Documento de Projeto de Sistema

Eventu

Vitória, ES

2024

Registro de Alterações:

Versão	Responsável	Data	Alterações
1.0	Gabriel Ferrari Wagnitz	21/04/2024	Versão inicial.

1 Introdução

Este documento apresenta o projeto (*design*) do sistema *Eventu*. O projeto consiste no desenvolvimento de uma aplicação web para organizadores de eventos acadêmicos. Eventos acadêmicos possuem necessidades particulares que não são normalmente supridas por sistemas de vendas de ingressos, que geralmente tem foco em eventos culturais e shows. Por isso, entendemos como necessário desenvolver uma plataforma voltada especificamente para o público universitário.

Os principais diferenciais do sistema Eventu são o suporte à multiplas atividades paralelas, em que os participantes dispõem de diversas atividades para escolher e o sistema faz a gestão da grade de programação, e a API para controle de presença, em que a participação numa programação pode ser registrada com auxilio da integração à um sistema de catracas.

Este documento está organizado da seguinte forma: a Seção 2 apresenta a plataforma de software utilizada na implementação do sistema; a Seção 3 apresenta a arquitetura de software; por fim, a Seção 4 apresenta os modelos FrameWeb que descrevem os componentes da arquitetura.

O projeto foi desenvolvido por Thaliys Antunes Daré e Gabriel Ferrari Wagnitz.

2 Plataforma de Desenvolvimento

Para o projeto serão utilizadas as tecnologias e bibliotecas recomendadas pelo Spring Framework em seu projeto *Spring Initializr*[1]. O *Spring Initializr* é uma ferramenta do Spring para facilitar a configuração inicial do projeto, facilitando a inclusão de bibliotecas para diversos propósitos.

Na Tabela 1 são listadas as tecnologias utilizadas no desenvolvimento da ferramenta, bem como o propósito de sua utilização.

[1] https://start.spring.io/

Tabela 1 – Plataforma de Desenvolvimento e Tecnologias Utilizadas.

Tecnologia	**Versão**	**Descrição**	**Propósito**
Spring Framework	3.2.4	Framework de inversão de controle e injeção de dependência. Possui um conjunto de APIs para auxiliar em diversas etapas do desenvolvimento de grandes sistemas web	Redução da complexidade do desenvolvimento, implantação e gerenciamento de aplicações Web a partir de seus componentes de infra-estrutura prontos para o uso.
Java	21	Linguagem de programação orientada a objetos e independente de plataforma.	Escrita do código-fonte das classes que compõem o sistema.
Spring Data JPA	3.2.4	API para persistência de dados por meio de mapeamento objeto/-relacional.	Persistência dos objetos de domínio sem necessidade de escrita dos comandos SQL.
Thymeleaf	3.1.2	Ferramenta para criação de (*templates*) XML/XHTML/HTML5 integrada ao Spring	Reutilização da estrutura visual comum às paginas, facilitando a manutenção do padrão visual do sistema.
TailwindCSS	3.4.3	Framework de classes CSS processadas em tempo de compilação	Facilitar a estilização de páginas web responsivas
MySQL Server	8.3	Sistema Gerenciador de Banco de Dados Relacional gratuito.	Armazenamento dos dados manipulados pela ferramenta.
Tomcat	10.1.19	Servidor de Aplicações Imbutido no Spring Boot.	Fornecimento de implementação das APIs citadas acima e hospedagem da aplicação Web, dando acesso aos usuários via HTTP.

Na Tabela 2 vemos os softwares que apoiaram o desenvolvimento de documentos e também do código fonte.

Tabela 2 – Softwares de Apoio ao Desenvolvimento do Projeto

Tecnologia	**Versão**	**Descrição**	**Propósito**
FrameWeb Plugin	1.0	Plugin do VisualParadigm para o método FrameWeb.	Criação dos modelos de Entidades, Aplicação, Persistência e Navegação.
VisualParadigm	17.1	Ferramenta de modelagem UML	Criação de Diagramas de Classe
Overleaf	-	Editor Online do LaTeX	Documentação do projeto arquitetural do sistema.
Apache Maven	3.6	Ferramenta de gerência/construção de projetos de software.	Obtenção e integração das dependências do projeto.
Spring Initializr	-	Ferramenta de geração inicial de projeto Spring.	Automação de configuração

3 Arquitetura de Software

A Figura 1 mostra a arquitetura do sistema *Eventu*.

