UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

ITALO MODESTO PEREIRA

SISTEMA MOBILE DE AGENDAMENTO DE HORÁRIOS PARA USO DO COMPLEXO ESPORTIVO DA UFCG

CAMPINA GRANDE - PB

2023

ITALO MODESTO PEREIRA

SISTEMA MOBILE DE AGENDAMENTO DE HORÁRIOS PARA USO DO COMPLEXO ESPORTIVO DA UFCG

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientadora: Professora Dra. Melina Mongiovi Brito Lira.

CAMPINA GRANDE - PB

2023

ITALO MODESTO PEREIRA

SISTEMA MOBILE DE AGENDAMENTO DE HORÁRIOS PARA USO DO COMPLEXO ESPORTIVO DA UFCG

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Professor Dr.(a.) Melina Mongiovi Brito Lira

Orientador – UASC/CEEI/UFCG

Professora Dr.(a.) Marcus Salerno De Aquino

Examinador – UASC/CEEI/UFCG

Professor Tiago Lima Massoni

Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 14 de fevereiro de 2023.

CAMPINA GRANDE - PB

ABSTRACT

The scheduling appointments to use any of the spaces in the sports complex of the Federal University of Campina Grande is done in person, at the complex's secretariat, or using whatsapp. Both strategies are uncomfortable for the user, since in the first one, the user needs to go to the secretariat and in the second one, the user needs to wait for an employee of the administrative part of the complex to be available, and only then, answer him. With the aim of presenting a solution that makes the scheduling process more practical, this work documents the development of an android application, with versions for users and administrators of the complex, which makes scheduling much simpler, since it can be done from any location, through the application and that the user does not need to wait for an employee of the complex's secretary to be available to answer it. As a result, the application that allows scheduling of spaces in the sports complex is presented, in addition to its validation with users through questions made in the form format, after the presentation of the application, and suggestions for future work.

Sistema Mobile de Agendamento de Horários Para Uso do Complexo Esportivo da UFCG

Italo Modesto Pereira
Universidade Federal de Campina Grande
Campina Grande, Paraíba, Brasil
italo.pereira@ccc.ufcg.edu.br

Melina Mongiovi Brito Lira
Universidade Federal de Campina Grande
Campina Grande, Paraíba, Brasil
melina@computacao.ufcg.edu.br

RESUMO

O agendamento de horários para uso de algum dos espaços do complexo esportivo da Universidade Federal de Campina Grande é feito presencialmente, na secretaria do complexo, ou via whatsapp. As duas estratégias são desconfortáveis para o usuário, visto que na primeira, o usuário precisa deslocar-se até a secretaria e na segunda, o usuário precisa aguardar que algum funcionário da parte administrativa do complexo esteja disponível, para só então, respondê-lo. Com objetivo de apresentar uma solução que torne o processo de agendamento de horários mais prático, este trabalho documenta o desenvolvimento de um aplicativo android, com versões para usuários e administradores do complexo, que torna o agendamento muito mais simples, visto que ele pode ser feito de qualquer local, através do aplicativo e que o usuário não precisa aguardar a disponibilidade de um funcionário da secretaria do complexo, para respondê-lo. Como resultado, é apresentado o aplicativo que permite a realização de agendamentos dos espaços do complexo esportivo, além da validação do mesmo com usuários através de questionamentos feitos no formato de formulário, após a apresentação do aplicativo, e sugestões para trabalhos futuros.

PALAVRAS-CHAVE

Sistema de agendamentos, agendamento de horários, agendamento de serviços, mobile, android.

LINKS ÚTEIS

Códigos fonte do back-end e do front-end, respectivamente:

https://github.com/italomp/SAHCE_UFCG_BACK

https://github.com/italomp/SAHCE_UFCG_FRONT

1. INTRODUÇÃO

Atualmente, para usar algum dos espaços disponíveis no complexo esportivo da Universidade Federal de Campina Grande (UFCG), é necessário realizar um agendamento de horário, presencialmente, na secretaria deste complexo, ou ainda, via contato por whatsapp (aplicativo de mensageria). Caso um usuário opte por usar algum dos espaços, sem agendar previamente o horário para usá-lo, esse usuário terá que sair do espaço, caso alguém tenha agendado aquele horário e compareça ao local no horário marcado. Dito isso, é possível perceber a importância de agendar o horário para garantir o uso ininterrupto do mesmo. Também é possível ver que realizar o agendamento é uma experiência desconfortável para os dias atuais, pois é necessário deslocar-se até a secretaria do complexo para realizar o agendamento, ou ainda, aguardar um atendimento via whatsapp. Da mesma forma, se um usuário quiser consultar se algum horário foi cancelado, para que este possa agendar o horário livre para si, ele precisa ir à secretaria novamente, ou aguardar novamente a disponibilidade do pessoal da secretaria do complexo esportivo para realizar o atendimento via whatsapp. Isso não é nada confortável. Outro fator desagradável, é que o agendamento não pode ser realizado das 11:01 às 13:59, pois este é o horário de almoço dos funcionários da secretaria do complexo esportivo. E este, poderia ser o único horário livre no dia de alguém com interesse em agendar um horário para usar algum dos espaços.

