



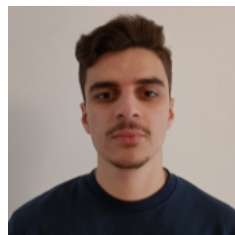
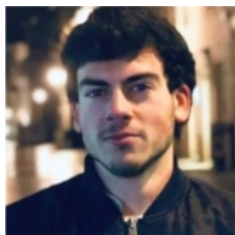
Universidade do Minho

Mestrado Engenharia Informática
Arquiteturas Aplicacionais e Sistemas Interativos Confiáveis

SISTEMA DE GESTÃO DE SALAS

Grupo 7

PG50384 Francisco Reis Izquierdo,
PG50528 José Pedro Martins Magalhães,
PG50779 Tiago Fernandes Ribeiro



9 de abril de 2023

Conteúdo

1	Introdução	5
2	Âmbito do Projeto	6
2.1	Contextualização	6
2.2	Definição do Produto	6
2.3	Motivação e Objetivos	7
3	Convenções de nomenclatura e definições	8
3.1	Modelo de Domínio	8
3.2	Definição de Entidades	8
3.2.1	Administrador	8
3.2.2	Utilizador	9
3.2.3	Sala	9
3.2.4	Evento	9
4	Âmbito do Sistema	10
4.1	Diagrama de <i>Use Cases</i>	10
4.2	Atores	10
4.2.1	Administrador	11
4.2.2	Utilizador	11
4.2.3	Aluno	11
4.2.4	Docente	11
4.3	Mockups	12
4.3.1	Ferramenta de <i>back-office</i>	12
4.3.2	Aplicação nos ecrãs perto das salas	17
5	Requisitos Funcionais	19
6	Requisitos Não Funcionais	20
6.1	Aparência	20
6.2	Usabilidade	20
6.3	Performance	20
6.4	Operacional	20
6.5	Manutenção e suporte	20
6.6	Segurança	20
6.7	Cultural e político	20
6.8	Legal	20
7	Contexto e Alcance	21
7.1	Perspetiva de Domínio	21
7.1.1	Cliente	21
7.1.2	Sistema de Gestão de Salas	21

7.2	Perspetiva Técnica	21
7.2.1	Clientes	21
7.2.2	Sistema de Gestão de Salas	22
8	Estratégia da Solução	24
8.1	Ferramentas utilizadas	24
8.2	Conceção dos modelos e geração de código	24
8.3	Implementação da lógica de negócios	25
8.4	Criação do servidor aplicacional	25
8.5	Implementação da interface gráfica	26
8.6	Implementação dos <i>beans</i>	26
9	Visualização em tempo de execução	27
10	Avaliação crítica	28
10.1	Implementação do servidor aplicacional	28
10.2	Padrões de design na interface	28
10.3	Avaliação heurística	29
11	Conclusões e Trabalho Futuro	30

Lista de Figuras

3.1	Modelo de domínio	8
4.1	Diagrama de <i>use cases</i>	10
4.2	Diagrama de <i>estados</i> da ferramenta de <i>back-office</i>	11
4.3	Mockup de início de sessão	12
4.4	Mockup de registo	13
4.5	Mockup da página de gestão de aulas/salas/cursos/disciplinas/docentes	13
4.6	Mockup da página de criação de aulas/salas/cursos/disciplinas/docentes	14
4.7	Mockup da página de edição de aulas/salas/cursos/disciplinas/docentes	14
4.8	Mockup da página de eliminação de aulas/salas/cursos/disciplinas/docentes	15
4.9	Mockup da página de edição de dados da conta	15
4.10	Mockup da página de eliminação da conta	16
4.11	Mockup da página de consulta de presenças	16
4.12	Mockup da página de inscrição numa disciplina	17
4.13	Mockup da página inicial	17
4.14	Mockup de entrada permitida	18
4.15	Mockup de entrada não permitida	18
7.1	Arquitetura do sistema	23
8.1	<i>Platform Specific Model</i> (PSM)	25
9.1	Diagrama de sequência de resposta a um pedido do cliente	27
10.1	Padrão de design <i>breadcrumbs</i>	28
10.2	Padrão de design <i>calendar picker</i>	28
10.3	Padrão de design <i>alternating row colors</i>	29

Lista de Tabelas

10.1 Avaliação heurística da interface do sistema	29
-------------------------------------------------------------	----

1. Introdução

O presente relatório surge no âmbito do trabalho prático das unidades curriculares de Arquiteturas Aplicacionais e de Sistemas Interativos Confiáveis do Mestrado em Engenharia Informática da Universidade do Minho.

Este projeto aborda o desenvolvimento de uma aplicação denominada Sistema de Gestão de Salas (SGS). Esta é uma solução tecnológica desenvolvida para otimizar a organização e utilização eficiente das salas em instituições de ensino. O sistema é projetado para atender às necessidades de diversas instituições de ensino, que possuem múltiplas salas onde ocorrem aulas, conferências e outros eventos. Através da instalação de telas eletrónicas próximas às portas de cada sala, o sistema fornece informações em tempo real sobre a disponibilidade da sala e os eventos que estão a ocorrer ou estão programados para acontecer.

2. Âmbito do Projeto

2.1 Contextualização

Este projeto surge no contexto da crescente necessidade das instituições de ensino de gerirem de forma mais eficiente e segura a ocupação das suas salas e a realização de eventos nos seus espaços. Com o aumento do número de alunos, professores e funcionários, assim como a diversificação das atividades realizadas nos seus espaços, as instituições de ensino têm enfrentado desafios cada vez maiores na gestão de suas infraestruturas.

Por um lado, a gestão de salas e eventos em instituições de ensino pode ser bastante complexa, com diversas atividades a acontecer simultaneamente em diferentes espaços, muitas vezes com necessidades específicas de equipamentos e recursos. Por outro lado, a segurança e o controlo de acesso às salas são fundamentais para garantir a integridade dos alunos, professores e funcionários, bem como para evitar situações de conflito ou violação de normas da instituição.

Assim, o desenvolvimento de um sistema de gestão de sinalização eletrónica, que permita a visualização em tempo real da ocupação das salas e a gestão de acesso aos espaços de forma automatizada, é uma solução importante para as instituições de ensino que procuram aprimorar a sua gestão de infraestrutura e garantir a segurança de seus utilizadores.