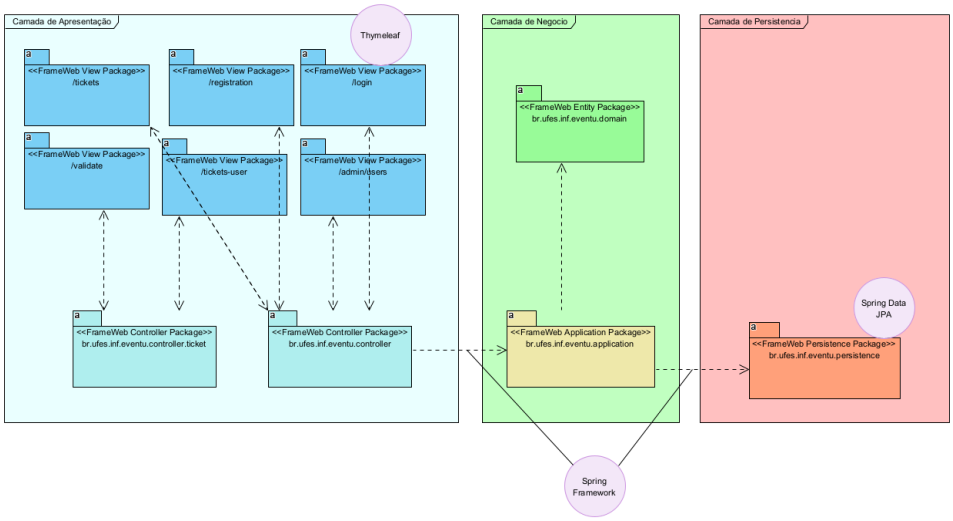

Figura 1 – Arquitetura de Software.

O sistema *Eventu* adota a arquitetura Model-View-Controller (MVC)(FOWLER, 2002) para prover organização, modularidade e separação de conceitos (do inglês *separation of concerns*, SoC). Além disso, a escolha da arquitetura MVC torna-se prática, dado o grande suporte que o framework Spring provê à esta arquitetura.

O sistema *Eventu* implementa cada um dos módulos MVC da seguinte forma:

Modelo: No núcleo do sistema, o Modelo encapsula os dados e regras de negócio, gerenciando o acesso e a manipulação de informações sobre eventos, palestrantes, participantes e inscrições. Nele estão os pacotes do modelo de aplicação (ex: ManageAttractionServiceImpl, AuthenticateUserServiceImpl), classes de dominio (ex: Ticket, User, Attraction) e de persistencia (ex: TicketJPADAO, UserJPADAO)

Visão: Representado pelas visualizações geradas por templates Thymeleaf e Tailwind CSS, a camada de Visão apresenta os dados do Modelo para os usuários, permitindo visualização, interação e manipulação de informações.

Controlador: O Controlador atua como intermediário, recebendo requisições do usuário, fazendo validação dos dados de entrada e direcionando-as para o Modelo e recuperando os dados processados para apresentá-los na Visão. As classes TicketController e UserController fazem parte da camada Controlador.

4 Modelagem FrameWeb

Eventu é um sistema Web cuja arquitetura utiliza *frameworks* comuns no desenvolvimento para esta plataforma. Desta forma, o sistema pode ser modelado utilizando a abordagem FrameWeb (SOUZA, 2020).

A Tabela 3 indica os *frameworks* presentes na arquitetura do sistema que se encaixam em cada uma das categorias de *frameworks* que FrameWeb dá suporte. Em seguida, os modelos FrameWeb são apresentados para cada camada da arquitetura.

Tabela 3 – *Frameworks* da arquitetura do sistema separados por categoria.

Categoria de *Framework*	***Framework* Utilizado**
Controlador Frontal	Spring Framework
Injeção de Dependências	Spring Framework
Mapeamento Objeto/Relacional	Spring Data JPA
Segurança	Spring Security