Atualmente, existem vários aplicativos para a realização de agendamento de espaços esportivos, como por exemplo: WebQuadras, Alô Quadras, Easyquadra, Agendei Quadras, entre outros. Entretanto, há uma particularidade no complexo esportivo da UFCG, que nenhum desses aplicativos leva em consideração: o complexo esportivo da UFCG conta com três perfis de usuários (administradores, usuários da comunidade interna e usuários da comunidade externa). Os administradores são os funcionários da secretaria do complexo esportivo, os usuários da comunidade interna são alunos e funcionários da UFCG e os usuários da comunidade externa são pessoas sem vínculo com a UFCG. A distinção entre usuários da comunidade interna e usuários da comunidade externa é relevante para o domínio do problema, pois os usuários da comunidade interna têm acesso aos horários disponíveis para agendamento antes dos usuários da comunidade externa e isso deve ser considerado no desenvolvimento de um sistema que automatize o processo de agendamento. Pensando nisso, este trabalho documenta o desenvolvimento de um aplicativo mobile para realização dos agendamentos, melhorando a experiência dos usuários do complexo, visto que não será necessário estar fisicamente na secretaria do complexo para realizar o agendamento, nem também, aguardar atendimento via whatsapp. Além disso, o aplicativo desenvolvido neste trabalho, considera os três perfis de usuários mencionados acima. A escolha por uma aplicação mobile, se dá pelo fato de que as pessoas levam seus celulares consigo para todos os locais que vão durante o dia. E dessa maneira, poderão realizar o agendamento a qualquer momento, assim como consultar possíveis cancelamentos de horários, para posteriormente agendarem esses horários para si.

2. SOLUÇÃO

A solução é um aplicativo android que atende aos três tipos de usuários mencionados na seção 1, permitindo o cadastro de espaços e horários, validação do cadastro de usuários, realização e cancelamento de agendamentos, por parte do administrador. E a

consulta e de horários disponíveis e agendamento dos mesmos, por parte dos outros dois tipos de usuários mencionados.

2.1 Visão Geral

Para uma fácil descrição do aplicativo, é interessante dividi-lo em duas versões: uma versão para os usuários internos e externos, e outra versão para os administradores. Além disso, é interessante não descrever as regras de negócio nessa seção, pois a explicação ficaria complexa. Faremos a descrição dessas regras na seção 2.2.

Discorrendo sobre a versão dos usuários internos e externos, a tela inicial é uma tela de login, onde os mesmos podem realizar o login ou clicarem no link "registre-se" para realizarem seus cadastros no sistema. Caso os usuários realizem o login, poderão acessar uma tela de consulta de horários disponíveis, filtrando a consulta por espaço (quadra de futsal, quadra de voleibol, etc) e período em que os horários acontecem. Além disso, podem reservar algum dos horários disponíveis ou cancelar algum agendamento feito.

Já na versão do administrador, ele possui acesso a tudo que foi descrito na versão dos usuários internos e externos e mais algumas telas, sendo elas: telas de listagem e de cadastro de espaços, telas de listagem e cadastro de horários e telas de listagem e validação de cadastros de usuários.

Para tornar a descrição mais lúdica, A figura 1, mostra o dashboard do administrador, com a tela de consulta de horários disponíveis, selecionada. O dashboard dos usuários não administradores é semelhante, mas diferencia-se por não possuir a barra de navegação inferior, que permite acessar telas exclusivas do administrador.

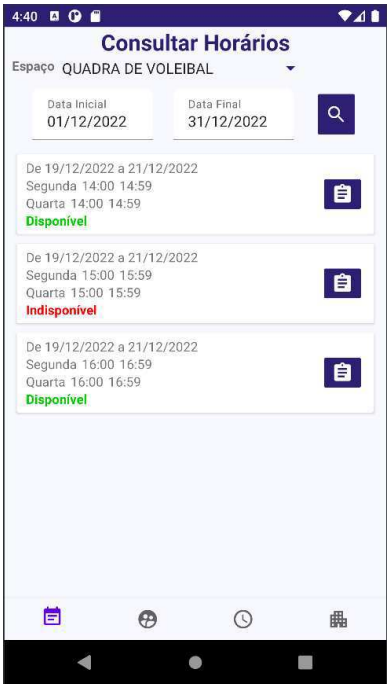


Figura 1 - Dashboard do administrador.

2.2 Funcionalidades

O autor conversou com funcionários da secretaria do núcleo de esportes da UFCG (Nuesp) e coletou os seguintes requisitos:

2.2.1 Cadastro de usuários.

Esta funcionalidade não possui nenhuma pré-condição a ser atendida e é executada por usuários ainda não cadastrados no sistema. Como restrição, a funcionalidade exige que os seguintes dados cadastrais sejam fornecidos: nome do usuário, uma senha para autenticação no sistema, endereço, e-mail, telefone, RDM (para usuários da comunidade interna) e foto de um documento de identificação com foto (para usuários da comunidade externa). E como pós-condição, o cadastro deve ser aprovado por um administrador.

Enquanto o usuário não tiver o cadastro validado pelo administrador, ele ficará com o estado inativo e não poderá efetuar login no sistema e consequentemente não poderá usá-lo.

2.2.2 Listagem dos usuários inativos.

Esta funcionalidade permite visualizar quais usuários estão aguardando a validação de seus cadastros e acessar a tela de detalhes do cadastro, onde a validação pode ser feita. Esta funcionalidade é exclusiva do administrador e como pré-condição, o administrador deve estar autenticado no sistema.

2.2.3 Validação dos cadastros dos usuários.

Esta função é exclusiva do administrador e como pré-condição, o administrador deve estar autenticado no sistema. Após a aprovação do cadastro o usuário cujo cadastro foi aprovado passará a ter o estado ativo e poderá usar o sistema.

2.2.4 Autenticação de usuários.

Esta funcionalidade será necessária para que os usuários sejam identificados e possam usar o sistema. Como pré-condição desta funcionalidade, o usuário deve estar cadastrado e ativo no sistema.

2.2.5 Cadastro de espaços.

Para a realização do cadastro, o administrador deve informar o nome do espaço e quais tipos de usuários (interno e/ou externo) podem usar aquele espaço. Esta funcionalidade é exclusiva do administrador e como pré-condição, o administrador deve estar autenticado no sistema.

