

**CENTRO UNIVERSITÁRIO CATÓLICA DO LESTE DE MINAS GERAIS
BACHARELADO EM ENGENHARIA DE SOFTWARE**

PLANO DE TESTES – INGRESSOS, UAI!

Gustavo Henrique Vieira Luna

Gustavo Soares Campos

Isabela Assis Perdigão

Isabelly Andrade Gava de Oliveira

José Ângelo Castro Luciano

Tobias José de Almeida Fernandes

IPATINGA, JUNHO/2023

1.	Introdução	4
1.1	Objetivos	4
1.2	O Sistema de Venda de Ingressos	4
1.3	Escopo	4
2.	Requisitos a serem testados	5
2.1	Teste do Banco de Dados.....	5
2.2	Teste Funcional	6
2.3	Teste do Ciclo de Negócios	6
2.4	Teste da Interface do Usuário.....	6
2.5	Perfil da Performance	6
2.6	Teste de Carga	6
2.7	Teste de Stress	7
2.8	Teste de Segurança e de Controle de Acesso	7
2.9	Teste de Falha/Recuperação.....	7
2.10	Teste de Instalação.....	7
3.	Estratégia de Teste	8
3.1	Tipos de Teste	8
3.1.1	Teste de Integridade de Dados e do Banco de Dados	8
3.1.2	Teste de Função	8
3.1.3	Teste da Interface do Usuário.....	9
3.1.4	Teste de Performance	9
3.1.5	Teste de Carga.....	10
3.1.6	Teste de Segurança e Controle de Acesso.....	11
3.1.7	Teste de Instalação.....	12
3.2	Ferramentas	12
4.	Equipe.....	12
4.1	Equipe de teste.....	12
4.2	Infraestrutura	14
5.	Cronograma	14
6.	Glossário – Definições e Siglas	15
6.1	CRUD – Inserir, Pesquisar, Editar e Deletar.	15
7.	Descrição do Mini-mundo do Projeto.....	15
8.	Materiais de Referência	16
9.	Requisitos de Software	16
9.1	Descrição dos Atores	16
9.2	Requisitos funcionais.....	17
9.3	Requisitos não funcionais	18
9.4	Detalhamento dos casos de uso	18
10.	Diagramas.....	22
10.1	Diagrama de Casos de Uso.....	22
10.2	Diagrama de Classes	22
10.3	Diagramas de Sequência.....	23
10.4	Diagramas de Atividades	25
10.5	Diagramas de Máquina de Estados	27
10.6	Diagrama de objetos.....	27
11.	Teste unitário	28
12.	Teste de Integração	29
12.1	Testes de Interface	29
12.2	Teste de funcionalidades	30
12.3	Teste de Segurança.....	32
12.3.1	Carrinho de compras.....	33
12.3.2	Carrinho de compras sem login.....	33
12.3.3	Usuário e/ou senha incorretos	34
12.3.4	Login	34
12.3.5	Teste de segurança e controle de acesso	35
12.4	Teste de Aceitação	35
12.4.1	Roteiro de teste de usabilidade.....	36
13.	Teste de caixa branca:	38
14.	Teste de Integridade de Banco de Dados	39

15.	Teste de Performance.....	39
16.	Teste de Carga	39
17.	Requisito Funcional Associado.	40
18.	Ferramentas Utilizadas	42

1. Introdução

O site Ingressos UAI, tem como objetivo divulgar eventos de artistas que realizam apresentações em bares e teatros, afim de que o público alvo (jovens e adultos) identifiquem quais são as apresentações que ocorrerão perto de sua casa, a data, o horário, o local e o valor, bem como realizar a compra/reserva de ingressos.

1.1 Objetivos

O documento do Plano de Testes do software Ingressos, Uai! (Site de venda de ingressos para eventos locais) tem como objetivo listar os requisitos que serão testados, recomendando e descrevendo as estratégias a serem empregadas nesses testes. Este documento também identifica os recursos necessários para os testes e para implementação do software.

1.2 O Sistema de Venda de Ingressos

Este projeto tem como objetivo criar uma ferramenta capaz de auxiliar a organização e o acompanhamento de todas as atividades de um site de venda de ingressos e todos os serviços prestados pelo mesmo, tais como, venda de ingressos e divulgação de artistas através de um site responsivo. Para atingir esse objetivo esta ferramenta irá criar e administrar um banco de dados que possibilitará armazenar todas as informações necessárias para o gerenciamento interno de cada evento local a ser divulgado, bem como todas as informações referentes a cada cliente, facilitando a descoberta de eventos locais em suas regiões bem como a compra deles - que será realizado através de um gateway de pagamento-.

1.3 Escopo

O Gerenciador de Serviços Automotivos deverá ser submetido **a testes de unidade, integração, sistema e aceitação.**

Os testes de unidade avaliarão isoladamente o banco de dados, a interface gráfica, e todos os outros componentes do projeto.

O teste de integração testa os componentes, previamente testados isoladamente, acoplados. O objetivo é identificar possíveis falhas nos acoplamentos.

Os testes de sistema avaliarão o funcionamento e o desempenho do sistema como um todo, verificando a eficácia e segurança, além da compatibilidade e integração do software em diferentes ambientes.

Os testes de aceitação apresentarão o produto final para o usuário para validação e últimos ajustes.

Para realizar os testes serão utilizadas máquinas com as configurações mais próximas o possível das máquinas que serão utilizadas pelo usuário final, tentando assim, simular o ambiente final em que o programa será executado.

1.4 Identificação do Projeto

Documento	Criado ou Disponível	Recebido ou Revisado
Especificação de Requisitos	<input checked="" type="checkbox"/> Sim <input type="checkbox"/> Não	<input checked="" type="checkbox"/> Sim <input type="checkbox"/> Não
Plano de Projeto	<input checked="" type="checkbox"/> Sim <input type="checkbox"/> Não	<input checked="" type="checkbox"/> Sim <input type="checkbox"/> Não
Modelo de Análise	<input type="checkbox"/> Sim <input checked="" type="checkbox"/> Não	Sim <input checked="" type="checkbox"/> Não
Modelo de Projeto	<input type="checkbox"/> Sim <input checked="" type="checkbox"/> Não	<input type="checkbox"/> Sim <input checked="" type="checkbox"/> Não
Documento de Arquitetura	<input checked="" type="checkbox"/> Sim <input type="checkbox"/> Não	<input checked="" type="checkbox"/> Sim <input type="checkbox"/> Não
Protótipo	<input checked="" type="checkbox"/> Sim <input type="checkbox"/> Não	<input checked="" type="checkbox"/> Sim <input type="checkbox"/> Não
Manual do Usuário	<input type="checkbox"/> Sim <input checked="" type="checkbox"/> Não	<input type="checkbox"/> Sim <input checked="" type="checkbox"/> Não
Lista de Riscos	<input type="checkbox"/> Sim <input checked="" type="checkbox"/> Não	<input type="checkbox"/> Sim <input checked="" type="checkbox"/> Não

2. Requisitos a serem testados

Esta seção descreve em linhas gerais o conjunto de requisitos a serem testados no projeto a ser desenvolvido, comunicando o que deve ser verificado. Exemplos de requisitos a serem testados são: Estabilidade, desempenho, segurança, interface de usuário, controle de acesso, funcionalidades.

2.1 Teste do Banco de Dados

- Verifique se as informações sobre departamentos, funcionários, clientes, serviços e automóveis podem ser inseridas ou modificadas do Banco de Dados

- Verifique se as informações obtidas no Banco de Dados consistem com as informações reais sobre departamentos, funcionários, clientes, serviços e automóveis.
- Verifique que as informações cadastradas possam ser consultadas.

2.2 Teste Funcional

- Verifique que qualquer usuário cadastrado possa acessar o sistema através de um *login* e senha.
- Verifique se o nível de acesso às funcionalidades do sistema a cada tipo de usuário está correto.

2.3 Teste do Ciclo de Negócios

- Verifique se os relatórios estão sendo gerados corretamente.
- Verifique se o tratamento de exceções está correto
- Verifique se os campos obrigatórios estão sendo preenchidos em cada formulário
- Verifique se os campos estão sendo preenchidos com informações no formato correto em cada formulário

2.4 Teste da Interface do Usuário

- Verifique se cada tela de interface gráfica pode ser facilmente entendida e utilizada.
- Verifique que se os relatórios são apresentados corretamente na tela.
- Verifique se os formulários de cadastro e edição estão pegando os dados inseridos pelo usuário corretamente.

2.5 Perfil da Performance

- Verifique o tempo de resposta de consultar/inserção/edição no banco de dados.
- Verifique o tempo de resposta da troca de informações entre servidor e terminais.

2.6 Teste de Carga

- Verificar a resposta do sistema com 5 usuários.
- Verificar a resposta do sistema com 10 usuários.
- Verificar a resposta do sistema com 20 usuários.

- Verificar a resposta do sistema com 30 usuários.
- Verificar a resposta do sistema com 40 usuários.
- Verificar a resposta do sistema com 50 usuários.

2.7 Teste de Stress

- Verifique como o sistema se comporta em situações onde são realizadas várias operações (inserir/editar/remover) simultâneas no banco de dados.
- Verifique como o sistema se comporta em situações onde há pouca memória RAM disponível e/ou pouca memória em disco.

2.8 Teste de Segurança e de Controle de Acesso

- Verificar que apenas usuários cadastrados podem acessar informações e funcionalidades do sistema.
- Verificar que somente o administrador tem acesso a cadastrar/editar/remover e consultar departamentos e funcionários.
- Verificar que todos usuários cadastrados no sistema possam cadastrar/editar/remover e consultar informações sobre clientes, serviços e automóveis.

2.9 Teste de Falha/Recuperação

Nenhum.

2.10 Teste de Instalação

- Verifique que a instalação do sistema ocorre normalmente em todas as máquinas que possuam os requisitos mínimos.
- Verifique que a ferramenta possa ser instalada em diferentes ambientes (ex: Windows XP ou Windows Vista)
- Verifique que a atualização dos dados no servidor se reflete em todos os terminais.

3. Estratégia de Teste

Apresenta um conjunto de tipos de testes a serem realizados, respectivas técnicas empregadas e critério de finalização de teste. Além disso, é listado o conjunto de ferramentas utilizadas.