4.1 Camada de Negócio

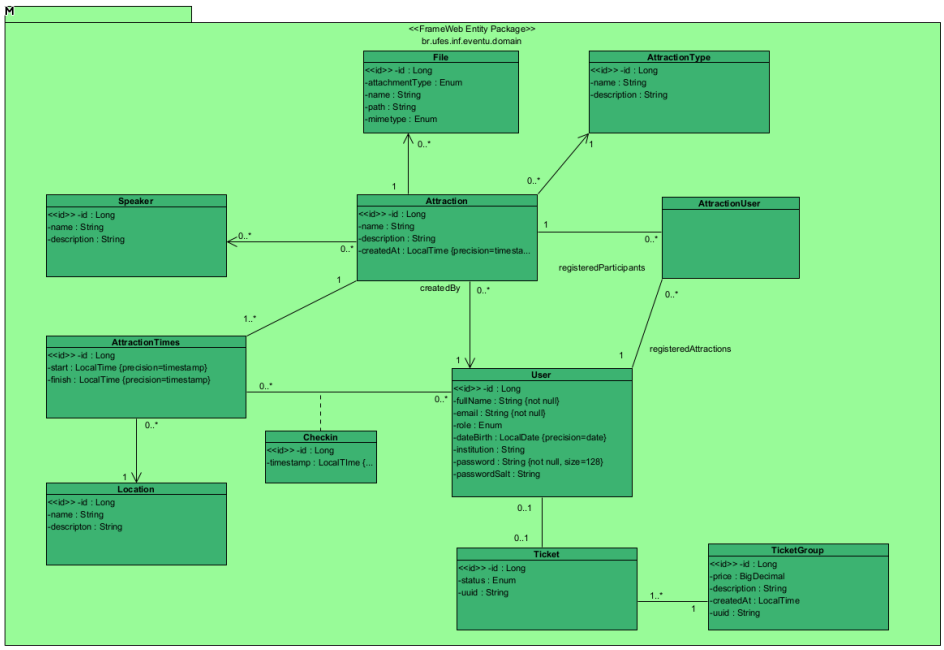

Figura 2 – Modelo de Entidades.

Attraction: Representa uma atração em um evento, como palestra, workshop ou visita técnica.

Speaker: Representa um palestrante ou outro profissional que participará de uma atração.

File: Representa um arquivo armazenado no sistema, como material de apresentação ou foto.

AttractionType: Representa o tipo de atração, como palestra, workshop ou visita técnica.

User: Representa um usuário do sistema, como participante, organizador ou staff.

Ticket: Representa um ingresso para um evento, com informações como tipo de ingresso, valor e local.

TicketGroup: Representa um grupo de ingressos, permitindo a emissão de vários ingressos de uma vez com um identificador comum.

CheckIn: Classe de relacionamento entre o horario da atração e um usuário, utilizada para registro de presença.

CheckIn: Classe de relacionamento entre o horario da atração e um usuário, utilizada para registro de presença.

AttractionUser: Representa a *Join Table* dos usuarios inscritos numa atividade, na relação muitos para muitos.

4.2 Camada de Acesso a Dados

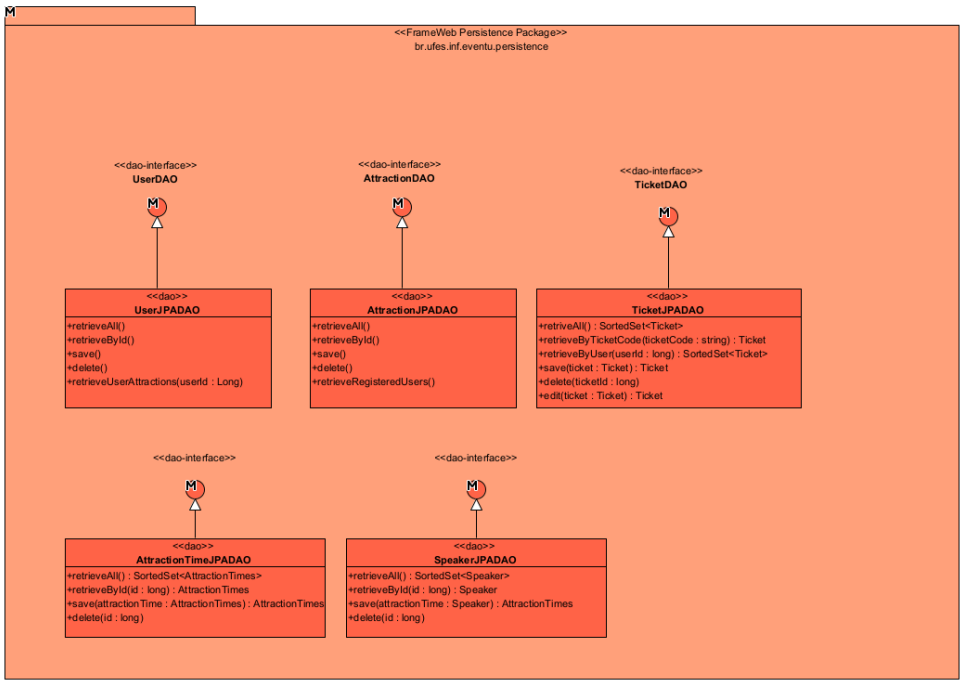

Figura 3 – Modelo de Persistencia.