2.2.6 Listagem de espaços cadastrados.

Esta funcionalidade é exclusiva do administrador e como pré-condição, o administrador deve estar autenticado no sistema.

2.2.7 Cadastro de horários associados a espaços cadastrados no sistema.

Para o cadastro de um horário, é necessário informar: o espaço associado ao horário, os dias da semana em que o horário acontece, os horários em que o horário acontece em cada um dos dias informados, as datas de início e fim do horário e por fim, as datas de liberação da realização de reservas para cada um dos tipos de usuários (internos e externos), pois essas datas são distintas, uma vez que os usuários internos têm prioridade no agendamento dos horários e, logo, têm a liberação da realização de reservas antes dos usuários externos. Esta funcionalidade é exclusiva do administrador e como pré-condição, o administrador deve estar autenticado no sistema.

2.2.8 Listagem de horários cadastrados.

Esta funcionalidade permite listar os horários cadastrados no sistema, filtrando os horários por um dos espaços cadastrados no sistema, para que a listagem não fique visualmente poluída. Esta, também é uma funcionalidade exclusiva do administrador e como pré-condição, o administrador deve estar autenticado no sistema.

2.2.9 Listagem de horários disponíveis para o agendamento.

Essa listagem pode ser feita tanto pelo administrador, como também pelos usuários interno e externo, e é através dessa listagem que os usuários escolhem um horário para realizar o agendamento. Essa é a listagem exibida na figura 1 e não deve ser confundida com a listagem mencionada na seção 2.2.8, que por sua vez, é exclusiva do administrador e serve para gerenciamento dos horários existentes. Como pré-condição, o usuário deve realizar autenticação no sistema.

2.2.10 Realização de agendamento.

Esta funcionalidade permite que o usuário, administrador ou não, realize o agendamento de um horário disponível. Como pré-condição, o usuário deve realizar autenticação no sistema. Além disso, esta funcionalidade possui as seguintes restrições: o usuário não pode agendar mais que um espaço para ser usado no mesmo horário, o usuário não pode agendar mais de um horário numa modalidade de esporte por semana, o usuário deve pertencer à comunidade autorizada a usar o espaço, pois existem espaços exclusivos para cada comunidade, e por fim, o usuário deve realizar o agendamento entre o dia de liberação para a realização do agendamento e o último dia em que o horário acontece.

2.2.11 Cancelamento de agendamento.

Esta funcionalidade pode ser executada por todos os usuários, desde que o usuário que está cancelando o horário seja o dono do horário, ou seja um usuário administrador. Como pré-condições, o usuário deve realizar autenticação no sistema.

2.3 Arquitetura

O sistema possui uma arquitetura Model-View-Presenter (MVP), sendo o model (modelo), a parte responsável pela lógica do sistema e por interagir com a camada de persistência dos dados; a view (visão), a parte responsável por exibir as informações e interagir com o usuário; E o presenter (apresentador) a parte responsável por intermediar a comunicação entre a view e o model. Ele recebe os eventos de usuário, enviados pela view, encaminha-os para o model, que por sua vez, executa a lógica de negócio acionada por aquele evento, responde ao presenter que executou o que devira e o presenter, finalmente, envia uma mensagem para a view com possíveis recursos solicitados ou apenas uma mensagem de confirmação. A imagem abaixo ilustra isso.

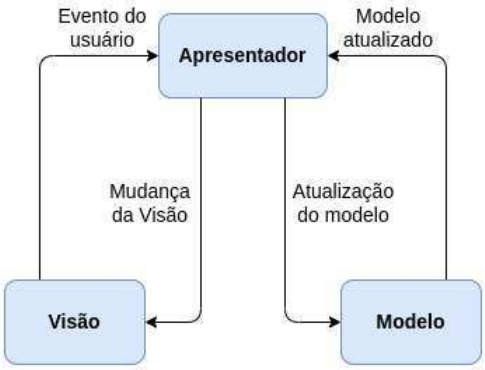


Figura 2 - Arquitetura do Sistema

2.4 Back-end

O back-end da aplicação contém seis módulos, sendo eles: o módulo de segurança, o módulo de usuários, o módulo de espaços, o módulo de exceções, o módulo de utilitários e o módulo de testes. Cada um dos módulos, exceto os módulos de segurança, de utilitários e de testes, possuem 4 camadas, sendo elas: a camada de modelos, onde as ficam as classes que representam as entidades gerenciadas pela aplicação; a camada de repositórios, que é responsável pela persistência e recuperação das informações gerenciadas pelo sistema; a camada de serviços, que é responsável por executar lógica da aplicação; e por fim, a camada de apresentadores, que é responsável por intermediar a comunicação entre a visão e o modelo (aqui representado pelas camadas de modelo, repositórios e serviços) da aplicação.

2.4.1 Módulo de Segurança

O módulo de segurança é responsável pela implementação das funcionalidades de autenticação, autorização e definição de níveis de permissão.

A autenticação e a autorização, funcionam da seguinte forma: o cliente (front-end) envia as credenciais para o back-end, estas credenciais são validadas pelo módulo de segurança e um token, que encapsula o e-mail do usuário é retornado para o cliente, para que toda vez que ele precise acessar alguma rota privada (rota que não pode ser acessada sem autenticação prévia), ele anexe esse token à requisição para que o token seja recebido pelo modo de segurança, o seu conteúdo seja extraído e validado, para que só então, se o conteúdo do token for válido e o token não tiver expirado, o acesso ao recurso (informação gerenciada pelo sistema) solicitado, possa ser feito pelo cliente. A parte responsável pela validação das credenciais e retorno do token, é a autenticação. E a parte da validação do token e liberação do acesso, é a autorização. A figura 3 ilustra o processo de autenticação e autorização, mencionado acima.