2.2 Definição do Produto

Com o objetivo de aprimorar a gestão de sinalização eletrónica em instituições de ensino, pretende-se desenvolver um sistema que forneça serviços especializados para essa finalidade. Assim, pretende-se desenvolver um sistema para fornecer serviços de gestão de sinalização eletrónicos para instituições de ensino. Estas instituições são caracterizadas por terem diversas salas, nas quais podem estar a decorrer aulas (ou outros eventos). Junto à porta de cada sala, pretende-se instalar um ecrã que informa sobre a disponibilidade da sala e sobre o evento que lá decorre. A gestão da ocupação das salas é feita através de uma ferramenta de back-office onde se calendarizam eventos (associados às salas do espaço). O ecrã principal da ferramenta apresenta um calendário e permite acrescentar eventos a uma determinada hora, associados a uma determinada sala. A ferramenta permite também criar/editar salas, para além de poder consultar a agenda de ocupação de uma determinada sala. Para além da ferramenta de back-office, pretende-se que exista uma aplicação de visualização de agenda, a ser exibida nos ecrãs ao lado das salas que, recebendo como parâmetro o identificador da sala, apresenta (pelo menos) informação sobre o evento que aí decorre ou, se a sala estiver livre, sobre o próximo evento a decorrer ali. Pretende-se ainda que o sistema faça gestão de acessos. Para tal o sistema deverá estar preparado para receber informação de leitores de cartões instalados nas salas. O tipo de gestão de acesso, depende do tipo de evento a decorrer. No caso das aulas, existe listas de alunos e docentes para cada turno e o sistema deverá registar a sua presença de aula. Se o evento for uma sessão de uma conferência, existe uma lista de inscritos na conferência e o sistema deverá assinalar se a entrada é permitida ou não. Se o evento é de entrada livre o sistema deverá apenas assinalar se a sala ainda tem capacidade para aceitar o participante.

2.3 Motivação e Objetivos

A motivação deste projeto é aprimorar a gestão de salas em instituições de ensino. Muitas vezes, esta gestão pode ser caótica, com salas sendo reservadas erroneamente ou sem qualquer informação sobre a sua ocupação, o que pode levar a aulas e eventos não acontecerem devido a conflitos de agendamento ou dificuldades na gestão de acesso.

Com este sistema de gestão de salas, o objetivo é fornecer uma ferramenta que possa auxiliar as instituições de ensino a gerir a ocupação das salas de forma mais eficiente, garantindo que as informações sobre os eventos em curso estejam sempre atualizadas e acessíveis a todos os interessados.

Além disso, o sistema tem como objetivo permitir o controlo de acesso às salas, de forma a garantir a segurança e o cumprimento das normas da instituição, seja através da utilização de cartões de acesso, da verificação de listas de participantes ou outras formas de identificação.

3. Convenções de nomenclatura e definições

3.1 Modelo de Domínio

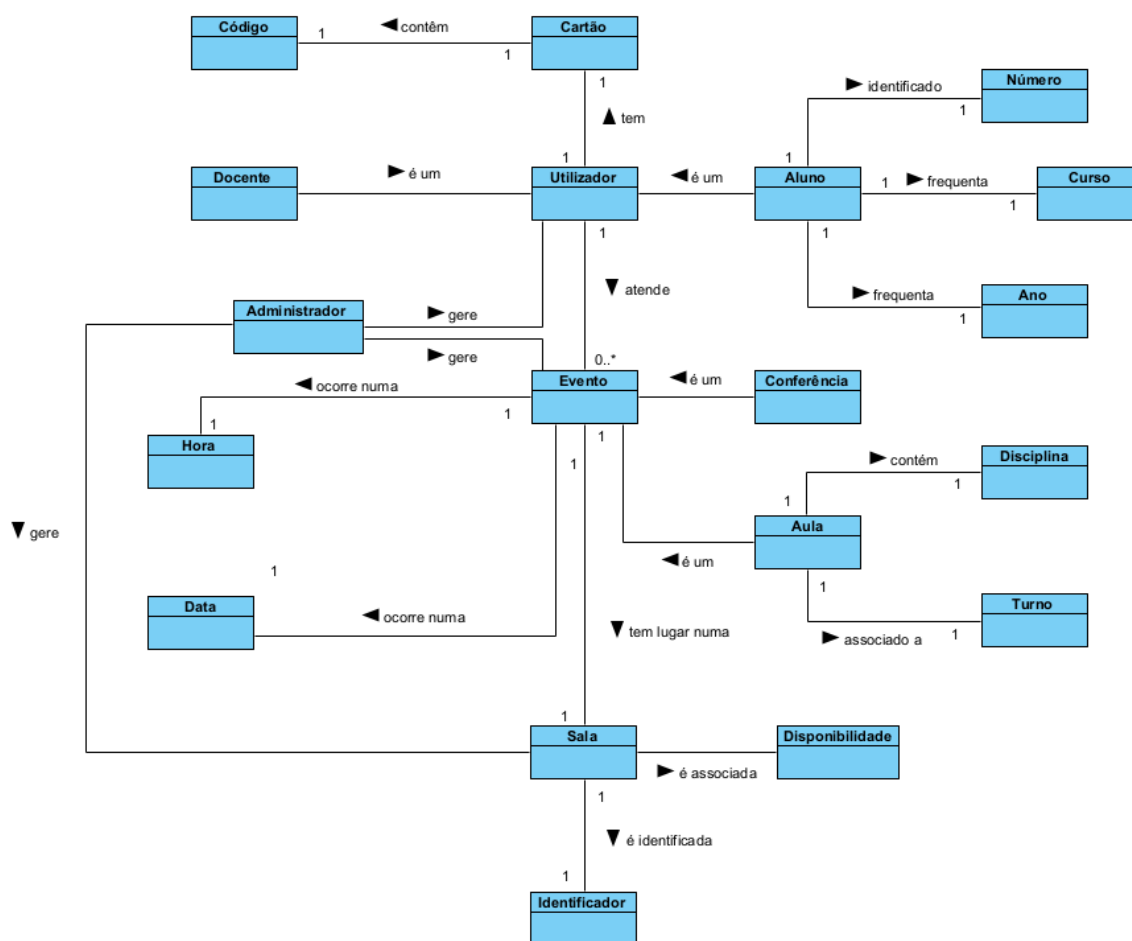


Figura 3.1: Modelo de domínio

3.2 Definição de Entidades

Nesta secção, são abordadas as diferentes entidades presentes no Modelo de Domínio apresentado na secção anterior (figura 3.1).

3.2.1 Administrador

O administrador representa um utilizador com privilégios elevados, responsável pela gestão do sistema como um todo. Ele pode criar e editar salas, eventos e utilizadores, além de gerir a

ocupação das salas, definir permissões de acesso e outras configurações importantes do sistema.

3.2.2 Utilizador

O utilizador representa um cliente comum que pode visualizar a agenda de eventos e reservar salas, desde que tenha permissão para tal. Dependendo do tipo de evento, ele pode ser obrigado a usar um cartão de acesso ou identificar-se de outra forma para aceder ao mesmo.