Além da implementação de funcionalidades CRUD (do inglês: Criação, Consulta, Atualização e Destruição), podemos ressaltar na camada de acesso a dados algumas outras

operações:

TicketJPADAO: +retireveByTicketCode - Operação responsável por encontrar o ticket no banco de dados durante o processo de validação do ingresso.

AttractionJPADAO: +retireveRegisteredUsers() - Operação responsável por retornar a lista de usuários inscritos em uma dada atração.

4.3 Camada de Apresentação

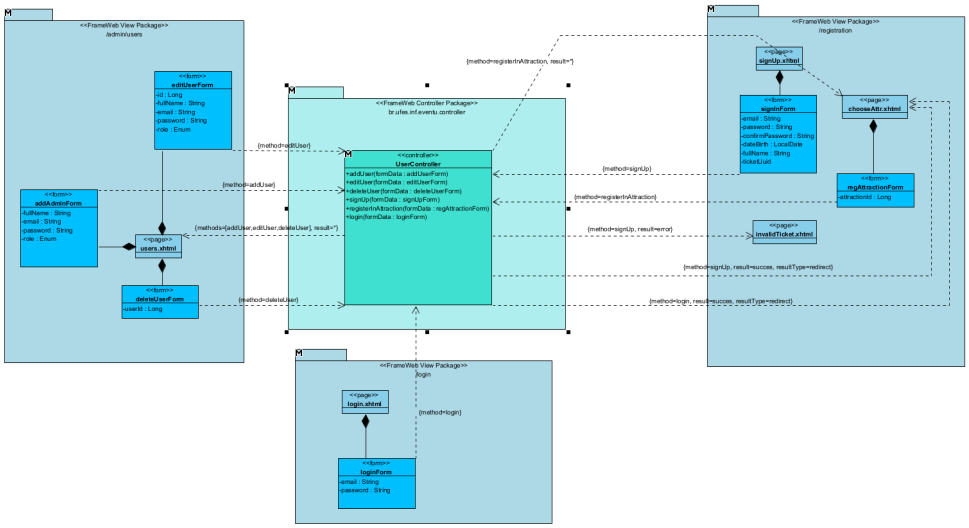

Figura 4 – Modelo de Navegação - Cadastro de Usuários.

A camada de apresentação é responsável pelas diversas telas do sistema. na Figura 4 vemos o 3 fluxos de navegação envolvendo usuários.

Em **/admin/users** vemos o processo de gestão dos usuários por administradores.

Em **/registration** vemos o cadastro de usuários, criação de login e senha, além da validação do ticket que é comprado fora do sistema. O sistema apenas faz validação do código e a associação ao usuário. Após isso, o usuário e apresentado à tela **chooseAttr.xhtml** em que ele escolhe progressivamente as atrações que vai participar.

Em **/registration** vemos o cadastro de usuários, criação de login e senha, além da validação do ticket que é comprado fora do sistema. O sistema apenas faz validação do código e a associação ao usuário. Após isso, o usuário e apresentado à tela **chooseAttr.xhtml** em que ele escolhe progressivamente as atrações que vai participar.

Em **/login** vemos o fluxo de autenticação do usuário.

Na Figura 5 vemos mais 3 fluxos de navegação, agora envolvendo envolvendo o cadastro de atrações.