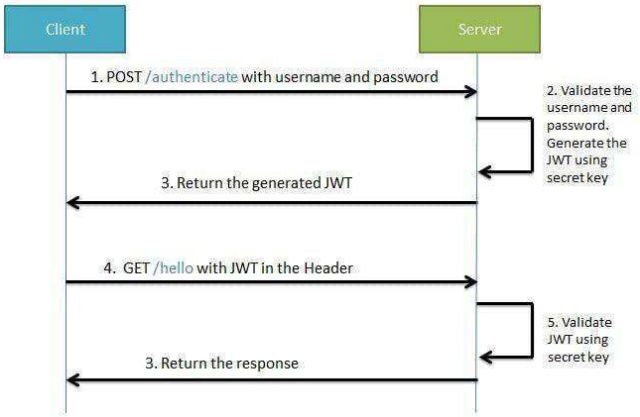


Figura 3 - Processo de autenticação e autorização.

Para a implementação das funcionalidades de autenticação e autorização, foram utilizados Spring Security e JSON Web Token (JWT).

O Spring Security é um framework do ecossistema Spring (conjunto de frameworks complementares para o desenvolvimento de aplicações) e é composto por um conjunto de filtros de segurança, dos quais o usuário pode escolher os que deseja configurar e adicionar ao projeto. Enquanto JWT, é uma biblioteca que criptografa mensagens trocadas entre cliente e servidor, aumentando a segurança na comunicação entre eles. Para este projeto, foram escolhidos os seguintes filtros: Username Password Authentication Filter, para configurar a autenticação via username (nome de usuário), que neste caso é o e-mail, e password (senha) do usuário. E Basic Authentication Filter, para realizar a autorização, com base no token fornecido pelo usuário. Além disso, é necessário configurar o adaptador Web Security Configurer Adapter para que os filtros mencionados sejam adicionados à cadeia de filtros do Spring Security.

Para a implementação dos níveis de permissão, foram usados o roles (perfis de usuários, chamados de papéis, em tradução livre) e as authorities (autoridades associadas às roles) que o spring security nos permite criar. Inicialmente, é intuitivo pensar que a aplicação possui 3 roles, visto que existem três tipos de usuários. Entretanto, os roles servem para restringir o acessos dos usuários a um conjunto de endpoints (endereços de acesso aos recursos do sistema) bem definidos. E os endpoints acessados pelos usuários não administradores, são os mesmos. Logo, a aplicação conta com apenas dois roles: USER e ADMIN. A distinção entre usuários interno e externo é feita através de um atributo no modelo dos usuários. E o controle do que esses usuários podem fazer, é feito na camada de serviços dos módulos que usam a entidade usuário. Esse controle é feito em diferentes serviços porque cabe ao serviço de cada módulo definir o que pode, ou não, ser feito naquele contexto.

2.4.2 Módulo de Usuários

O módulo de usuários é responsável pelas operações diretamente ligadas às alterações de estado (alteração dos dados) dos usuários. Ele permite o cadastro de usuários, desde que a requisição de cadastro forneça os dados descritos na seção 2.2.1, permite a ativação dos usuários, permite a obtenção de um usuário específico, através do seu email, e permite também a obtenção de uma lista de usuários inativos. Lista essa que é exibida no front-end para que o administrador veja quais usuários estão aguardando a aprovação do cadastro.

2.4.3 Módulo de Espaços

O módulo de espaços permite que o administrador cadastre os espaços no sistema, informando que tipo de usuário (interno e/ou externo) pode acessar o espaço. Além de permitir a obtenção de uma lista de espaços já cadastrados nos sistema, para que o administrador possa visualizar quais espaços já foram cadastrados.

2.4.4 Módulo de Horários

O módulo de horários, é o módulo que manipula a principal entidade do domínio deste sistema, os horários. Além disso, este módulo gerencia o agendamento destes horários. Não foi criado um módulo de agendamentos, pois embora haja uma separação de responsabilidades nessa abordagem, haveria também um aumento na complexidade para relacionar as entidades gerenciadas pelo sistema.

As funcionalidades encapsuladas nesse módulo, permitem a criação de um novo horário; a obtenção de uma lista de horários associados a um determinado espaço e, opcionalmente, filtrando pela data em que os horários acontecem; a reserva de um horário para um usuário; e por fim, o cancelamento de uma reserva feita pelo usuário.

Vale ressaltar que a criação de um horário obedece às pré-condições da seção 2.2.7. A obtenção de uma listagem de horários atende às pré-condições das seções 2.2.8 e 2.2.9. O agendamento de um horário obedece às restrições e pré-condições da seção 2.2.10. E o cancelamento de um agendamento, obedece às restrições da seção 2.2.11.

2.4.5 Módulo de Exceções

O módulo de exceções, possui um conjunto de exceções customizadas, que estão mais próximas do domínio do problema e são menos generalistas que o conjunto de exceções fornecido pela linguagem escolhida para fazer o back-end (java). Além disso, este módulo possui um tratador de exceções global, que é uma entidade que captura todas as exceções lançadas nos controladores e monta mensagens de erro customizadas para serem enviadas ao usuário, quando estas exceções ocorrerem. O motivo da criação de mensagens de erro customizadas, é retornar mensagens mais objetivas (mensagens visualmente mais limpas) para facilitar a leitura dos erros, pelo cliente (o front-end) que consome o back-end.

2.4.6 Módulo de Utilitários

O módulo de utilitário, encapsula funções que são usadas em diferentes classes do sistema. E tem como objetivo reduzir a duplicidade e aumentar o reuso de código, e assim, facilitar a manutenção e evolução do sistema.