3.2.3 Sala

Uma sala representa um espaço onde decorre um evento. Cada sala tem um identificador único, além de uma descrição e outras informações relevantes, como capacidade, equipamentos disponíveis, etc.

3.2.4 Evento

Um evento representa um acontecimento que ocorre numa sala, como uma aula ou uma conferência. Cada evento tem um identificador único, uma descrição e outras informações relevantes, como data, hora de início e término, participantes, etc. Dependendo do tipo de evento, pode ser necessário existir um controlo de acessos.

4. Âmbito do Sistema

4.1 Diagrama de *Use Cases*

De maneira a compreender melhor o contexto do sistema, é apresentado um diagrama de *Use Cases*. Neste vão ser explicitadas algumas das principais funcionalidades do sistema, bem como os atores presentes no mesmo. Com este diagrama, é ainda possível identificar, quais as funcionalidades a que cada tipo de ator do sistema vai ter acesso.

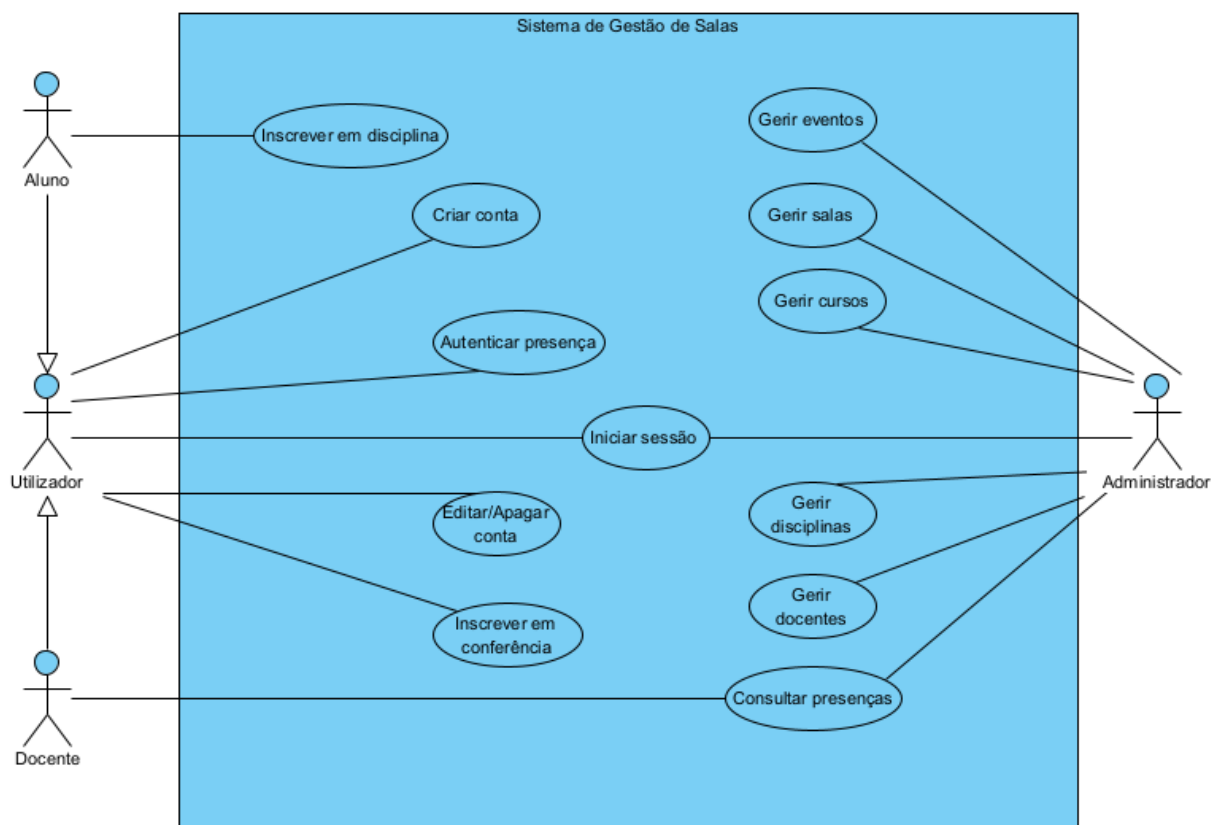


Figura 4.1: Diagrama de *use cases*

4.2 Atores

Como representado no diagrama anterior, o sistema suporta 4 tipos de atores: Administrador, Docente, Aluno e Docente.

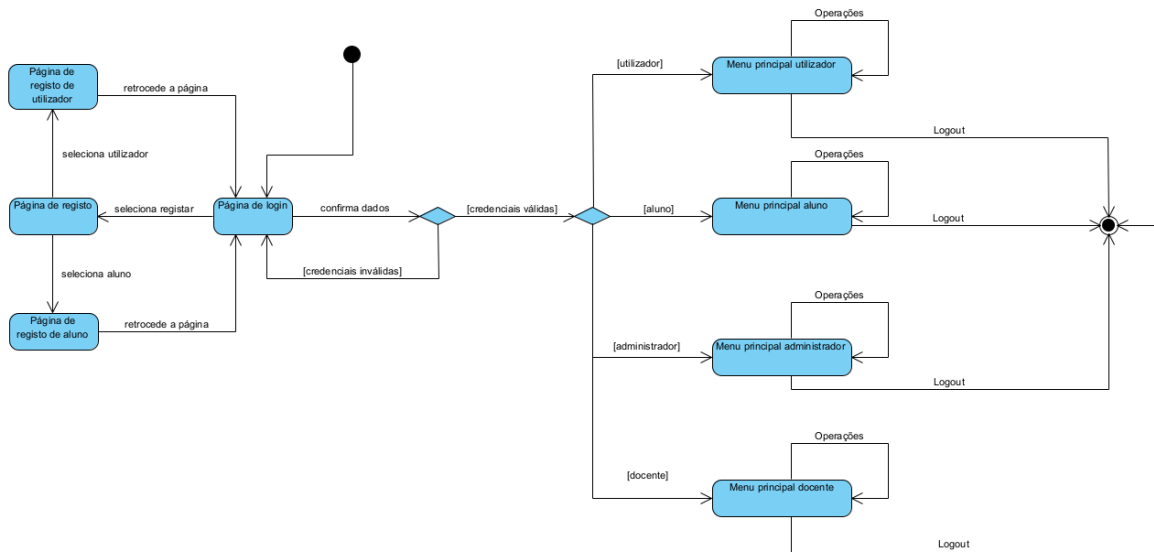


Figura 4.2: Diagrama de *estados* da ferramenta de *back-office*

4.2.1 Administrador

O Administrador desempenha um papel fundamental na administração e configuração do sistema, sendo que possui privilégios avançados, permitindo-lhe criar, editar e eliminar eventos, salas, disciplinas, docente e cursos. Estas funcionalidades permitem ao Administrador gerir de forma eficiente a agenda e a disponibilidade das salas. Além disso, o Administrador tem a capacidade de consultar a ocupação das salas e verificar as presenças registadas durante os eventos.