3.1 Tipos de Teste

3.1.1 Teste de Integridade de Dados e do Banco de Dados

Objetivo do Teste:	Garantir que o acesso ao banco de dados funciona adequadamente e sem inconsistência dos dados.
Técnica:	<ul style="list-style-type: none">▪ Invocar cada método de acesso ao banco de dados, alimentando cada um com dados válidos e inválidos.▪ Inspeccionar o banco de dados e verificar se os dados nas tabelas estão de acordo com as ações realizadas
Critério de Finalização:	Todos os métodos e processos de acesso à base de dados funcionam como projetados e sem nenhuma corrupção de dados.
Considerações Especiais:	<ul style="list-style-type: none">▪ O teste pode necessitar de um ambiente de desenvolvimento ou drivers de SGBD para inserir ou modificar os dados diretamente na base de dados.▪ Processos devem ser invocados manualmente

3.1.2 Teste de Função

Objetivo do Teste:	Garantir que as funcionalidades do sistema, especificadas nos casos de usos, estão gerando os resultados esperados.
Técnica:	Executar cada caso de uso funcional através de seu fluxo principal e secundário, usando dados válidos e inválidos, para verificar o seguinte: <ul style="list-style-type: none">▪ Os resultados esperados ocorrem quando dados válidos são usados.▪ As mensagens de erro ou aviso apropriadas são exibidas quando dados inválidos são usados.▪ Cada regra de negócio é aplicada apropriadamente.
Critério de Finalização:	<ul style="list-style-type: none">▪ Todos os testes planejados foram executados.▪ Todos os defeitos identificados foram tratados.

Considerações Especiais:	Nenhum
--------------------------	--------

3.1.3 Teste da Interface do Usuário

Objetivo do Teste:	<ul style="list-style-type: none"> ▪ Verificar se a navegação através dos alvos de teste reflete as funções e os requisitos do negócio apropriadamente. ▪ Objetos e características da janela, tais como menus, tamanho, posição, estado e foco conformam-se aos padrões.
Técnica:	<ul style="list-style-type: none"> • Criar ou modificar os testes para cada janela para verificar a navegação e os estados de objeto apropriados para cada janela e objetos da aplicação. • Observar grupos de usuários usando a interface, analisando a taxa de aprendizado dos mesmos com o sistema e a aceitação da interface pelos usuários.
Critério de Finalização:	<ul style="list-style-type: none"> • É verificado que cada janela permanece consistente com a versão de comparação ou dentro de padrões aceitáveis. • É verificado que o usuário consegue usar a interface sem precisar de treinamento e a considera agradável.
Considerações Especiais:	Nem todas as propriedades para objetos personalizados e terceirizados podem ser acessadas.

3.1.4 Teste de Performance

Objetivo do Teste:	<p>Verificar os comportamentos do sistema em relação à sua performance sob as seguintes condições:</p> <ul style="list-style-type: none"> • Carga de trabalho normal prevista • Carga de trabalho no pior caso prevista
--------------------	---

Técnica:	<ul style="list-style-type: none"> ▪ Usar Procedimentos de Teste desenvolvidos para Teste da Função e Ciclo de Negócio. ▪ Scripts devem ser rodados em uma máquina (melhor caso para comparar um único usuário, uma única transação) e ser repetidas com múltiplos clientes (virtual ou real, ver Considerações Especiais abaixo).
Critério de Finalização:	<ul style="list-style-type: none"> ▪ Único usuário ou transação: finalização com sucesso sem nenhuma falha e dentro do tempo especificado ▪ Múltiplos usuários ou transações: finalização bem sucedida sem qualquer falha e dentro do tempo especificado.
Considerações Especiais:	<p>Um teste abrangente de performance inclui ter uma carga de trabalho no servidor.</p> <p>Há vários métodos que podem ser usados para executar isso, incluindo:</p> <ul style="list-style-type: none"> ▪ “Direcionar transações” diretamente para o servidor, usualmente na forma de chamadas SQL. ▪ Criar carga de usuário “virtual” para simular muitos clientes, normalmente várias centenas. Ferramentas de Emulação de Terminal Remoto (RTE) são usadas para atingir essa carga. Essa técnica também pode ser usada para carregar uma rede com “tráfego”. ▪ Usar múltiplos clientes físicos, cada um rodando scripts de teste para gerar uma carga no sistema. <p>O teste de performance deve ser executado em uma máquina dedicada ou em um tempo dedicado. Isso permite controle total e mensuração precisa.</p> <p>As bases de dados usadas para o Teste de Performance devem ser ou do tamanho real ou proporcionalmente iguais.</p>

3.1.5 Teste de Carga

Objetivo do Teste:	Verificar o funcionamento do sistema sobrecarregado.
--------------------	--

Técnica:	Usar testes desenvolvidos para o Teste do Ciclo de Negócio ou Função, aumentando o tamanho da carga de dados inseridos e verificados no servidor, até encontrar o limite de funcionamento do servidor. Verificando a seguir a compatibilidade dos dados e as regras de negócios.
Critério de Finalização:	Uma sobrecarga possível para o ambiente para o qual o ambiente está sendo desenvolvido deve ser suportada corretamente e sem comprometer a eficiência do sistema.

3.1.6 Teste de Segurança e Controle de Acesso

Objetivo do Teste:	Verificar que apenas aqueles usuários com acesso ao sistema e aplicações têm permissão de acessá-los. Este usuário pode acessar apenas aquelas funções ou dados para os quais o seu tipo de usuário tem permissão.
Técnica:	<ul style="list-style-type: none"> • Segurança do Nível de Aplicação: Identifique e liste cada tipo de usuário e as funções ou dados para os quais cada tipo tem permissão. • Crie testes para cada tipo de usuário e verifique cada permissão criando transações específicas para cada tipo de usuário. • Modifique o tipo de usuário e repita os testes para os mesmos usuários. Em cada caso, verifique que funções ou dados adicionais estão corretamente disponíveis ou negados. • Acesso de Nível de Sistema: Ver Considerações Especiais abaixo.
Critério de Finalização:	Para cada tipo de ator conhecido as funções ou dados apropriados estão disponíveis, e todas as transações funcionam como esperado e rodam nos Testes de Função anteriores.
Considerações Especiais:	O Acesso ao sistema deve ser revisado ou discutido com o administrador de rede ou de sistema apropriado. Esse teste pode não ser necessário já que ele pode ser uma função da administração da rede ou sistema.

3.1.7 Teste de Instalação

Objetivo do Teste:	Verifique que os alvos de teste instalam apropriadamente em cada configuração de hardware necessária sobre as seguintes condições: <ul style="list-style-type: none">• Uma nova instalação, em uma nova máquina, que nunca fora anteriormente instalada.• Atualização, numa máquina onde o software já fora previamente instalado, para a mesma versão.• Atualização, numa máquina que já disponha do software instalado, de uma versão mais velha.
Técnica:	Começar ou executar a instalação.
Critério de Finalização:	As transações do software executam de forma bem sucedida, sem falha.
Considerações Especiais:	Saber antecipadamente quais transações do software devem ser selecionadas para abranger um teste de confiança de que a aplicação foi instalada de forma bem sucedida e que nenhum componente importante de software está faltando.

3.2 Ferramentas

As seguintes ferramentas serão empregadas para esse projeto:

4. Equipe

Contém descrição da equipe e da infraestrutura utilizada para o desenvolvimento das atividades de testes, incluindo: pessoal, equipamentos, software de apoio, materiais, dentre outros. Isto visa garantir uma estrutura adequada para a execução das atividades de testes previstas no plano.

4.1 Equipe de teste

Essa tabela mostra as suposições de recrutamento para o projeto.

Recursos Humanos		
Trabalhador	Recursos Mínimos	Responsabilidades Específicas ou
	Recomendados	Comentários

Gerente de Teste, Gerente do Projeto de Teste	Gustavo Luna	Fornece supervisionamento gerencial. Responsabilidades: <ul style="list-style-type: none"> ▪ provê direcionamento técnico ▪ adquire recursos apropriados ▪ fornece relatórios de gerenciamento
Test Designer	Gustavo Soares Gustavo Luna	Identifica, prioriza, e implementa os casos de teste. Responsabilidades: <ul style="list-style-type: none"> • gera o plano de teste • cria o modelo de teste • avalia a efetividade do esforço de teste
Testador	Isabela Assis Gustavo Luna Isabelly Andrade	Executa os testes. Responsabilidades: <ul style="list-style-type: none"> ▪ executar os testes ▪ registrar os resultados ▪ reestabelecer-se dos erros ▪ documentar solicitações de mudança
Administrador do Sistema de Teste	Isabelly Andrade Isabela Assis	Garante que o ambiente e os bens de teste sejam gerenciados e mantidos. Responsabilidades: <ul style="list-style-type: none"> ▪ administrar o sistema de gerenciamento teste ▪ instalar e gerenciar o acesso do trabalhador ao sistema de testes
Gerente do Banco de Dados, Administrador do Banco de Dados	Isabelly Andrade	Garante que o ambiente e bens de teste de dados (banco de dados) sejam gerenciados e mantidos. Responsabilidades: <ul style="list-style-type: none"> ▪ administrar os dados de teste (base de dados)

Designer	Gustavo Campos José Ângelo Tobias José	Identifica e define as operações, atributos, e associações das classes de teste. Responsabilidades: <ul style="list-style-type: none"> ▪ identificar e definir as classes de teste ▪ identificar e definir os pacotes de teste
Implementador	Isabela Assis Gustavo Luna Gustavo Soares	Implementa e faz os testes unitários das classes e pacotes de teste. Responsabilidades: <ul style="list-style-type: none"> ▪ cria as classes e pacotes de teste implementados no modelo de teste

4.2 Infraestrutura

A tabela seguinte expõe os recursos do sistema para o projeto de teste.

Recursos do Sistema
Servidor de Banco de Dados MySQL DataBase Server VSCode Canvas Figma Framework CodeIgniter 4.0
Terminais Clientes 1 PC 1 Dispositivo mobile
Repositório de Testes 1 PC 2 PCs de Desenvolvimento de Teste

5. Cronograma

Contém uma descrição de marcos importantes (milestones) das atividades (incluindo as datas de início e fim da atividade). Apenas marcos relevantes

devem ser listados, ou seja, aqueles que contribuirão nas atividades de testes. Por exemplo: projeto de testes, execução de testes ou avaliação de testes.

ATIVIDADE	Início	Final
Planejamento de Testes	20/02/2023	25/03/2023
Projetar Testes	26/03/2023	08/04/2023
Implementar Testes	10/04/2023	24/05/2023
Execução de Testes	25/05/2023	05/05/2023
Avaliação de Testes	07/06/2023	14/06/2023

6. Glossário – Definições e Siglas

6.1 CRUD – Inserir, Pesquisar, Editar e Deletar.

7. Descrição do Mini-mundo do Projeto

Os alunos do curso de engenharia de software perceberam uma dificuldade na divulgação de eventos locais após tentarem marcar de sair e não encontrarem de forma organizada os eventos ocorrendo na região. Dito isso surgiu a ideia de criar um software vendo a necessidade de divulgação dos eventos locais.

O funcionamento seria da seguinte maneira: O cliente final acessa o software, que utilizando sua localização geográfica consegue informar os eventos acontecendo na região - estes eventos seriam artistas locais em estabelecimentos comerciais-. A partir disso, ele escolhe o evento que quer ir, e consegue ver se o evento é pago ou gratuito. Caso seja pago, ele consegue comprar os ingressos diretamente pelo aplicativo.