2.4.7 Módulo de Testes

O módulo de testes, possui testes unitários que garantem o bom funcionamento do sistema e facilitam a evolução do mesmo, visto que quando uma nova funcionalidade for adiciona ao sistema, não será necessário testar todas as outras funcionalidades manualmente para garantir que nenhum efeito colateral seja adicionado ao sistema, pela adição da nova funcionalidade.

Testes unitários, são testes automatizados que testam pequenas unidades do sistema, chamadas de classes. E efeito colateral, é um comportamento que é adicionado involuntariamente ao sistema ao realizar alguma alteração no mesmo, o que causa mau funcionamento do código já implementado.

2.4.8 Persistência

A persistência dos dados é feita no PostgreSQL, um banco de dados relacional (banco que armazena informações em tabelas, também chamadas de relações, no meio acadêmico). A escolha por um banco de dados relacional se deu pelo fato de esse tipo de banco de dados dar suporte às propriedades básicas de uma transação: atomicidade, consistência, isolamento e persistência. Propriedades estas que garantem que o front-end sempre irá receber do back-end, dados consistentes e atualizados.

A atomicidade garante que uma transação no banco de dados é atômica, ou seja, ou todas as suas operações são executadas, ou nenhuma delas é executada. Essa propriedade evita inconsistências no banco, causadas por uma transação apenas parcialmente executada. A consistência garante que apenas dados válidos sejam persistidos no banco, evitando inconsistências no banco, como por exemplo, uma coluna de uma tabela armazenar um tipo de dado diferente do que ela foi configurada para armazenar. O isolamento, garante que as transações são executadas isoladamente, sem que uma transação mexa em dados que estão sendo manipulados naquele momento por outra transação. Isso garante um controle de concorrência e evita que computações erradas sejam feitas sobre os valores do banco. E por fim, a persistência garante que os dados que forem armazenados no banco, não serão perdidos.

2.4.9 Tecnologías

Para a construção do back-end, foi usada a linguagem de programação java, os frameworks Spring Boot, Spring Security, Spring Data JPA e Hibernate. Além das bibliotecas JWT, JUnit e Mockito e o banco de dados PostgreSQL.

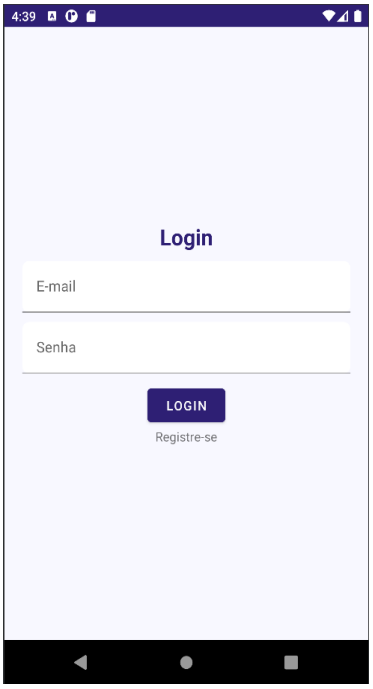
Frameworks e bibliotecas, são conjuntos de soluções prontas, em forma de código-fonte (código construído com uma linguagem de programação), que solucionam problemas recorrentes. A diferença entre framework e biblioteca, é que um framework invoca o código do desenvolvedor, enquanto uma biblioteca é invocada pelo código do desenvolvedor. O Spring Boot, é um framework que auxilia na construção de sistemas web. Spring Data JPA, é uma um framework que permite que o back-end acesse e altere as informações do banco de dados. Hibernate, é um framework que permite mapear entidades da linguagem de programação java para tabelas de um banco de dados relacional. JUnit e Mockito são bibliotecas java para realização de testes unitários. PostgreSQL é um banco de dados relacional, como mencionado na seção 2.4.8. Por fim, Spring Security e JWT já foram apresentados na seção 2.4.1.

2.5 Front-end

O front-end da aplicação, assim como o back-end, também foi dividido em módulos, sendo eles: módulo de login, módulo de cadastro de usuários, módulo de agendamentos, módulo de horários e módulo de locais.

2.5.1 Módulo de Login

O módulo de login permite a autenticação do usuário no sistema e conta com apenas uma tela. Como mostra a figura 4, basta o usuário inserir suas credenciais (e-mail e senha) e clicar no botão de login, para que a requisição de autenticação seja enviada para o back-end da aplicação e caso as credenciais sejam válidas, o usuário seja direcionado para o dashboard do usuário, descrito na seção 2.1.



Fígura 4 - Tela de login

Caso o usuário ainda não possua cadastro no sistema, ele pode

tocar no link “Registre-se” e ser direcionado para uma das telas do módulo de cadastro de usuários, a tela de cadastro de usuários.

2.5.2 Módulo de Cadastro de Usuários

O módulo de cadastro de usuários, conta com três telas que permitem realizar o cadastro de usuários, a listagem de cadastros ainda não aprovados pelo administrador e a aprovação dos cadastros dos usuários.

A figura 5, mostra a tela de cadastro e através dela, o usuário consegue inserir todos os dados mencionados na seção 2.2.1 e em seguida, tocar no botão “enviar”, para que o cadastro seja submetido e posteriormente aprovado pelo administrador.

Figura 5 - Tela de cadastro de usuários

Já a figura 6, apresenta a tela de listagem de usuários inativos, que aguardam a aprovação do cadastro pelo administrador.

Figura 6 - Tela de listagem de usuários inativos

O ícone de prancheta, dentro do card (componente visual que parece um cartão) de cada usuário, permite que o administrador acesse a tela de detalhes do cadastro de um usuário, para que possa visualizar os dados cadastrais e em seguida, aprovar o cadastro, como é mostrado na figura 7. Além disso, a tela de listagem de usuários também permite que o administrador toque no botão com o símbolo de adição e seja redirecionado para a tela de cadastro de usuários, para que ele também seja capaz de realizar o cadastro de novos usuários.