4.2.2 Utilizador

O Utilizador interage com a aplicação de visualização de agenda e com a aplicação de *back-office*, sendo que usa as funcionalidades disponíveis para a sua conveniência. O Utilizador pode criar, editar e eliminar o seu próprio perfil de utilizador no sistema. Este também pode efetuar a inscrição em conferências, sendo este o único tipo de evento a que o mesmo pode assistir. Para acessar às salas, o Utilizador precisa de autenticar o seu acesso usando um cartão ou outro método apropriado de identificação.

4.2.3 Aluno

O Aluno é uma abstração do Utilizador, sendo capaz de assistir a aulas. Este tem todas as funcionalidades do Utilizador, sendo que também pode inscrever-se em disciplinas, onde depois pode assistir às aulas das mesmas.

4.2.4 Docente

O Docente também é uma abstração do Utilizador com privilégios adicionais, porém ao contrário deste, o Docente não é capaz de criar a sua conta sendo este procedimento efetuado pelo Administrador. Além das funcionalidades disponíveis para o Utilizador, o Docente também pode consultar as presenças registadas durante os eventos lecionados por ele. Esta funcionalidade adicional é especialmente útil para professores que desejam acompanhar a participação dos alunos nas suas aulas.

4.3 Mockups

Os mockups são representações visuais que permitem visualizar e testar a interface da aplicação antes mesmo desta ser implementada completamente. Estes desempenham um papel crucial no processo de design, permitindo explorar diferentes ideias e refinar as funcionalidades da aplicação antes de investir tempo e recursos na implementação da mesma. Para a criação destes mockups foi utilizada a ferramenta *Pencil*, especializada para a criação de protótipos de interfaces.

4.3.1 Ferramenta de *back-office*

4.3.1.1 Iniciar sessão



The mockup displays a login interface for a 'Sistema de gestão de salas'. It features a title, two input fields for email and password, two buttons for login and registration, and two links for user and student registration.

Sistema de gestão de salas

E-mail

E-mail

Palavra-passe

Iniciar sessão Registrar

[Registar como Utilizador](#)
[Registar como Aluno](#)

Figura 4.3: Mockup de início de sessão

4.3.1.2 Criar conta

Sistema de gestão de salas

E-mail

E-mail

Nome

Nome

Idade

Idade

Ano

Ano

Cartão

Cartão

Curso

Curso

Palavra-passe

Criar conta

Cancelar

Figura 4.4: Mockup de registo

4.3.1.3 Gerir eventos/salas/cursos/disciplinas/docentes

Sistema de gestão de salas

Página Inicial > Gerir aulas

Nome	Disciplina	Turno	Sala	Hora Início	Hora Fim	Data	Docente			
Aula 7 - TP	Arquiteturas Aplicacionais	1	cp2-1.07	16:00h	19:00h	15/03/2023	Jane Doe	Presenças		
Aula 5 - PL	Sistemas Interativos Confiáveis	1	cp2-1.07	13:00h	15:00h	16/03/2023	Johny Doe	Presenças		

Criar aula

Voltar Atrás

Figura 4.5: Mockup da página de gestão de aulas/salas/cursos/disciplinas/docentes

Sistema de gestão de salas

Página Inicial > Gerir aulas > Criar aula

Nome

Hora início

Hora fim

Hora

▼

Hora

▼

Data

Data

▼

Disciplina

Disciplina

▼

Turno

Criar aula

Cancelar

Figura 4.6: Mockup da página de criação de aulas/salas/cursos/disciplinas/docentes

Sistema de gestão de salas

Página Inicial > Gerir aulas

Nome	Disciplina	Turno	Sala	Hora Início	Hora Fim	Data	Docente			
Aula 7 - TP	Arquiteturas Aplicacionais	1	cp2-1.07	16:00	19:00	15/03/2023	Jane Doe	Presenças		
Aula 5 - PL	Sistemas Interativos Confiáveis	1	cp2-1.07	13:00h	15:00h	16/03/2023	Johny Doe	Presenças		

Criar aula

Voltar Atrás

Figura 4.7: Mockup da página de edição de aulas/salas/cursos/disciplinas/docentes

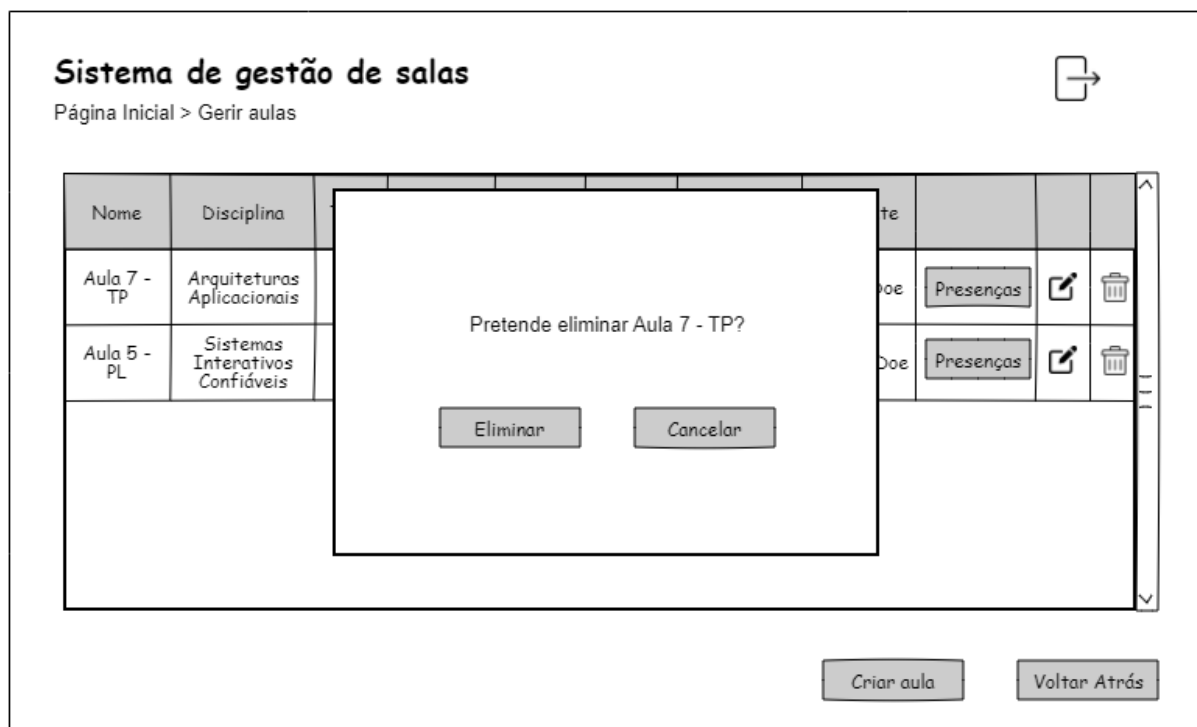


Figura 4.8: Mockup da página de eliminação de aulas/salas/cursos/disciplinas/docentes