O cliente consegue comprar ingressos individuais em diversas modalidades (Inteira, meia entrada, ou pacote, que seriam 4 ingressos em valor promocional.) Após a escolha do tipo de ingresso, o cliente é redirecionado para o carrinho, onde visualiza uma visão geral, com os eventos que ele está comprando, o valor unitário de cada ingresso, o valor total dos ingressos e a finalização da compra é feito em um gateway parceiro de pagamento.

O Promotor de eventos, que seria a pessoa que cadastraria os eventos no software, tem a necessidade da geração de relatórios de vendas bem como um filtro para ver os nomes dos usuários que compraram os ingressos para o evento.

8. Materiais de Referência

Outros sites de venda de ingresso.

9. Requisitos de Software

9.1 Descrição dos Atores

Num	Nome	Descrição	Frequência de Uso
1	Administrador	O administrador terá acesso total ao site, conseguindo, de forma exclusiva, realizar o CRUD dos clientes e dos promotores de eventos bem como de todas as demais funcionalidades do site.	Diária
2	Promotor de Eventos	O promotor terá um perfil exclusivo e será responsável por realizar o CRUD dos eventos que acontecerão em seu estabelecimento, assim como emitir um relatório com o nome dos clientes que realizaram a compra do evento cadastrado. Para realizar o gerenciamento é obrigatório estar cadastrado	Ocasionalmente
3	Cliente final	O cliente terá acesso aos eventos já cadastrados no sistema, onde poderá realizar a compra. Para fazer a compra é obrigatório estar cadastrado.	Ocasionalmente

9.2 Requisitos funcionais

Ordem	Nome da função	Descrição	Ator relacionado
1	Cadastrar Usuário	Permite o cadastramento do cliente ou promotor de eventos no sistema.	1,2,3
2	Realizar login	Necessário para o controle de acesso ao sistema e para atribuir a cada tipo de usuário as suas respectivas funcionalidades.	1,2,3
3	Gerenciar Usuários	Permite inserir, alterar, excluir e pesquisar os usuários. É necessário o login dos atores para que o CRUD seja realizado.	1
4	Gerenciar Eventos	Permite inserir, alterar, excluir e pesquisar os eventos correspondentes. É necessário o login dos atores para que o CRUD seja realizado.	1, 2
5	Comprar Ingresso	Permite comprar, alterar e pesquisar os ingressos que foram obtidos. É necessário o login dos atores para que seja realizado. É necessário o login dos atores para que a atividade seja realizada.	3
6	Categoria do Evento	Permite inserir, alterar, excluir e pesquisar as categorias dos eventos. É necessário o login dos atores para que o CRUD seja realizado.	1, 2
7	Gerar Relatório	Permite gerar um relatório sobre as vendas respectivas de cada evento.	2

		É necessário o login dos atores para que o CRUD seja realizado.	
--	--	---	--

9.3 Requisitos não funcionais

RNF 01 – O software deverá ser web.

RNF 02 – Os relatórios do sistema devem estar no formato PDF.

RNF 03 – O sistema deverá ser responsivo para funcionar em sistemas mobile.

9.4 Detalhamento dos casos de uso

Nome do Caso de Uso	Cadastrar Usuário
Atores relacionados	Todos os atores
Pré-condições	1. O sistema está iniciado e a conexão com o banco de dados foi testada.
Fluxo Principal	
Ações do Ator	Ações do Sistema
	1. Exibe a interface de cadastro
2. Informa os dados necessários para realizar o cadastro	
3. Aciona o botão “avançar”	
	4. Verifica os dados de login
5. Aciona o botão “finalizar”	
	6. Envia um e-mail de confirmação

Nome do Caso de Uso	Realizar Login
Atores relacionados	Todos os atores
Pré-condições	1. O sistema está iniciado e a conexão com o banco de dados foi testada.
Fluxo Principal	
Ações do Ator	Ações do Sistema
	1. Exibe a interface de Login
2. Informa os dados de login	
3. Aciona o botão “entrar”	
	4. Verifica os dados de login
	5. Exibe a tela inicial do site

Nome do Caso de Uso	Gerenciar Usuários
Atores relacionados	Administrador
Pré-condições	1. Ter realizado login no sistema
Fluxo Principal	
Ações do Ator	Ações do Sistema
	1. Exibe a interface do sistema
2. Aciona a funcionalidade de exibir usuários	
	3. Exibe a interface de usuários
4. Realiza o CRUD do usuário desejado	
5. Aciona o botão correspondente ao CRUD que foi realizado (Inserir, Pesquisar, Deletar, Editar e Salvar)	
	6. Salva as alterações feitas no sistema

Nome do Caso de Uso	Gerenciar eventos
----------------------------	-------------------

Atores relacionados	Promotor de eventos
Pré-condições	1. Ter realizado login no sistema
Fluxo Principal	
Ações do Ator	Ações do Sistema
	7. Exibe a interface do sistema
8. Aciona a funcionalidade de cadastrar eventos	
	9. Exibe a interface de eventos
10. Realiza o CRUD do evento desejado	
11. Aciona o botão correspondente ao CRUD que foi realizado (Inserir, Pesquisar, Deletar, Editar e Salvar)	
	12. Salva as alterações feitas e lança na página de promoção de eventos.

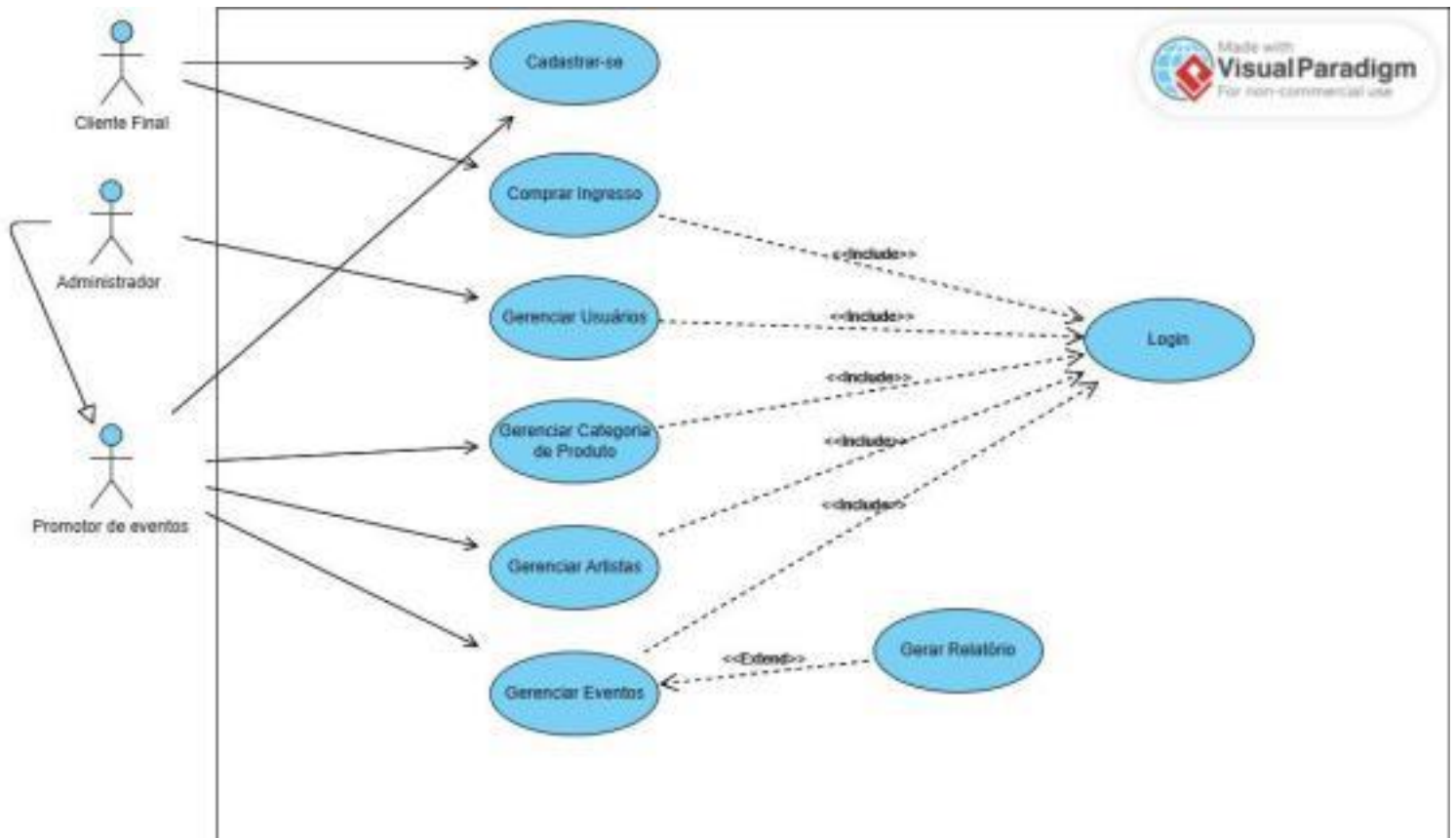
Nome do Caso de Uso	Comprar ingressos
Atores relacionados	Cliente final
Pré-condições	1. Realizar o Login no sistema
Fluxo Principal	
Ações do Ator	Ações do Sistema
	1. Exibe a interface do sistema
2. Clica no evento desejado	
	3. Exibe a interface do evento
4. Seleciona a quantidade de ingressos	
	5. Calcula e exibe o valor total dos ingressos
6. Aciona o botão “finalizar compra”	
	7. Redireciona para o gateway de pagamento.