Figura 7 - Tela de detalhes do cadastro de um usuário

2.5.3 Módulo de Agendamentos

O módulo de agendamentos, conta com duas telas que permitem a listagem dos horários, o agendamento de horários e o cancelamento de agendamentos.

A figura 1, mostra a tela de listagem dos horários, a qual possui campos para que o usuário insira o período e o espaço, nos quais os horários acontecem, para que o usuário possa filtrar apenas os horários associados ao período e ao espaço de seu interesse.

Já a figura 8, mostra a tela de detalhes de um horário, na qual o usuário pode visualizar todos os detalhes de um horário e realizar o agendamento de um horário ou o cancelamento de um agendamento, ambas as funcionalidades respeitando as restrições citadas nas seções 2.2.10 e 2.2.11. Além disso, para melhorar a experiência do usuário, se ele não estiver apto a realizar o agendamento de um horário ou o cancelamento de um agendamento, os respectivos botões, mostrados na figura 8, estarão inativos conforme acontece com o botão de cancelamento, na figura 8.

Figura 8 - Tela de detalhes de um horário

2.5.4 Módulo de Horários

O módulo de horários, conta com duas telas que permitem que o administrador liste os horários já cadastrados, além de permitir que ele cadastre novos horários associados aos espaços cadastrados no sistema.

A figura 9, mostra a tela de listagem de horários, através da qual é possível ser direcionado para a tela de cadastro de horários, pressionando o botão com símbolo de adição.

Figura 9 - Tela de listagem de horários para o administrador

Já a figura 10, mostra a tela de cadastro de horários, através da qual é possível fornecer os dados citados na seção 2.2.7 e submeter o cadastro através do botão "cadastrar".

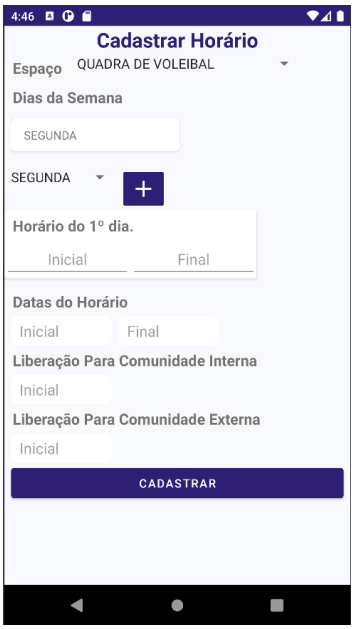


Figura 10 - Tela de cadastro de horários

2.5.5 Módulo de Espaços

O módulo de espaços, conta com duas telas que permitem a listagem dos espaços já cadastrados e o cadastro de um novo espaço.

A figura 11, mostra a tela de listagem de horários, que contém uma listagem de cards que representam os horários e indicam, através de checkboxes (caixas de seleção), a quais tipos de usuários, cada espaço é destinado.

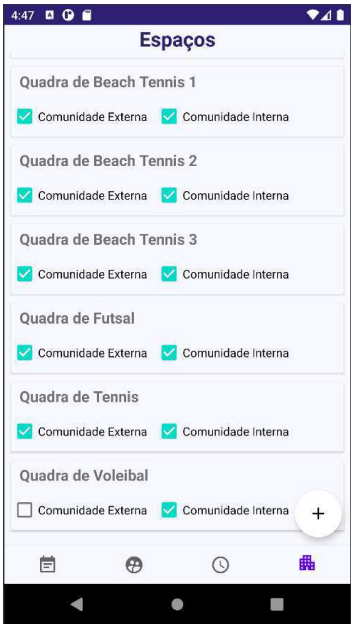


Figura 11 - Tela de listagem de espaços

Além disso, através do botão com um ícone de adição, localizado na tela de listagem de espaços, é possível acessar a tela de cadastro de espaços. Nesta tela, o usuário pode inserir os dados mencionados na seção 2.2.5 e realizar o cadastro do espaço, pressionando o botão "enviar", conforme mostra a figura 12.

Figura 12 - Tela de cadastro de espaços.

3. METODOLOGIA

Inicialmente, o autor conversou com funcionários do Nuesp para entender a logística por trás dos agendamentos de horários e levantou os requisitos apresentados na seção 2 como funcionalidades. Com os requisitos bem definidos, o autor utilizou o quadro kanban do trello (ferramenta web que auxilia o desenvolvimento de projetos) para montar um backlog do produto (lista com tarefas a serem realizadas) e ter monitoramento organizado e em tempo real do andamento do projeto. Após isso, começou a implementar as funcionalidades, uma por uma, desenvolvendo-as fim a fim, no front-end e no back-end, sempre investigando como implementá-las no front-end e às vezes investigando como implementá-las no back-end, como aconteceu nas funcionalidades de autenticação e autorização, além de algumas consultas ao banco de dados, feitas via Spring Data JPA. E após a implementação das das funcionalidades propostas, o autor apresentou o aplicativo resultante a alguns usuários para coletar feedbacks e sugestões de trabalhos futuros, mencionados nas seções 5 e 6, respectivamente.

4. EXPERIÊNCIA

Com relação à experiência vivenciada na construção deste trabalho, é possível citar dificuldades enfrentadas ao longo do processo e ressaltar pontos importantes para o sucesso do trabalho.