4.3.1.4 Editar dados

The mockup shows the 'Editar dados' page in the 'Sistema de gestão de salas'. It has a breadcrumb 'Página Inicial > Editar dados' and a share icon. The form contains fields for 'Nome' (John Doe), 'Idade' (20), 'Ano' (2), 'Cartão' (XXXXXX), and 'Palavra-passe' (masked with asterisks). At the bottom, there are 'Salvar' and 'Cancelar' buttons.

Figura 4.9: Mockup da página de edição de dados da conta

4.3.1.5 Eliminar conta

Sistema de gestão de salas
Página Inicial



Pretende eliminar a sua conta?
Insira a sua palavra-passe para eliminar a conta.


Eliminar conta

Cancelar

Figura 4.10: Mockup da página de eliminação da conta

4.3.1.6 Consulta de presenças

Sistema de gestão de salas
Página Inicial > Gerir Aulas > Presenças Aula 7 - TP



Nome	Email	Presente
João	PGXXXXX@alunos.uminho.pt	Sim
Francisco	PGYYYYY@alunos.uminho.pt	Sim
Tiago	PGZZZZZ@alunos.uminho.pt	Não


Voltar Atrás

Figura 4.11: Mockup da página de consulta de presenças

4.3.1.7 Inscrever em disciplina/conferência

Sistema de gestão de salas

Página Inicial > Inscrever em disciplina



Disciplina	Curso	Docente	Carga Horária	Descrição	Inscrito
Sistemas Interativos Confiáveis	MEI	Jonh Doe	5	Disciplina que trata ...	<input type="checkbox"/>
Arquitetura Aplicacionais	MEI	Jonhy Doe	7	Disciplina focada ...	<input checked="" type="checkbox"/>

Voltar Atrás

Figura 4.12: Mockup da página de inscrição numa disciplina

4.3.2 Aplicação nos ecrãs perto das salas

4.3.2.1 Autenticar acesso

Sala cp2-1.07

Complexo Pedagógico 2, Piso 1, Sala 7

Lotação: 07/35

Evento: Aula

Nome: Aula 7 - TP Disciplina: Arquiteturas Aplicacionais Hora Início: 16:00h

Docente: Jane Doe Turno: 1 Data: 15/03/2023 Hora Fim: 19:00h

Inserir código

Marcar presença

Figura 4.13: Mockup da página inicial

Sala cp2-1.07
Complexo Pedagógico 2, Piso 1, Sala 7
Lotação: 07/35

Evento: Aula
Nome: Aula 7 - TP
Docente: Jane Doe

Entrada permitida. A sua presença foi marcada

Hora Início: 16:00h
Hora Fim: 19:00h

Inserir código

Figura 4.14: Mockup de entrada permitida

Sala cp2-1.07
Complexo Pedagógico 2, Piso 1, Sala 7
Lotação: 07/35

Evento: Aula
Nome: Aula 7 - TP
Docente: Jane Doe

Entrada não permitida.
Você não pode frequentar este evento

Hora Início: 16:00h
Hora Fim: 19:00h

Inserir código

Figura 4.15: Mockup de entrada não permitida

5. Requisitos Funcionais

Requisitos relativos à aplicação de back-office:

- Os eventos estão associados às salas do espaço;
- Permite agendar eventos a uma determinada hora;
- Permite criar, editar e eliminar salas, eventos, cursos, disciplinas e docentes;
- Permite registar alunos e utilizadores;
- Permite editar e eliminar alunos e utilizadores;
- Permite que alunos, utilizadores e docentes se inscrevam em conferências;
- Permite que os alunos se inscrevam em disciplinas;
- Permite que os alunos sejam inscritos em aulas;

Requisitos relativos à aplicação de visualização:

- O sistema recebe informação de acesso relativa ao utilizador;
- Permite visualizar a agenda de uma sala dado o identificador da sala;
- Em caso de aulas o sistema deve registar a presença de alunos e professores nas mesmas;
- No caso de uma conferência, existe uma lista de inscritos e o sistema deve assinalar se a entrada é permitida ou não a quem pretenda assistir;
- Se o evento tiver entrada livre, o sistema deve permitir a entrada até esgotar a capacidade.
- Permite consultar agenda de ocupação de uma sala.

6. Requisitos Não Funcionais

6.1 Aparência

- O sistema deverá ser simples e atrativo para os utilizadores.

6.2 Usabilidade

- Um utilizador com 10 minutos de experiência com a aplicação deverá conseguir perceber as várias funcionalidades.

6.3 Performance

- O sistema deverá ser escalável;
- O sistema deve ser rápido e responsivo.

6.4 Operacional

- O sistema deverá estar preparado para receber as informações dos leitores de cartões presentes nas salas.

6.5 Manutenção e suporte

- O sistema deverá estar preparado para ser traduzido em qualquer linguagem;
- O sistema deve permitir a programadores uma fácil compreensão do código desenvolvido.

6.6 Segurança

- O sistema deve permitir a confidencialidade e segurança dos dados dos utilizadores;
- O sistema rejeita a inserção de dados errados.

6.7 Cultural e político

- O sistema deve incluir a língua falada pelos criadores.

6.8 Legal

- O sistema deve respeitar a legislação portuguesa acerca da proteção de dados.

7. Contexto e Alcance

Uma vez especificados os objetivos principais da aplicação, convém mencionar o contexto e o alcance que é pretendido obter. Deste modo, irão ser abordados aspetos relativos aos utilizadores da aplicação e os sistemas envolvidos na mesma.

7.1 Perspetiva de Domínio

7.1.1 Cliente

O cliente é o interveniente responsável pela interação direta com a aplicação. Este interage com o pretexto de obter uma resposta da aplicação que satisfaça o seu pedido, no qual o mesmo pode ser em prol de um dado contexto. Acima de tudo, um utilizador ao interagir com a aplicação, pretende conhecer ou aceder e usufruir do serviço disponível.

7.1.2 Sistema de Gestão de Salas

Sistema principal que é responsável por receber e responder aos pedidos provenientes de um qualquer utilizador. O sistema oferece uma variedade de serviços relacionados à gestão de salas, permitindo aos utilizadores interagir com o sistema de forma eficiente e conveniente.