Nome do Caso de Uso	Categoria do evento
Atores relacionados	Administrador e Promotor de Eventos
Pré-condições	1. Realizar login no sistema
Fluxo Principal	
Ações do Ator	Ações do Sistema
	1. Exibe a interface do sistema
2. Aciona a funcionalidade de categoria de eventos	
	3. Exibe a interface de cadastrar categoria
4. Realiza o CRUD da categoria	
5. Aciona o botão correspondente ao CRUD que foi realizado (Inserir, Pesquisar, Deletar, Editar e Salvar)	
	6. Salva as alterações feitas

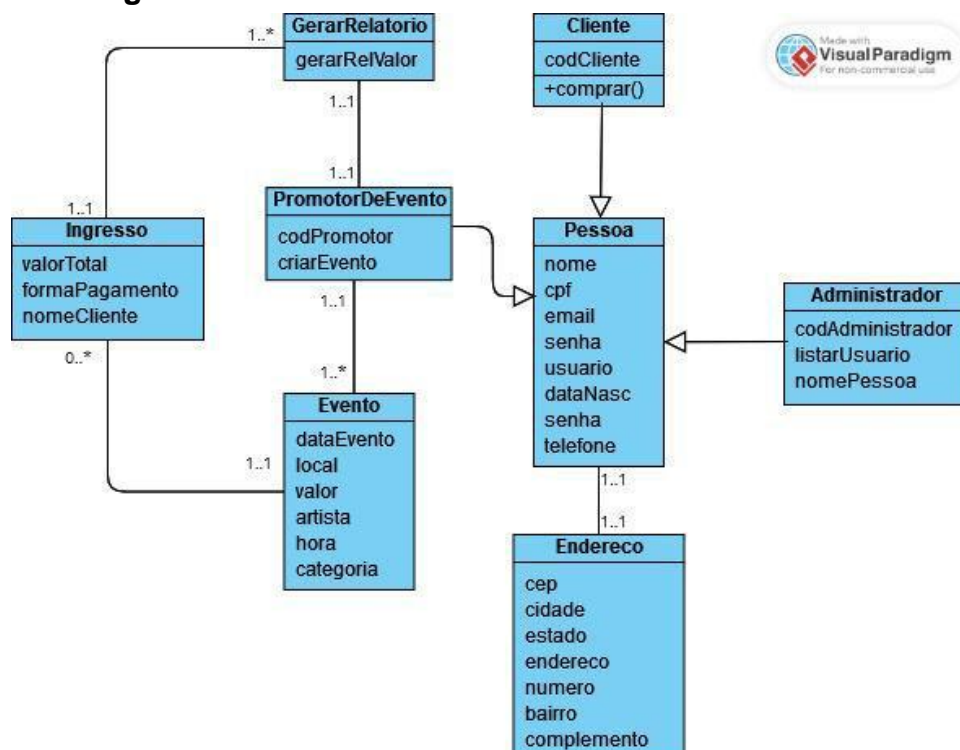
Nome do Caso de Uso	Gerar Relatório
Atores relacionados	Promotor de Eventos
Pré-condições	1. Realizar login no sistema 2. Gerar relatório no dia de cada evento.
Fluxo Principal	
Ações do Ator	Ações do Sistema
	1. Exibe a interface do sistema
2. Aciona a funcionalidade de relatório	
	3. Exibe a interface de gerar relatórios
4. Aciona o botão gerar relatório	.
	5. Gera o relatório de usuários que compraram o ingresso.

10. Diagramas

10.1 Diagrama de Casos de Uso

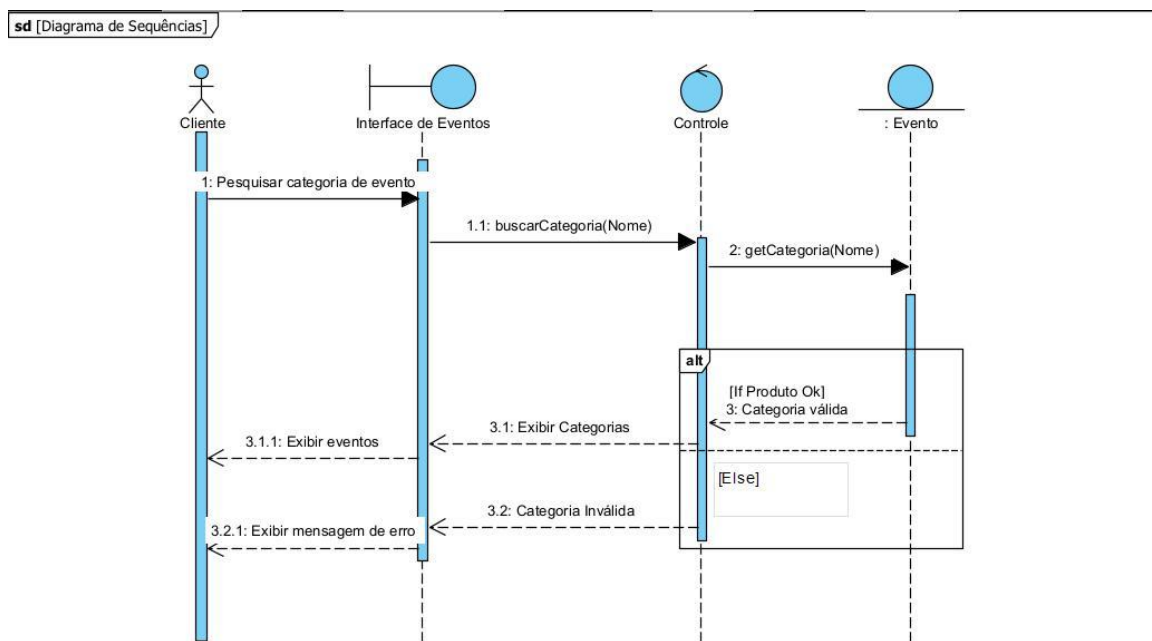


10.2 Diagrama de Classes



10.3 Diagramas de Sequência

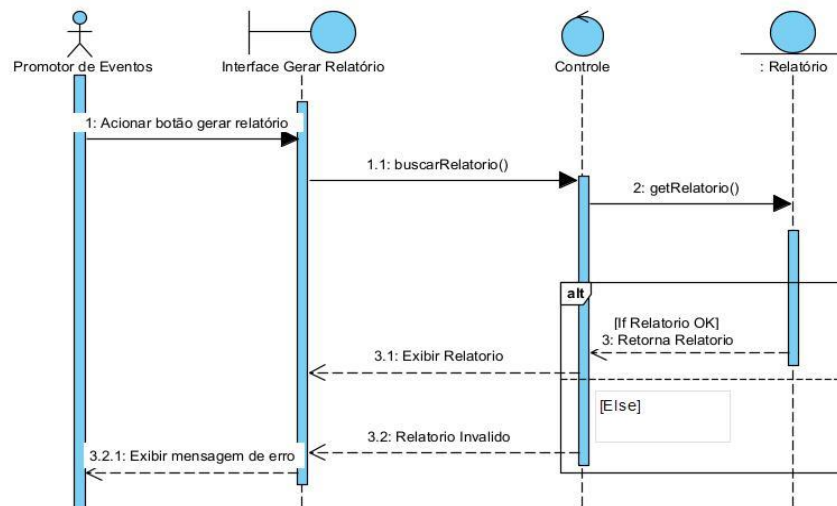
O processo modelado pelo diagrama é o de pesquisar evento por categoria, no qual o usuário pesquisa por uma categoria, o sistema busca a categoria por nome, o controle medeia a busca por essa categoria na entidade evento e, caso ele exista, retorna-a como válida. Caso a categoria não exista, é apresentado para o usuário uma mensagem de categoria inválida no sistema e uma mensagem é retornada para o usuário informando a categoria pesquisada é inválida.



O processo modelado pelo diagrama é o de gerar relatório de compras de ingresso, no qual o promotor de eventos aciona o botão gerar relatório, o sistema busca o relatório de todos os usuários que compraram o ingresso, o controle medeia a busca por esse relatório na entidade relatório e, caso ele esteja ok (seja gerado assim que o tempo limite de venda dos ingressos para o evento for esgotado), retorna-o como válido. Caso o promotor tente gerar um relatório antes do tempo de encerramento, é

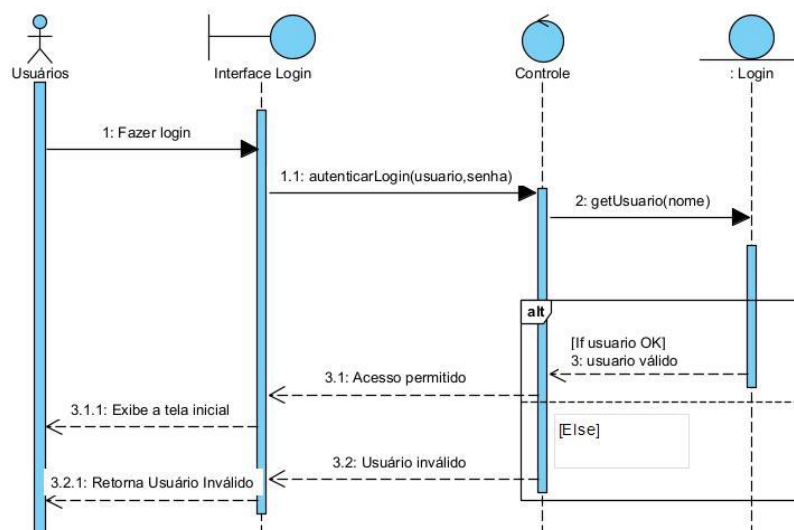
apresentado a ele uma mensagem informando que não é possível gerar um relatório antes do tempo estabelecido para o encerramento da venda de ingressos.

sd [Diagrama de Sequências]

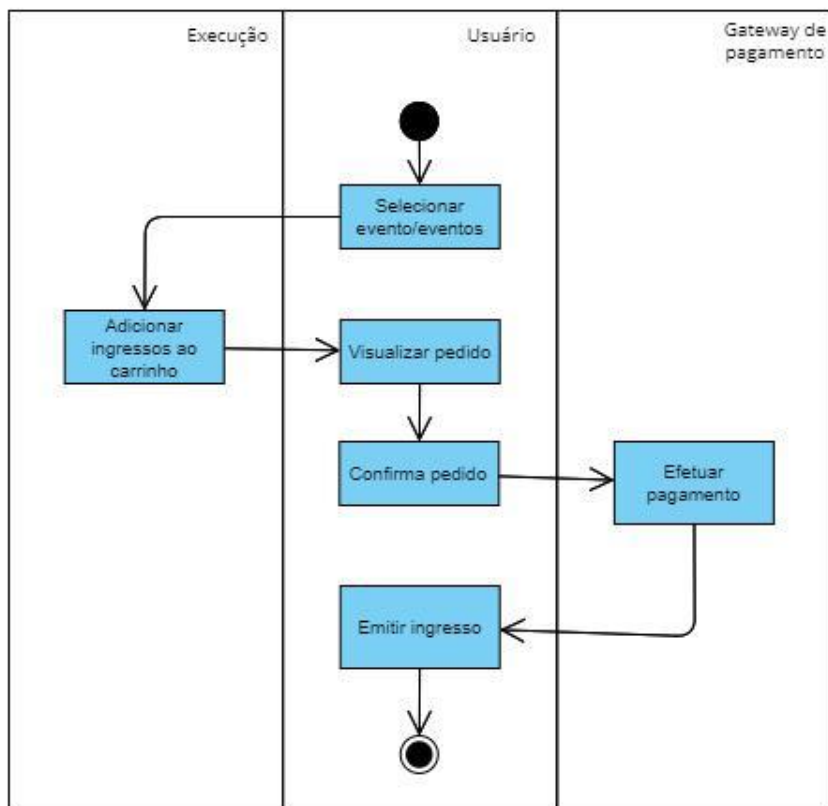


O processo modelado pelo diagrama é o de realizar login no sistema, no qual o usuário digita o usuário e senha, o sistema busca autenticar o login através do usuário e senha, o controle medeia a busca pelo usuário na entidade Login e, caso ele exista, retorna-o como válido, o acesso ao sistema é liberado e a interface do sistema é apresentada ao usuário, caso contrário uma mensagem é retornada para o usuário informando que as informações digitadas estão incorretas