O processo se assemelha a um simples CRUD, assim como na gestão de usuários por administradores. O ponto a ressaltar é que as atrações estão relacionadas à entidade

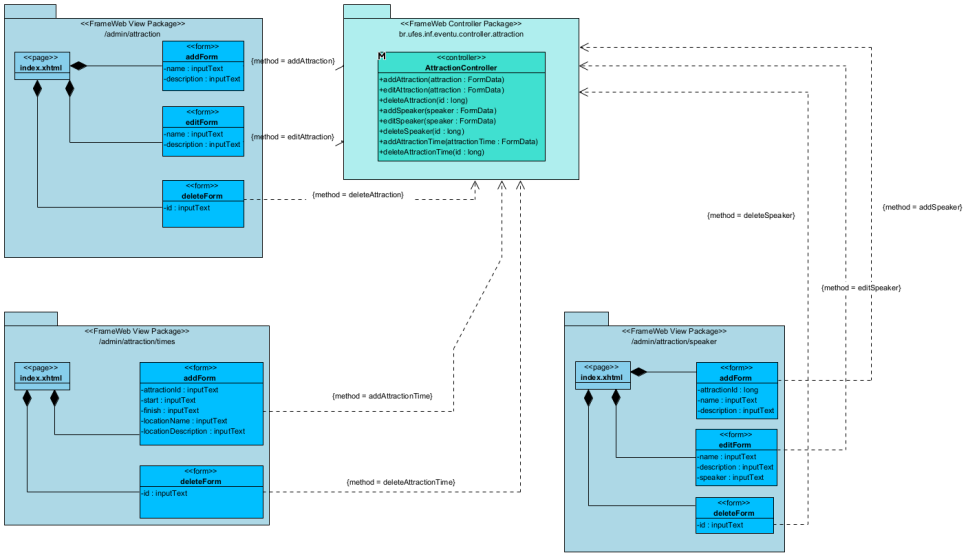

Figura 5 – Modelo de Navegação - Cadastro de Atrações.

palestrante e as instâncias de horário. Instancias de horário foram modeladas dessa forma para acomodar atrações que se estendam por mais de um dia, em locais diferentes. Para isso temos a classe *AttractionTimes*

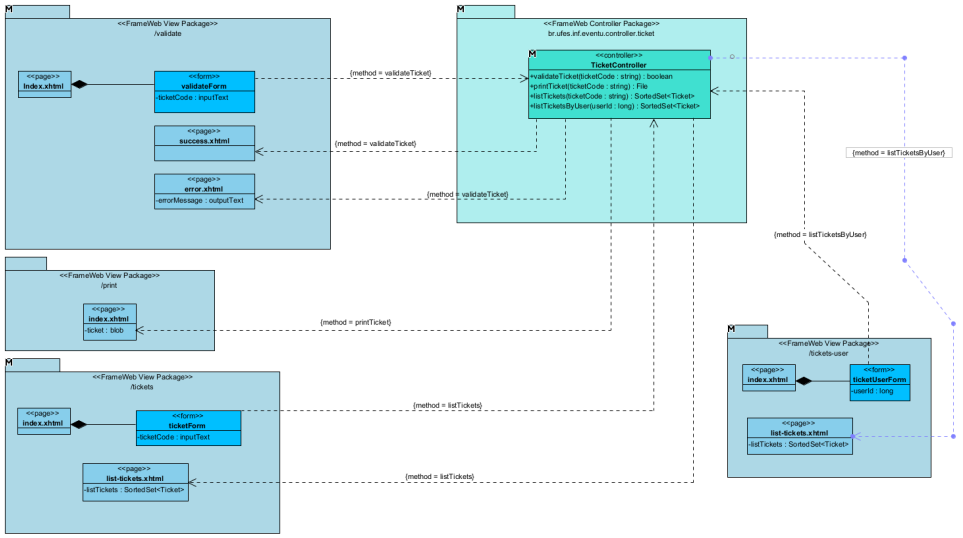

Figura 6 – Modelo de Navegação - Cadastro de Tickets.

Na Figura 6 vemos mais 4 fluxos de navegação, desta vez envolvendo envolvendo os ingressos (*Tickets*).

O ponto interessante de se ressaltar é a operação **+printTicket()** que permite um administrador imprimir um ingresso, facilitando a emissão para venda física.

Referências

FOWLER, M. *Patterns of Enterprise Application Architecture*. 1. ed. [S.l.]: Addison-Wesley, 2002. ISBN 9780321127426. Citado na página 4.

SOUZA, V. E. S. The FrameWeb Approach to Web Engineering: Past, Present and Future. In: ALMEIDA, J. P. A.; GUIZZARDI, G. (Ed.). *Engineering Ontologies and Ontologies for Engineering*. 1. ed. Vitória, ES, Brazil: NEMO, 2020. cap. 8, p. 100–124. ISBN 9781393963035. Disponível em: <http://purl.org/nemo/celebratingfalbo>. Citado na página 5.