7.2 Perspetiva Técnica

7.2.1 Clientes

Podemos destacar 4 tipos de clientes que interagem com o sistema:

- **Administrador:** O Administrador é responsável pela administração e configuração do sistema. Possui privilégios avançados, permitindo a criação, edição e eliminação de eventos, salas, disciplinas, docentes e cursos. Pode consultar a ocupação das salas e verificar as presenças registadas durante os eventos. Utiliza a interface de *back-office* para gerir eficientemente a agenda e a disponibilidade das salas.
- **Utilizador:** O Utilizador é um utilizador regular do sistema, que interage com a aplicação de visualização de agenda. Tem a capacidade de criar, editar e eliminar o seu próprio perfil de utilizador no sistema. Pode inscrever-se em conferências e assistir às mesmas. Autentica o seu acesso utilizando um cartão ou outro método apropriado de identificação.
- **Aluno:** O Aluno é uma subcategoria do Utilizador, com funcionalidades específicas relacionadas ao contexto académico. Pode inscrever-se em disciplinas e assistir às aulas das mesmas.
- **Docente:** O Docente é outra subcategoria do Utilizador, com funcionalidades adicionais em relação ao contexto académico. Além das funcionalidades do Utilizador, pode consultar as presenças registadas durante as aulas que ministra. Utiliza as funcionalidades do sistema

para acompanhar a participação dos alunos nas suas aulas e monitorizar o seu desempenho académico.

7.2.2 Sistema de Gestão de Salas

Nesta secção, é descrita a arquitetura do sistema de gestão de salas e as tecnologias utilizadas na sua implementação. A arquitetura é dividida em três camadas principais: a camada de persistência de dados, a camada de negócios e a camada de apresentação.

- **Camada de Persistência de Dados:** Foi utilizada a tecnologia *Hibernate* para mapear as classes do modelo de dados em tabelas da base de dados. O *Hibernate* permite a persistência dos objetos do modelo numa base de dados relacional de forma transparente. As classes de entidade e os mapeamentos correspondentes são gerados automaticamente a partir do modelo no *Visual Paradigm*.
- **Camada de Negócios:** Foram criadas classes adicionais para gerir a lógica de negócios do sistema. Estas classes encapsulam a lógica de criação, edição e eliminação de eventos, salas, disciplinas, docentes e cursos, além de consultas de ocupação e presenças, entre outras funcionalidades. Implementam as regras de negócios e fazem uso das classes de entidade geradas pelo *Hibernate* para aceder e manipular os dados do sistema. Foi utilizado o servidor aplicacional *Wildfly* para hospedar a aplicação. Foram criados *servlets* para processar os pedidos dos utilizadores e reencaminha este para poder serem geradas as respostas apropriadas, bem como *beans* para a preservação e gestão dinâmica do número de conexões persistentes à base de dados.
- **Camada de Apresentação:** As páginas JSP (*JavaServer Pages*) foram utilizadas para a criação da interface de *front-end*, permitindo a geração dinâmica de conteúdo HTML com base nas informações do sistema.

Com isto, podemos enunciar a utilização do padrão arquitetural no qual fizeram parte entidades fulcrais do sistema, que importa salientar. Assim, o padrão utilizado foi o *Model-View-Controller* (MVC), no qual importa salientar o papel das entidades *servlets* cujo o papel é o do *Controller*, no qual realiza a comunicação entre as entidades *Model* e *View*, tal como indicado previamente. Relativamente à entidade *View*, esta é desempenhada pelas entidades *JSP*, relativas às diversas páginas web com as quais o utilizador irá interagir, sendo responsável por receber os pedidos e fornecer as respostas previamente geradas pelo *Model*. Por fim, a entidade *Model* é desempenhada pelas entidades *beans* que além de terem o propósito acima descrito, realizam as tarefas de gerar as respostas, através do processamento dos pedidos reencaminhados pela entidade *Controller*.

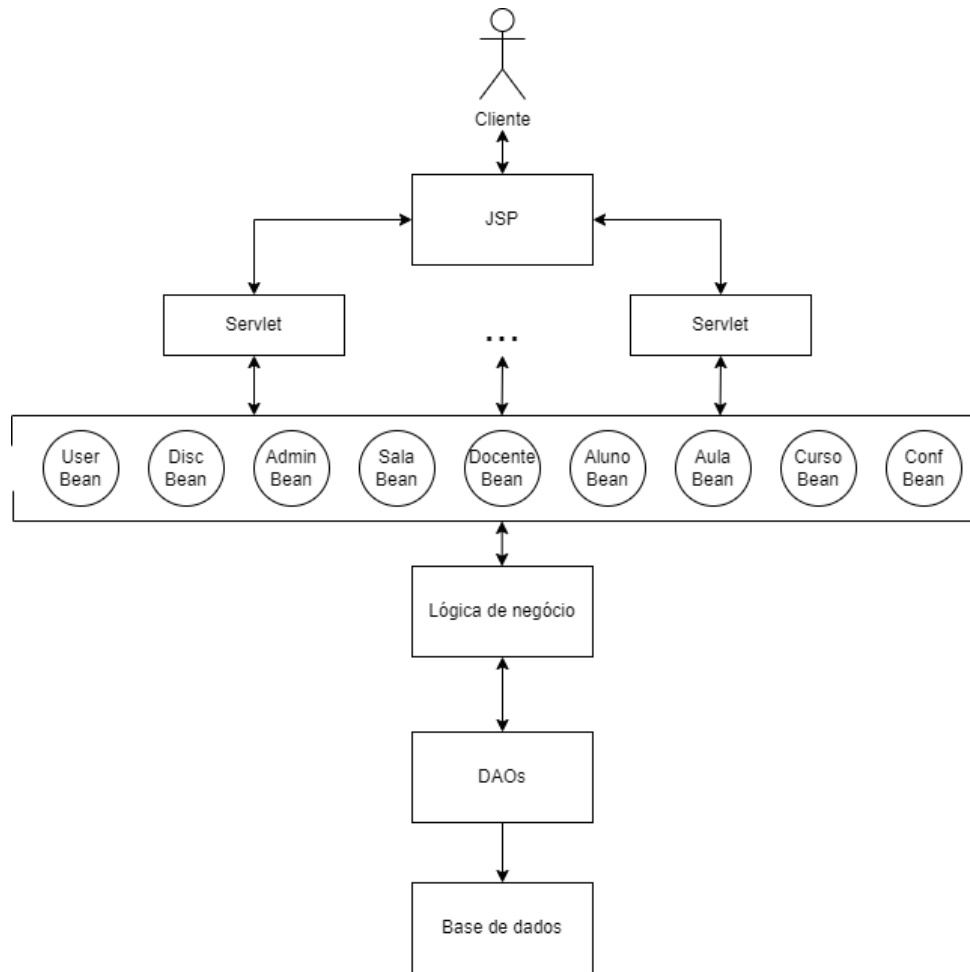


Figura 7.1: Arquitetura do sistema

Esta arquitetura permite uma separação clara das responsabilidades, facilitando a manutenção, extensibilidade e escalabilidade do sistema. As tecnologias utilizadas, como *Visual Paradigm*, *Hibernate*, *Wildfly* e os componentes da framework *Java Enterprise Edition* (JEE), como os *servlets*, os *beans* (EJB) e as JSP, fornecem uma base sólida para a implementação de um sistema eficiente e robusto de gestão de salas em instituições de ensino.