A primeira dificuldade encontrada foi para realizar o login no sistema. O autor configurou o front-end da aplicação para enviar requisições para o back-end no endereço de localhost (hospedeiro local de aplicações). Entretanto, o front-end estava sendo executado em um emulador de android, que é uma máquina virtual e possui seu próprio localhost, que por sua vez, é diferente do localhost da máquina onde o emulador de android está executando. Então, as requisições do front-end estavam sendo enviadas para o localhost do emulador, enquanto deveriam ser enviadas para o local host da máquina em que o back-end estava executando, que é a mesma máquina onde o emulador de android estava executando. Após uma investigação, o autor descobriu que o emulador de android faz um mapeamento entre o endereço de IP 10.0.2.2 e o localhost da máquina na qual o emulador está executado. E com isso, conseguiu resolver este impedimento.

O segundo problema, aconteceu com a serialização de objetos que representavam as datas manipuladas pelo sistema. Inicialmente, não era possível serializar objetos que tivessem atributos do tipo LocalDate e houve a necessidade de adicionar adaptadores que mapeassem as datas para JSON para que essas entidades pudessem ser transferidas entre front-end e back-end. No entanto, existe uma solução muito mais simples, que é usar objetos de transferência de dados (data object transfer - DTOs) que representem a entidade que possui uma atributo do tipo LocalDate, mas substituindo esse atributo por um atributo do tipo string (tipo de dado que representa valores textuais) e que armazene o valor em string que representa a data. A partir dessa string o objeto LocalDate pode ser construído no destino da requisição (back-end ou front-end) que transmite o DTO. Como a primeira solução, resolveu o problema e foi trabalhosa de ser implementada, ela não foi desfeita, pois ainda faltavam muitas funcionalidades para serem implementadas e essas foram priorizadas. Essa primeira solução foi construída no front-end com um mapeador de entidades chamado Moshi e no back-end foi feita com um mapeador de objetos chamado Jackson. Contudo, fica registrado aqui, a possibilidade de uma solução simples, como foi mencionado acima.

O terceiro e mais agravante dos problemas, foi o baixo poder computacional da máquina do autor. A IDE (Integrated Development Environment) ideal para desenvolver aplicativos para android (sistema operacional do google), segundo a documentação oficial do android, é o Android Studio. E executar o Android Studio no notebook do autor, que possui um processador core i3 de quinta geração e apenas 8GB de memória ram, com frequência de 1666mhz, foi extremamente custoso e atrasou muito o desenvolvimento do front-end. Para mitigar esse problema, um upgrade do notebook foi feito, adicionando mais 8GB de memória ram ao computador, além de usar o github para compartilhar o código e trabalhar de outro computador com maior poder computacional. O upgrade do notebook, tornou o uso do Android Studio totalmente possível, embora a execução do mesmo ainda tivesse um pouco de latência nas respostas aos comandos dados.

Já falando sobre os pontos positivos do processo de construção do aplicativo, é possível mencionar dois pontos extremamente relevantes para o autor. São eles: o uso de um quadro kanban para monitorar o desenvolvimento do projeto e visualizar a quantidade de tarefas restantes, evitou que o autor subestimasse o trabalho restante e desse menos atenção que o necessário para o desenvolvimento do projeto, evitando assim, possíveis atrasos. Além disso, outro ponto relevante para o autor foi a investigação sob demanda, para implementar as funcionalidades do sistema, pois essa abordagem otimizou o tempo necessário para geração de resultados, evitando que assuntos que não foram aplicados ao

aplicativo fossem estudados, causando um aumento no tempo necessário para essa geração de resultados.

5. AVALIAÇÃO

Após a implementação das funcionalidades propostas, o autor apresentou o aplicativo resultante a oito usuários não administradores, exibindo apenas as funcionalidades destinadas a esse tipo de usuário, para coletar feedbacks dos mesmos. E apresentou o aplicativo, com todas as funcionalidades a um dos funcionários da Nuesp, com o mesmo intuito. Vale ressaltar que todos os funcionários do Nuesp estavam de férias no mês de janeiro, quando o autor foi apresentar o aplicativo resultante para coletar feedbacks, inclusive o usuário que respondeu aos questionamentos apresentados aqui. Devido a esse fato, houve feedback de apenas um dos funcionários do Nuesp, que se dispôs a colaborar com este trabalho durante o seu período de férias.

Para avaliar o aplicativo, foram formulados dois conjuntos de perguntas, um para os usuários não administradores e outro para o usuário administrador.

As perguntas feitas aos usuários não administradores foram as seguintes: “O aplicativo possui uma interface intuitiva para a realização das funcionalidades propostas?”, “É possível realizar o cadastro de usuário de maneira simples e intuitiva?”, “É possível realizar agendamentos de maneira simples e intuitiva?”, “É possível cancelar um agendamento de maneira simples e intuitiva?”. E não houve nenhum dos oito usuários que respondeu “não” para alguma dessas perguntas acima.

Já para o usuário administrador, as seguintes perguntas foram feitas: “O aplicativo possui uma interface intuitiva para a realização das funcionalidades propostas?”, “É possível cadastrar espaços de maneira simples e intuitiva?”, “É possível cadastrar horários de maneira simples e intuitiva?”, “É possível validar o cadastro de usuários de maneira simples e intuitiva?”, “É possível realizar um agendamento de maneira simples e intuitiva?”, “É possível cancelar um agendamento de maneira simples e intuitiva?”. E assim como os usuários não administradores, o usuário administrador respondeu sim para todas as perguntas mencionadas, o que demonstra que o aplicativo realmente cumpre o que se propõe a fazer.