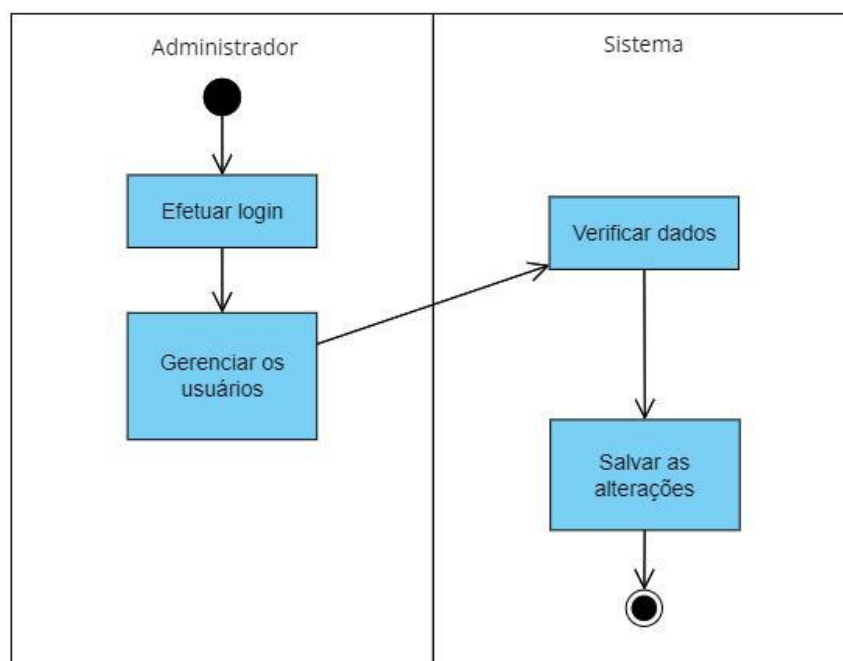
sd [Diagrama de Sequências]



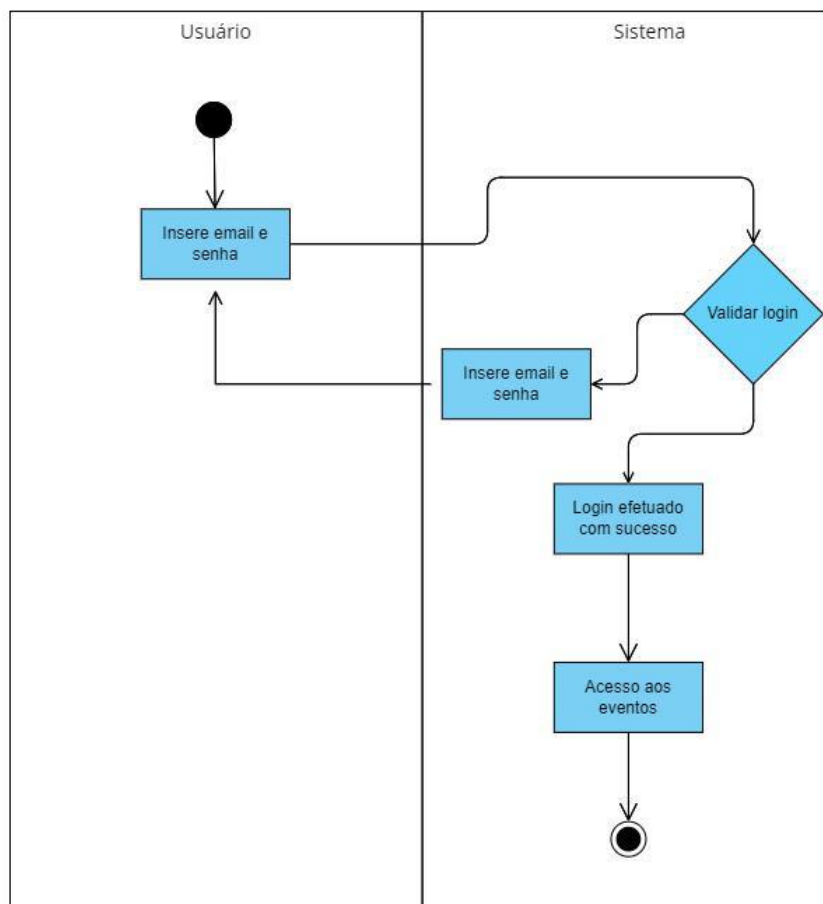
10.4 Diagramas de Atividades



O fluxo modelado pelo diagrama é o de efetuar a compra de um ingresso, na qual o usuário realiza o pedido, adiciona ao carrinho, confirma o pedido, realiza o pagamento pelo gateway de pagamento e emite o ingresso.

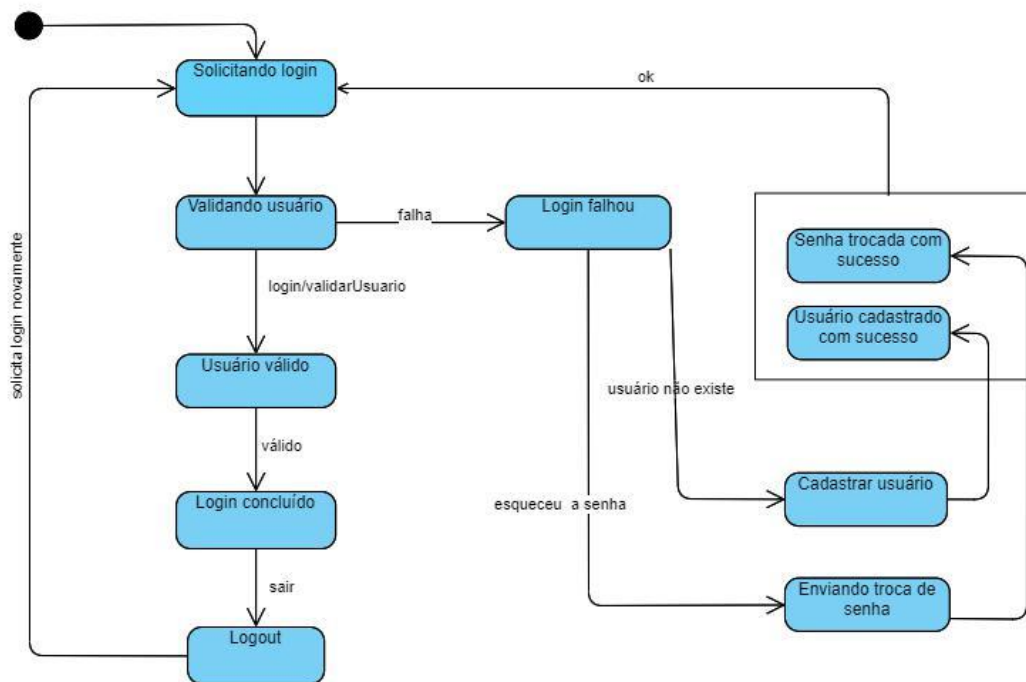


O processo modelado pelo diagrama é o gerenciar usuários, no qual é feito único e exclusivamente pelo administrador, que ao fazer login, preenche os dados necessários com as informações do usuário que ele tem. Ao finalizar o preenchimento, o sistema verifica se os dados estão corretos (ex. CPF) e então salva as alterações do usuário no sistema (Login e senha) para que o usuário possa utilizar os recursos necessários para utilizar o site/app.



Para poder entrar no sistema o usuário precisa inserir o seu e-mail e sua senha, o sistema vai verificar se está correto. Caso esteja incorreto o usuário vai precisar novamente de inserir suas informações, ou se for o caso, alterar a senha, se estiver correto o usuário vai ser logado no sistema.

10.5 Diagramas de Máquina de Estados



10.6 Diagrama de objetos

O diagrama de objetos é ligado ao diagrama de classes, como uma especialização do diagrama de objetos, portanto, postei um recorte do diagrama de classes que foi utilizado para o nosso diagrama de objetos.

Diagrama de classes:

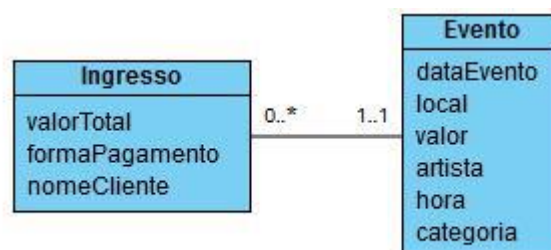
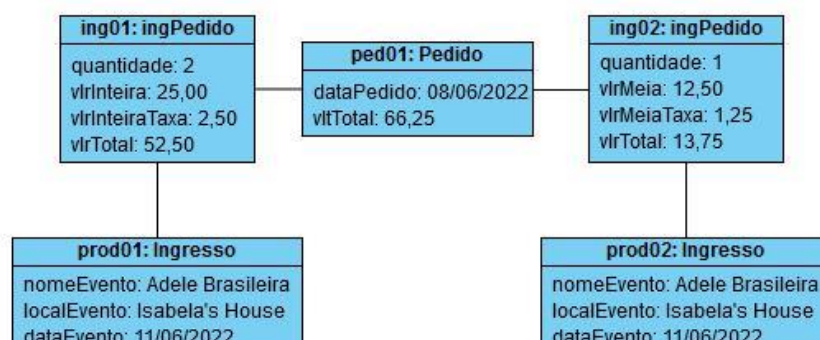


Diagrama de objetos:



11. Teste unitário

Nós realizamos um teste unitário utilizando a ferramenta JUnit, um framework de testes unitários muito popular para Java. O objetivo do teste foi verificar a funcionalidade de exibição dos eventos com base na localização geográfica do usuário.

Iniciamos o teste abrindo o software e concedendo a permissão necessária para acessar a localização geográfica do dispositivo. Utilizando o JUnit, estruturamos e executamos os casos de teste para verificar se a página inicial do software exibia corretamente os eventos locais disponíveis na região em que nos encontramos. Utilizando os recursos do JUnit, selecionamos um evento específico para verificar se todas as informações relevantes eram exibidas corretamente, incluindo o nome do evento, localização, data e preço. Também verificamos se a opção de compra de ingressos era exibida apenas para eventos pagos, garantindo que os eventos gratuitos não apresentassem essa opção.

Ao selecionar um evento gratuito, utilizamos o JUnit para verificar se a opção de compra de ingressos não era exibida, pois não era necessária nesse caso. Porém, ao selecionar um evento pago, verificamos se a opção de compra de ingressos era exibida corretamente.