8. Estratégia da Solução

Neste capítulo, é abordada a estratégia *Model Driven Development* adotada na construção da nossa aplicação. Exploraremos como o uso de ferramentas como o *Visual Paradigm*, o *Hibernate*, *JavaBeans* e *JavaServer Pages* (JSP) foram empregados para desenvolver a solução de forma eficiente e sustentável.

O *Model Driven Development* é uma abordagem que enfatiza a criação e utilização de modelos como partes fundamentais durante todo o ciclo de vida do desenvolvimento de software. Estes modelos capturam os requisitos, a lógica de negócios, a estrutura de dados e outros aspectos essenciais do sistema, permitindo uma representação visual clara e concisa do mesmo.

8.1 Ferramentas utilizadas

No desenvolvimento da aplicação SGS foram utilizadas diversas ferramentas, como: *Visual Paradigm*, *Hibernate*, *Enterprise JavaBeans* e *JavaServer Pages*. O *Visual Paradigm* é uma ferramenta de modelagem poderosa e abrangente utilizada para criar os modelos necessários para representar a estrutura e o comportamento do sistema. Quanto ao *Hibernate*, este é um framework de mapeamento objeto-relacional (ORM) amplamente utilizado na camada de persistência. Através do *Visual Paradigm*, foi possível gerar automaticamente o código Java correspondente às entidades de negócio e aos mapeamentos do *Hibernate*. O servidor aplicativo utilizado é baseado em *JavaBeans*, que fornecem uma infraestrutura robusta para o desenvolvimento de aplicações em Java. As páginas web foram desenvolvidas utilizando *JavaServer Pages* (JSP), permitindo a criação dinâmica de conteúdo web e a integração perfeita com os componentes *JavaBeans*.

8.2 Conceção dos modelos e geração de código

O processo de implementação da solução foi iniciado com o desenvolvimento de modelos no *Visual Paradigm*, representando as entidades de negócio, as relações entre elas e a lógica do sistema. Estes modelos foram refinados e validados meticulosamente, garantindo uma compreensão comum do sistema. No final desta fase foi obtido o seguinte *Platform Specific Model* (PSM):

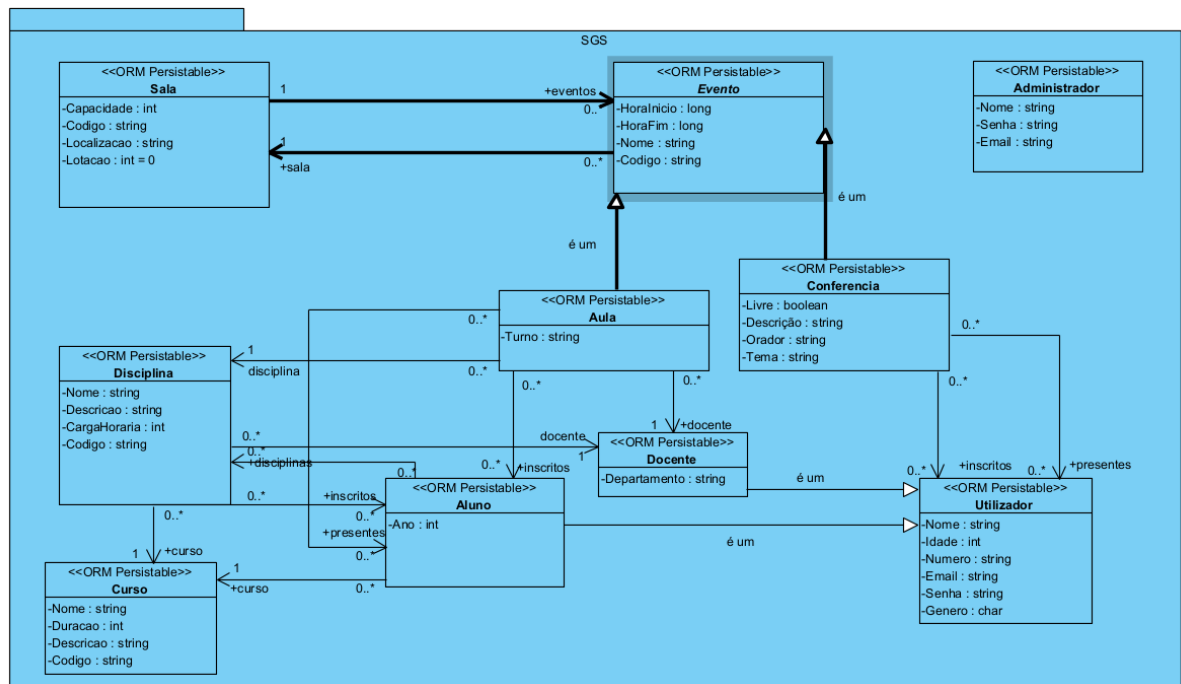


Figura 8.1: *Platform Specific Model (PSM)*

Com os modelos validados, foi utilizada a funcionalidade de geração de código do *Visual Paradigm* para produzir automaticamente as classes Java correspondentes às entidades de negócio e aos mapeamentos do *Hibernate*. Isto permitiu economizar tempo e manter a consistência entre os modelos e o código gerado.

8.3 Implementação da lógica de negócios

Apesar da geração automática de código a partir dos modelos, algumas funcionalidades específicas requeriam a implementação de uma lógica de negócios personalizada. Estas funcionalidades envolvem operações de criação, atualização, eliminação de objetos, ou até regras de negócio mais específicas. Para atender a estas necessidades, foram criadas classes dedicadas à implementação destas lógicas.

Para implementar a lógica de negócios do sistema, foram criadas várias classes Java regulares, incorporando método necessários para realizar as operações de negócio específicas. Estas classes foram integradas no sistema através de chamadas apropriadas a partir de outras partes do código. Isto inclui a interação com as classes geradas a partir dos modelos, a fim de fornecer os dados necessários para a execução das operações de negócio.

