benefícios. Revista interface tecnológica, v. 16, n. 1, p. 19–30, 2019.

PRESSMAN, R.; MAXIM, B. Engenharia de Software-8ª Edição. Rio de Janeiro: McGraw Hill Brasil, 2016.

SANTOS, A. K. D.; ZAMBERLAN, A. Sistemas pervasivos integrados por agentes inteligentes em Jason e Raspberry Pi. Santa Maria, RS, Brasil. Disponível em https://tfgonline.lapinf.ufn.edu.br: Trabalho de Conclusão de Curso Ciência Da Computação - Universidade Franciscana (UFN), 2021.

SANTOS, R. E. *et al*. Proposta de uma plataforma de cloud computing para disponibilização de um sistema online para consultórios e clínicas por meio do modelo SaaS. Tópicos em Ciências da Saúde. Volume 24, p. 7, 2021.

SUTHERLAND, J. Scrum: a arte de fazer o dobro do trabalho na metade do tempo. São Paulo: Leya, 2016.