Além do feedback coletado acima, o usuário administrador relatou que o aplicativo tornaria o processo de agendamento mais prático, visto que com o uso do aplicativo, o administrador não precisaria responder inúmeras mensagens através do whatsapp do Nuesp, referentes aos agendamentos. Além disso, ele informou que com o uso do aplicativo, haveria um maior controle sobre os usuários que estão usando o complexo, pois com o aplicativo, os usuários precisam realizar a autenticação com suas credenciais para realizar os agendamentos, o que aumenta a confiabilidade sobre a autenticidade dos mesmos. Já pelo whatsapp, esse controle sobre a autenticidade dos usuários seria, no mínimo, mais trabalhoso de ser feito.

Apesar das respostas positivas em relação à avaliação do aplicativo, houveram sugestões de melhoria. Estas são mencionadas na conclusão deste trabalho.

6. CONCLUSÃO

Como resultado deste trabalho, foi obtido um aplicativo android que contém todas as funcionalidades inicialmente propostas, sendo capaz de realizar os agendamentos de espaços do complexo esportivo, como foi almejado no início do trabalho. Além disso, o aplicativo foi validado por usuários, como foi apresentado na seção 5. Apesar disso, existem algumas sugestões de melhorias, feitas pelo autor, pelos usuários não administradores e pelo usuário administrador.

Começando pela sugestão do autor, é interessante que haja a adição de barras de progresso enquanto as telas do aplicativo estão carregando. Isso evita que mensagens ANR (App Not Responding) apareçam e encerrem a aplicação. Isso pode acontecer se, enquanto uma tela está sendo carregada, o usuário interagir com ela através de toques em componentes que geram eventos e o app não for capaz de responder a esses eventos em um intervalo de cinco segundos. Esse problema não aconteceu durante os testes, mas sem o uso das barras de progresso, haverá um cenário que não impede o acontecimento de mensagens ANR. Já os usuários não administradores, sugeriram que a exibição dos horários disponíveis para agendamento fosse feita em formato tabular e que a interface do aplicativo poderia ficar com um design mais bonito. A primeira sugestão é uma evolução que aumentaria a complexidade do aplicativo, pois seriam necessárias n tabelas para n estilos de horários. Por exemplo: uma tabela com horários semanais, uma tabela com horários mensais, uma tabela com horários semestrais, etc. Apesar disso, a apresentação dos horários ficaria realmente melhor, pois seria mais fácil visualizar dia e hora em que existem horários vagos, se comparado com a apresentação em forma de lista. E com relação à segunda sugestão, vale ressaltar que uma tela com design profissional, deve ser feita por um profissional da área de design, não de computação. Portanto, para o alcance desse propósito, é interessante que haja a colaboração de tal profissional. Por fim, com relação às sugestões do usuário administrador, foram sugeridas duas melhorias. A primeira é uma evolução para a funcionalidade de cancelamento de agendamento, onde ao realizar o cancelamento de um agendamento, uma mensagem justificando o cancelamento deve ser informada pelo administrador e o usuário dono do horário deve receber uma notificação sobre o cancelamento e deve também poder visualizar a mensagem enviada pelo administrador. A segunda, é a adição de um prazo de validade ao cadastro dos usuários, visando excluir os ex-alunos da comunidade interna e passando a adicioná-los na comunidade externa. Para alcançar esse intuito, o cadastro deveria ter validade de um período acadêmico. Entretanto, as férias entre períodos não possuem a mesma duração (as férias que ocorrem no final de ano são mais longas que as férias que ocorrem no meio do ano). Além disso, alguns eventos podem afetar o calendário acadêmico, como por exemplo, greves ou até mesmo situações excepcionais como epidemias ou pandemias, como aconteceu nos últimos anos. Todos esses fatores dificultam o estabelecimento de um prazo de validade para os cadastros, que acompanhe o calendário acadêmico. Por esse motivo, o período de validade sugerido foi de um ano, que é um período que considera uma margem adicional para eventuais alterações no calendário acadêmico e ainda assim, serve para transferir ex-alunos da comunidade interna para a comunidade externa, mesmo que possibilite que um ex-aluno consiga usar o complexo por mais alguns meses após a sua formatura, como se ainda fosse membro da comunidade interna.

7. REFERÊNCIAS

[1] Documentation. Disponível em: <https://developer.android.com/docs>. Acesso em: 19 dez. 2022.

[2] Overview (java SE 11 & JDK 11). Disponível em: <https://docs.oracle.com/en/java/javase/11/docs/api/>. Acesso em: 19 dez. 2022.

[3] DEINUM, M.; RUBIO, D.; LONG, J. Spring Security. Em: Spring 5 Recipes. Berkeley, CA: Apress, 2017. p. 297–344.

[4] GIERKE, O. et al. Spring data JPA - reference documentation. Disponível em: <https://docs.spring.io/spring-data/jpa/docs/current/reference/html/>. Acesso em: 19 dez. 2022.

[5] Retrofit. Disponível em: <https://square.github.io/retrofit/>. Acesso em: 19 dez. 2022

[6] ROSA, D. Como configurar a autenticação e a autorização no JWT para o Spring Boot em Java. Disponível em: <https://www.freecodecamp.org/portuguese/news/como-configurar-a-autenticacao-e-a-autorizacao-no-jwt-para-o-spring-boot-em-java/>. Acesso em: 19 dez. 2022.

[7] WebQuadras. Disponível em: <https://www.webquadras.com.br/quadras/saopaulo>. Acesso em: 29 jan. 2023.

[8] Alô Quadras. Disponível em: <https://apps.apple.com/br/app/al%C3%B4-quadras/id1572518044>. Acesso em: 29 jan. 2023.

[9] easyquadra. Disponível em: <https://play.google.com/store/apps/details?id=com.dysopsis.easycancha&hl=pt_BR&gl=US>. Acesso em: 29 jan. 2023.

[10] TECNOLOGIAS, A. App Agendei Quadras. Disponível em: <https://www.agendeiquadras.com.br/appAgendei/>. Acesso em: 29 jan. 2023.