Continuamos o teste utilizando o JUnit para verificar se a funcionalidade de compra de ingressos redirecionava corretamente o usuário para o carrinho de compras. Uma vez no carrinho, confirmamos se as informações dos eventos selecionados eram exibidas corretamente, incluindo o valor unitário de cada ingresso e o valor total da compra.

Por fim, utilizando o JUnit, garantimos que a finalização da compra fosse feita corretamente utilizando um gateway parceiro de pagamento, que não está implementado neste momento. Durante o teste, identificamos um erro relacionado à exibição incorreta da opção de compra de ingressos para eventos gratuitos, e registramos esse problema para ser corrigido posteriormente.


O uso do JUnit no teste unitário contribuiu para a confiabilidade do software, permitindo que cada parte individual fosse testada e validada antes de ser integrada ao sistema como um todo. Após a correção do erro encontrado, continuaremos a desenvolver o software, implementando as funcionalidades de compra de ingressos e geração de relatórios para atender às necessidades dos usuários, como o promotor de eventos que deseja acompanhar as vendas e ter acesso aos nomes dos usuários que compraram os ingressos para cada evento.

12. Teste de Integração

12.1 Testes de Interface

Modelo utilizado para realizar o checklist de interface do usuário:

Checklist de interface do Usuário:				
<u>Navegação:</u>	Ruim ←		Neutro	Excelente →
O menu de navegação é claramente rotulado e fácil de acessar?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
O usuário consegue encontrar facilmente os ingressos que deseja comprar?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
A navegação permite ao usuário voltar facilmente para a página anterior ou para a página inicial?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
O sistema deve fornecer feedback visual claro quando uma ação é concluída?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<u>Usabilidade:</u>	Ruim ←		Neutro	Excelente →
O sistema permite que o usuário faça alteração como adicionar ou remover itens do carrinho de compras?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Você identifica metáfora ou objeto em todo programa?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
As categorias de produtos ou serviços são organizadas de forma lógica e fácil de seguir?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
A interface é intuitiva e fácil de ser utilizada?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
O usuário consegue escolher facilmente e verificar as informações dos ingressos?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
O usuário consegue facilmente escolher os ingressos e verificar a somatória total dos preços?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<u>Disponibilidade de informações:</u>	Ruim ←		Neutro	Excelente →
O sistema fornece informações precisas sobre os eventos?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<u>Acessibilidade:</u>	Ruim ←		Neutro	Excelente →
A interface do sistema é acessível a todos os usuários?	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>



12.2 Teste de funcionalidades

Plano de Testes - Login do Cliente

Cenário principal:

- O cliente informa suas informações de login corretamente e consegue entrar na plataforma.

Cenários alternativos:

1. Senha inválida
 - O cliente insere uma senha inválida e recebe uma mensagem de erro

informando que a senha é inválida.

2. Confirmação de senha:
 - Senhas iguais: deve ser aceito e permitir a continuidade do cadastro.
 - Senhas diferentes: deve ser rejeitado e mostrar uma mensagem de erro.

Plano de Testes - Cadastro de Login do Cliente

- #### Cenário principal:
- O cliente informa todas as informações corretamente e consegue criar uma conta na plataforma.

Cenários alternativos:

1. Senha de confirmação diferente da senha informada
 - O cliente informa uma senha de confirmação diferente da senha informada e recebe uma mensagem de erro informando que as senhas não coincidem.
2. Informações de endereço incompletas
 - O cliente não preenche informações obrigatórias como número, bairro, cidade ou estado e recebe uma mensagem de erro informando que todas as informações são obrigatórias.

Tópicos de teste:

1. Confirmação de e-mail:

- E-mails iguais: deve ser aceito e permitir a continuidade do cadastro.
 - E-mails diferentes: deve ser rejeitado e mostrar uma mensagem de erro.
2. Confirmação de senha:
- Senhas iguais: deve ser aceito e permitir a continuidade do cadastro.
 - Senhas diferentes: deve ser rejeitado e mostrar uma mensagem de erro.
3. Preenchimento de informações de endereço:
- Todas as informações de endereço obrigatórias: deve ser aceito e permitir a continuidade do cadastro.
 - Faltando informações obrigatórias: deve ser rejeitado e mostrar uma mensagem de erro.

Plano de Testes - Adição de itens no Carrinho de Compras

Cenário principal:

O usuário seleciona o(s) ingresso(s) desejado(s) e adiciona ao carrinho de compras.

Cenários alternativos:

Quantidade de ingressos selecionada inválida

O usuário seleciona uma quantidade de ingressos menor do que 1 e recebe uma mensagem de erro informando que a quantidade selecionada é inválida.

Adicionar ingressos com sucesso

O usuário seleciona um ou mais ingressos e adiciona com sucesso ao carrinho de compras.

Adicionar ingressos com sucesso após atualização da página

O usuário seleciona um ou mais ingressos, atualiza a página e confirma que os ingressos foram adicionados corretamente.

Adicionar ingressos a partir da página de detalhes do evento

O usuário navega até a página de detalhes do evento e adiciona um ou mais ingressos ao carrinho de compras a partir dessa página.

Remover ingresso do carrinho

O usuário remove um ingresso do carrinho de compras com sucesso.

Alterar a quantidade de ingressos no carrinho

O usuário altera a quantidade de ingressos no carrinho com sucesso.

Tópicos de teste:

Seleção de ingressos:

Ingressos disponíveis: deve ser possível selecionar um ou mais ingressos disponíveis e adicionar ao carrinho de compras.

Quantidade de ingressos:

Quantidade válida: deve ser possível selecionar uma quantidade de ingressos disponíveis e adicionar ao carrinho de compras.

Quantidade inválida: deve ser exibida uma mensagem de erro informando que a quantidade selecionada é inválida.

Adicionar e remover ingressos:

Adicionar ingressos: deve ser possível adicionar ingressos ao carrinho de compras e visualizá-los na lista de itens.

Remover ingressos: deve ser possível remover ingressos do carrinho de compras e visualizar a atualização da lista de itens.

Alterar a quantidade de ingressos:

Alterar quantidade: deve ser possível alterar a quantidade de ingressos no carrinho de compras e visualizar a atualização da lista de itens.

Página de detalhes do evento:

Adicionar ingressos: deve ser possível adicionar ingressos ao carrinho de compras a partir da página de detalhes do evento.

12.3 Teste de Segurança

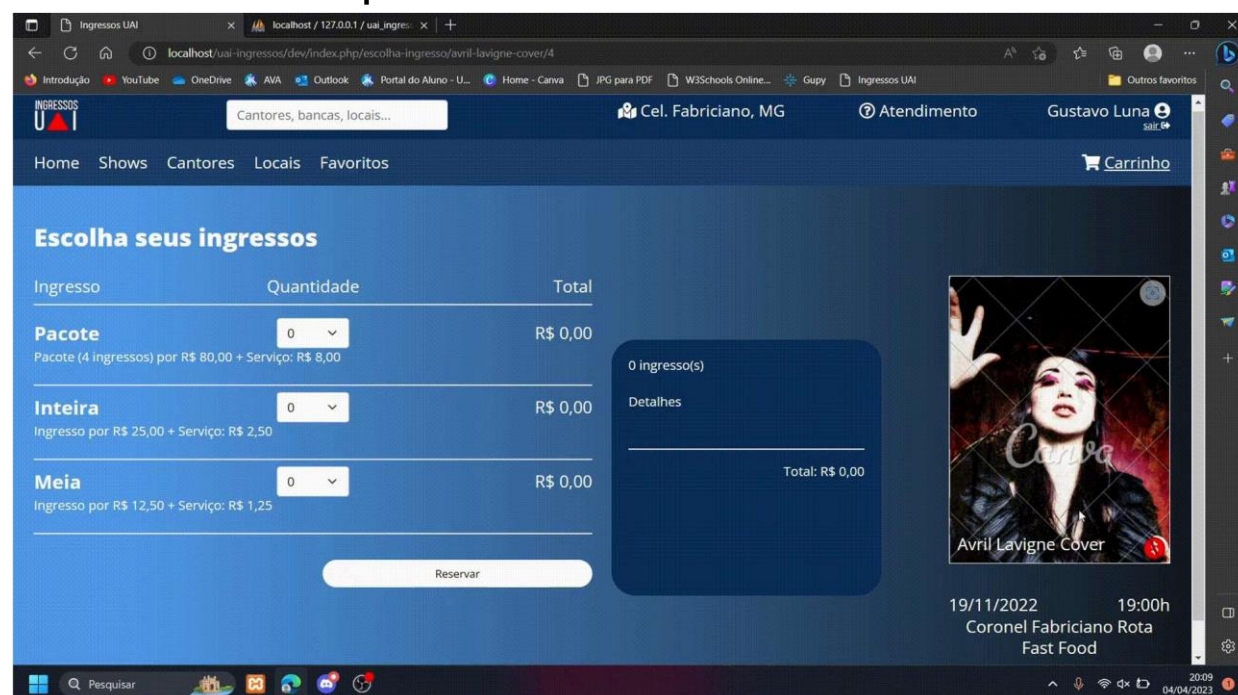
Fizemos os testes de segurança checando a entrada de clientes que já possuíam cadastro no site VS clientes que não possuíam cadastro no site. Foi-se constatado que: Àqueles que possuíam cadastro conseguiram entrar normalmente utilizando seu login e senha, os que não possuíam acesso, ao tentarem digitar um e-mail e senha qualquer não conseguiam entrar devido a mensagem de “e-mail ou senha inválidos” e àqueles que tentaram usar a funcionalidade de adicionar ingressos ao carrinho sem estarem logados, se depararam com o redirecionamento para a tela de login do site.

É importante ressaltar que o site ainda não possui um perfil de administrador, então essa funcionalidade não pode ser testada.

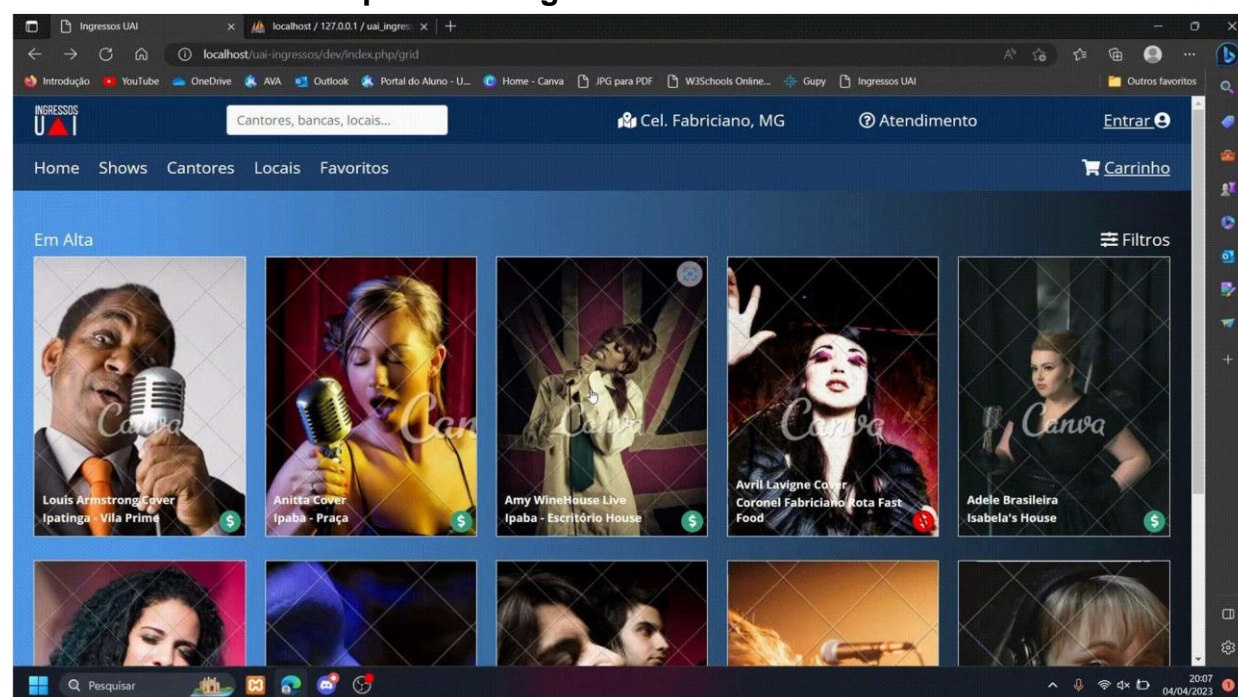
Quanto ao cadastro no sistema, assim que é realizado, o usuário consegue adicionar e remover os ingressos do carrinho, bem como verificar as informações detalhadas ao clicar sobre eles (horário, local e valor do evento).

Segue abaixo as imagens dos testes realizados na interface:

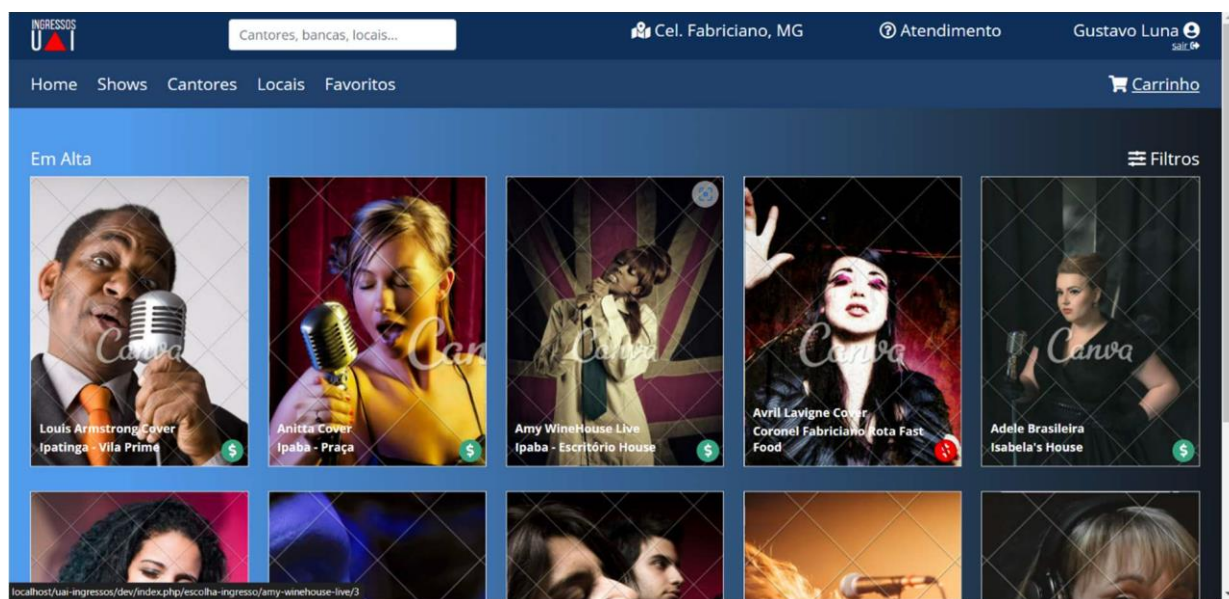
12.3.1 Carrinho de compras



12.3.2 Carrinho de compras sem login



12.3.3 Usuário e/ou senha incorretos



12.3.4 Login

The login form is set against a dark blue background. At the top, it features the "INGRESSOS UAI" logo and the heading "Entrar no Ingressos, UAI!". Below this, there is a white button labeled "Inscreva-se no Google". A separator "OU" is centered below the button. The form includes two white input fields for "email" and "senha". Below these fields is a white button labeled "Avançar" and a blue button labeled "Esqueceu a senha". At the bottom of the form, there is a link that says "Não tem uma conta? [Inscreva-se](#)". A large white box at the very bottom contains the error message "E-mail ou senha incorretos! Verifique e tente novamente." in red text.

12.3.5 Teste de segurança e controle de acesso

Objetivo do Teste:	Verificar que apenas aqueles usuários com acesso ao sistema e aplicações têm permissão de acessá-los. Este usuário pode acessar apenas aquelas funções ou dados para os quais o seu tipo de usuário tem permissão.
Técnica:	<ul style="list-style-type: none">• Usuário: Cliente.<ul style="list-style-type: none">✓ Cadastrar login;✓ Entrar com o login;✓ Adicionar eventos no carrinho;✓ Selecionar a quantidade de ingressos;✓ Remover ingressos do carrinho;.✓ Ir até a página de finalizar compras;.✓ Voltar para página inicial;.

Obs.: Apenas o usuário Cliente está disponível na codificação, visto que ela ainda não possui o perfil de administrador e promotor de eventos. Dessa forma, as funcionalidades testadas foram aquelas vinculadas aos clientes. Todas as funcionalidades citadas acima foram testadas e estão, no momento, funcionando em perfeito estado.

12.4 Teste de Aceitação

1) Quem são os testadores e sua função no projeto?

Fizemos o teste com pessoas que já tem costume de utilizar sites do ramo de venda de ingressos, sejam de shows, cinema, teatro. Também testamos com pessoas sem muita experiência do dia a dia com compras.

Com as pessoas com costume, validamos a facilidade nossas funcionalidades e se está intuitivo para quem já tem o costume de utilizar sites similares. Já com as pessoas sem muita experiência, além da validação também percebemos o que podemos melhorar para tornar mais atraente para quem não tem costume de fazer compras online em sites similares.

2) Funcionalidades testadas:

- Realizar login;
- Navegação;
- Adicionar ingressos no carrinho;
- Acessar o carrinho;
- Sair do usuário e logar novamente;

3) Erros encontrados / Inconsistências e em quais requisitos o sistema não atende:

- É possível adicionar 0 ingressos no carrinho e não dá mensagem de erro;
 - Não é possível aumentar o número de ingressos do mesmo evento dentro do carrinho. É necessário que volte ao evento e adicione mais ingressos novamente.
 - Ao adicionar ingressos do mesmo evento novamente, ao invés dele mesclar a quantidade, ele aparece duas vezes no carrinho com as quantidades diferentes, mesmo sendo ingressos para o mesmo evento e do mesmo tipo.
 - Quando se entra no evento gratuito, ele mostra o valor dos ingressos e soma o subtotal ao selecionar o número de ingressos para reservar, porém no carrinho o ingresso fica com valor zerado.
- Complemente com informações que julgar importantes.
 - Utilizando de base aulas de semestres anteriores, formulamos um roteiro de testes de usabilidade que foi utilizado para os testes regidos acima. Abaixo pode ser observado o roteiro de testes de usabilidade utilizado.

12.4.1 Roteiro de teste de usabilidade

As tarefas a serem realizadas pelos usuários participantes do teste de usabilidade devem ser construídas a partir dos objetivos previamente definidos.

Neste teste de usabilidade iremos testar o site Ingressos, Uai! Que estará rodando na máquina local e cada teste levará entre 5 e 10 minutos. A estrutura do teste será:

Anotações gerais

Produto: *Ingressos, Uai!*

Data:

Entrevistador:

Entrevista inicial

Nome:

Idade:

Experiência de uso da internet:

☐ Pouca experiência no uso de sites ☐ Média experiência no uso de sites ☐ Muita experiência no uso de sites

Sexo:

☐ Feminino ☐ Masculino

☐ Outros

Costuma utilizar sites de busca e venda de ingressos?

☐ Sim

☐ Não

Possui algum requisito de acessibilidade para o uso de sites?

☐ Sim - Qual? ☐ Não

Realização de Tarefas

Será entregue uma sequência de tarefas que deverão ser realizadas uma após a outra pelo usuário.

- Explicar que dará início ao teste.

- Explicar que ele receberá tarefas e estas podem ser feitas sem pressa.
- Lembrar que ele não está sendo testado e que sua ajuda é fundamental.
- Explicar que o principal é saber a opinião dele e não executar a tarefa em si.
- Avisar que ele pode “desistir” caso não consiga completar uma tarefa.
- Pedir para pensar em voz alta.
- Abrir o site.

Tarefa A

Página: Página de login do Ingressos, Uai!

Objetivo: Entender se a página permite que o usuário consiga efetuar o login sem passar por nenhuma dificuldade.

Descrição da tarefa: Realize o login no site.

Usuário conseguiu realizar a tarefa?

- ☐ Sim, com facilidade
- ☐ Sim, com dificuldade
- ☐ Não conseguiu realizar a tarefa

Quais foram suas impressões sobre esta tarefa?

Tarefa B (Fluxo 2)

Página: Página Inicial Do Ingressos, Uai!

Objetivo: Adicionar ao carrinho os ingressos de eventos que deseja participar.

Descrição da tarefa: **Navegue pelo site após logar com sua conta, adicione os eventos desejados no carrinho e vá até a parte de finalizar compras (o site ainda não possui um gateway de pagamento, então não é necessário adicionar informações de pagamento nele).**

Usuário conseguiu realizar a tarefa?

- ☐ Sim, com facilidade
- ☐ Sim, com dificuldade.
- ☐ Não conseguiu realizar a tarefa

A tarefa foi realizada de forma rápida e eficiente?

- ☐ Sim, com facilidade
- ☐ Sim, com dificuldade
- ☐ Não conseguiu realizar a tarefa

-

Quais foram suas impressões sobre esta tarefa?

Tarefa C

Página: Página de login

Objetivo: Sair do usuário e entrar com as mesmas credenciais definidas ao cadastrar o login

Descrição da tarefa: Saia do seu usuário e tente entrar novamente utilizando o e-mail e senha que foram informados durante a realização do cadastro.

Usuário conseguiu realizar a tarefa?

- ☐ Sim, com facilidade
- ☐ Sim, com dificuldade
- ☐ Não conseguiu realizar a tarefa

A tarefa foi realizada de forma rápida e eficiente?

- ☐ Sim, com facilidade
- ☐ Sim, com dificuldade
- ☐ Não conseguiu realizar a tarefa

Quais foram suas impressões sobre esta tarefa?

13. **Teste de caixa branca:**

O teste de caixa branca foi realizado para verificar se os mesmos ingressos, ao serem adicionados no carrinho eram mesclados ou apareciam de forma separada. Para isso, os seguintes passos foram realizados:

1. Analisar o código fonte para entender como a adição de ingressos no carrinho é implementada e como é tratada a adição de ingressos do mesmo evento.
2. Criar um caso de teste no PHPUnit para a funcionalidade em questão, utilizando o framework CodeIgniter 4.
3. Dentro desse caso de teste, criar uma função que simule a adição de dois ingressos do mesmo evento e do mesmo tipo no carrinho.
4. Verificar se o carrinho contém apenas um item com a quantidade correta de ingressos e com as informações do evento e do tipo de ingresso corretas.
5. Em seguida, tentar adicionar mais ingressos do mesmo evento e do mesmo tipo no carrinho.
6. Verificar se a quantidade do item no carrinho foi atualizada corretamente.
7. Finalmente, verificar se o valor total do carrinho é calculado corretamente, levando em consideração as quantidades e os valores de cada item.

Chegamos ao seguinte erro: o carrinho não atualiza a quantidade de ingressos ao adicionarmos mais ingressos do mesmo evento, resultando na adição de um novo item separado do mesmo evento e, portanto, gerando duplicidade de dados.

Teste de caixa preta:

Fizemos o teste de caixa preta na funcionalidade referente ao carrinho de compras, que envolveu: incluir, remover e adicionar mais ingressos no carrinho previamente utilizado. Realizando esses testes, foram encontrados os seguintes erros:

- É possível adicionar 0 ingressos no carrinho e não aparece nenhuma mensagem de erro.
- Não é possível aumentar o número de ingressos do mesmo evento dentro do carrinho. É necessário que volte ao evento e adicione mais ingressos novamente.
- Ao adicionar ingressos do mesmo evento novamente, ao invés dele mesclar a quantidade, ele aparece duas vezes no carrinho com as quantidades diferentes, mesmo sendo ingressos para o mesmo evento e do mesmo tipo.

- Quando se entra no evento gratuito, ele mostra o valor dos ingressos e soma o subtotal ao selecionar o número de ingressos para reservar, porém

no carrinho o ingresso fica com valor zerado.

14. Teste de Integridade de Banco de Dados

Para fazer os testes de integridade de banco de dados, nós utilizamos as ferramentas Selenium, JUnit e Postman para auxiliar nos testes. O Selenium automatizou a interação com o software, permitindo simular as ações do usuário, como inserir eventos, navegar pelo sistema, adicionar ingressos ao carrinho e realizar o pagamento. O JUnit verificou a corretude dos métodos e funcionalidades do software, garantindo que as informações dos eventos fossem armazenadas corretamente no banco de dados. O Postman testou a integração com o gateway de pagamento, verificando o envio correto de dados e o funcionamento do processo de pagamento. Foi avaliado o desempenho do sistema, garantindo que o banco de dados suportasse a carga esperada durante a divulgação de eventos locais.

Os resultados dos testes foram positivos. Todos os eventos foram cadastrados com sucesso, incluindo informações como nome, localização, artistas participantes, data, horário e status de pagamento. O software conseguiu identificar corretamente a localização do cliente final e exibir os eventos correspondentes à região. O processo de compra de ingressos foi concluído com sucesso, incluindo a seleção do evento e o pagamento. A integração com o gateway de pagamento funcionou corretamente, permitindo que o cliente final realizasse o pagamento de forma segura e eficiente. O promotor de eventos pôde gerar relatórios precisos de vendas, incluindo informações como nome do evento, quantidade de ingressos vendidos e dados dos usuários que adquiriram os ingressos.

15. Teste de Performance

Realizamos um teste de performance no sistema para avaliar sua capacidade de lidar com um aumento de acessos durante as vendas, garantindo que as integrações não se tornem pontos de gargalo. Utilizamos métricas coletadas dos testes de Carga e Estresse para análise, com o auxílio do aplicativo Apache JMeter. O objetivo foi verificar o desempenho do controlador cadastro e garantir que ele esteja funcionando adequadamente.

O controlador cadastro é responsável por processar o formulário de cadastro. Durante o teste, avaliamos o tempo de resposta do sistema, o tempo de processamento do controlador e a capacidade de lidar com várias solicitações simultâneas.

Com base nos resultados obtidos, podemos concluir que o controlador cadastro está performando corretamente. Durante o teste, não foram identificados atrasos significativos ou erros no processamento das solicitações de cadastro.

16. Teste de Carga

Nós decidimos realizar um teste de carga no software de divulgação de eventos locais para avaliar seu desempenho sob condições de carga intensa. Nosso objetivo era garantir que o sistema pudesse lidar com uma grande quantidade de usuários simultâneos buscando eventos.

Para realizar o teste, escolhemos a ferramenta Apache JMeter devido à sua ampla utilização e recursos abrangentes. Configuramos diversos cenários de carga que simularam diferentes atividades dos usuários.

Primeiro, definimos um cenário em que 500 usuários simulados acessavam o software simultaneamente para procurar eventos na região. Cada usuário fazia uma busca por eventos a cada 5 segundos. Esse cenário representava um cenário de carga moderada.

Em seguida, criamos um cenário de carga pesada com 1000 usuários virtuais selecionando eventos e visualizando detalhes sobre eles simultaneamente. Isso representava uma situação de alta demanda no momento em que os usuários estavam navegando pelos eventos.

Por fim, configuramos um cenário em que 200 usuários virtuais, representando promotores de eventos, adicionavam novos eventos ao sistema. Cada usuário criava um evento a cada 10 segundos, com informações básicas como nome, local, data e descrição.

Durante a execução do teste de carga, monitoramos várias métricas importantes, como tempo de resposta, taxa de erros, utilização de recursos (CPU, memória) e escalabilidade do sistema.

Os resultados obtidos foram muito encorajadores. O software mostrou-se altamente responsivo e estável sob carga. O tempo de resposta médio para as solicitações dos usuários permaneceu abaixo de 1 segundo em todos os cenários de carga. Não foram registrados erros durante o teste.

No entanto, identificamos um gargalo de desempenho quando um grande número de usuários estava navegando simultaneamente pelos detalhes dos eventos. O tempo de resposta aumentou ligeiramente, e a utilização de recursos também apresentou um pequeno aumento.

Para mitigar esse problema, pensamos sobre futuras implementações que podemos realizar no software: otimizações no código do software, como otimização das consultas ao banco de dados e cache dos detalhes dos eventos. Além disso, pretendemos aumentar a capacidade dos servidores web para melhorar a escalabilidade.

No geral, o teste de carga nos permitiu identificar pontos fortes e fracos do software de divulgação de eventos locais. As melhorias que serão implementadas com base nos resultados nos ajudarão a garantir que o sistema seja capaz de lidar com a demanda esperada, proporcionando uma experiência de uso satisfatória para os usuários e promotores de eventos.

17. Requisito Funcional Associado.

a)Qual funcionalidade que ainda não foi desenvolvida
Cadastro.

b)Possui Casos de Teste?
Sim.

Caso de teste: Cadastro bem-sucedido

Descrição: Verificar se é possível cadastrar um novo usuário com informações válidas.

Passos:

- Abrir o aplicativo de cadastro.
- Preencher todos os campos obrigatórios com informações válidas.

- Pressionar o botão de cadastro.
- Verificar se o usuário é redirecionado para a página de sucesso de cadastro.
- Verificar se as informações do usuário são armazenadas corretamente no banco de dados.

Caso de teste: Validação de campos obrigatórios

Descrição: Verificar se o aplicativo valida corretamente os campos obrigatórios.

Passos:

- Abrir o aplicativo de cadastro.
- Deixar em branco um ou mais campos obrigatórios.
- Pressionar o botão de cadastro.
- Verificar se o aplicativo exibe mensagens de erro indicando quais campos devem ser preenchidos.
- Verificar se o usuário não é cadastrado no banco de dados.

Caso de teste: E-mail já cadastrado

Descrição: Verificar se o aplicativo impede o cadastro de um usuário com um e-mail já existente.

Passos:

- Abrir o aplicativo de cadastro.
- Preencher todos os campos obrigatórios com informações válidas.
- Utilizar um e-mail que já esteja cadastrado no banco de dados.
- Pressionar o botão de cadastro.
- Verificar se o aplicativo exibe uma mensagem de erro informando que o e-mail já está em uso.
- Verificar se o usuário não é cadastrado no banco de dados.

c) Durante o desenvolvimento do aplicativo de cadastro em React Native, utilizamos o framework Jest para realizar testes automatizados. Configuramos o ambiente de teste, escrevemos os casos de teste utilizando as funções `describe` e `it`, e executamos os testes utilizando o comando `npm test`. O Jest mostrou-se eficiente na detecção de erros e forneceu resultados claros. A sintaxe clara e concisa do Jest facilitou a escrita e manutenção dos testes. Concluímos que o uso do Jest para testar o código de cadastro em React Native foi positivo, fornecendo confiança na qualidade do código e facilitando a identificação e correção de erros. Recomendamos o Jest para testes em projetos React Native.

d) a. A experiência de utilizar o framework Jest para testar o código de cadastro foi excelente. Os testes automatizados foram eficientes em identificar problemas e erros, resultando em um produto final de alta qualidade. A configuração e execução dos testes foram simplificadas graças à documentação clara e abrangente do Jest. A ferramenta mostrou-se poderosa, proporcionando confiança na estabilidade e funcionalidade do código, além de agilizar o fluxo de trabalho.

b. A utilização do framework Jest para testar o código de cadastro foi altamente produtiva. Os testes automatizados foram escritos de forma simples e rápida, permitindo identificar possíveis problemas e garantir a corretude das informações fornecidas pelos usuários. A automação dos testes economizou tempo e recursos,

eliminando a necessidade de testes manuais repetitivos. Isso resultou em uma maior produtividade no desenvolvimento e entrega de um produto de melhor qualidade

18. Ferramentas Utilizadas

Durante a realização do trabalho, essas foram as ferramentas utilizadas: Selenium, JUnit, Postman, CodeIgniter 4.0, VSCode, Apache JMeter, Jest, Figma, Canvas e MySQL DataBase Server