8.4 Criação do servidor aplicacional

O servidor aplicacional foi criado utilizando o servidor *Wildfly*. Este atua como um ambiente de execução para a aplicação, permitindo gerir sessões, transações e outros aspetos importantes do sistema. Para implementar o servidor aplicacional foram utilizadas ferramentas pertencentes ao *Java Enterprise Edition (JEE)*, tais como os *servlets* e *beans (EJB)*. Primeiramente foram criados os *servlets*, sendo estes componentes Java que são executados no servidor aplicacional e lidam com pedidos e respostas HTTP. Estes fornecem uma maneira eficiente e flexível de tratar os pedidos do cliente e gerar respostas dinâmicas. Os *servlets* foram utilizados para receber e processar as requisições provenientes das páginas web, interagindo com as classes da lógica de negócios e retornando as respostas apropriadas para o cliente.

8.5 Implementação da interface gráfica

Como forma de interagir com o sistema de maneira fácil e eficaz foi criada uma interface gráfica. Recorrendo ao uso da ferramenta JSP esta foi dividida em duas partes fundamentais, o ecrã das salas e o *website* dos utilizadores (alunos, professores, visitantes e administrador). Ambos as interfaces gráficas permitem a troca de informações entre a *front-end* e a *backend*, tanto a partir dos atributos utilizados na *session* e nos *requests* como através do envio de pedidos POST recorrendo a javascript

Como forma de interagir com o sistema de maneira fácil e eficaz, foi desenvolvida uma interface gráfica para a aplicação. Utilizando a ferramenta JSP (JavaServer Pages), a interface gráfica foi dividida em duas partes essenciais: o painel das salas e o website dos utilizadores (alunos, professores, utilizadores e administrador). Ambos as interfaces gráficas permitem a troca de informações entre a *front-end* e a *backend*, tanto a partir dos atributos utilizados na *session* e nos *requests*, como através do envio de pedidos *POST* recorrendo a *JavaScript*.

- **Ecrã das salas:** O ecrã das salas apresenta uma visualização das salas, permitindo que os utilizadores vejam as informações relevantes, como a disponibilidade, capacidade, e informações acerca do evento que está a decorrer ou que pode vir a decorrer. Através da interface gráfica, os utilizadores podem interagir com as salas e consultar informações detalhadas. A implementação desta interface utiliza JSP para renderizar as informações e JavaScript para controlar aspetos específicos, como a validação de formulários e interações dinâmicas;
- **Website dos utilizadores:** O *website* dos utilizadores abrange diferentes perfis, como alunos, professores, visitantes e administradores. Cada perfil possui uma interface personalizada que permite acesso a funcionalidades específicas. Esta interface gráfica permite aos utilizadores visualizar informações relevantes, realizar ações relacionadas às entidades do sistema e interagir com o sistema de forma intuitiva. O uso de JSP e JavaScript permite a troca de informações entre a camada de *front-end* e a camada de *back-end*, permitindo que os utilizadores enviem solicitações *POST* para executar operações específicas e recebam respostas relevantes.

A integração do JavaScript nas interfaces gráficas permite que os utilizadores tenham uma experiência mais dinâmica e responsiva. O JavaScript é utilizado para validar formulários, fornecer *feedback* em tempo real, atualizar informações na página sem recarregar completamente a interface e melhorar a usabilidade geral do sistema.

8.6 Implementação dos *beans*

Quanto à implementação dos *beans*, esta surgiu mais tarde no ciclo de desenvolvimento da aplicação. Os EJBs são componentes de negócio em Java que fornecem recursos avançados de gestão de transações, segurança, persistência e concorrência. Estes facilitam a implementação de lógica de negócios complexa e oferecem um ambiente que garante a consistência e a integridade dos dados.

9. Visualização em tempo de execução

No Sistema de Gestão de Salas, o processamento de um pedido de um cliente segue um fluxo bem definido. Primeiramente, o cliente envia um pedido ao servidor aplicacional, utilizando a interface de *front-end* implementada em JSP. O pedido pode ser, por exemplo, uma a criação de um novo evento ou uma consulta de presenças.

De seguida, o servidor aplicacional, hospedado no *Wildfly*, recebe o pedido e encaminha-o para o *servlet* correspondente. O *servlet* atua como um controlador, responsável por receber, processar as solicitações dos clientes e reencaminhá-las para o *bean* respetivo. O *servlet*, após receber o pedido, chama o *bean* apropriado para executar a lógica de negócios relacionada ao pedido. Os *beans* encapsulam a lógica de criação, edição, eliminação, consulta e outras operações relacionadas aos componentes da aplicações.

Os *beans*, por sua vez, interagem com as classes que gerem as várias entidades do sistema, sendo que estas utilizam as classes de entidade geradas pelo *Hibernate*, os DAOs, para acessar e manipular os dados do sistema. Isto envolve a consulta, atualização, inserção ou exclusão de informações na base de dados, conforme necessário para atender ao pedido.

Após a execução da lógica de negócios e a manipulação dos dados, o *servlet* recebe o resultado e gera uma resposta adequada ao cliente. Essa resposta pode ser uma página JSP dinamicamente gerada com informações relevantes, uma mensagem de confirmação ou qualquer outro tipo de retorno necessário. A resposta é enviada de volta ao cliente através do servidor aplicacional, que a entrega ao navegador do cliente para este proceder à sua exibição.

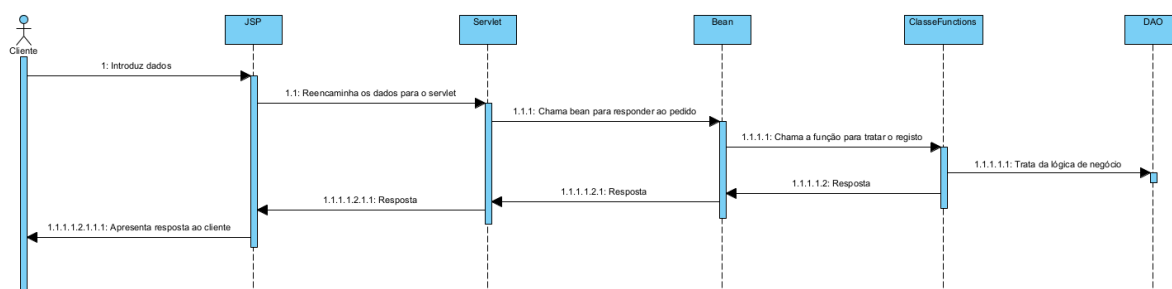


Figura 9.1: Diagrama de sequência de resposta a um pedido do cliente

O fluxo descrito mostra como o pedido do cliente é processado de forma coordenada por diferentes componentes da arquitetura. O servidor aplicacional, os *servlets*, os *beans*, os DAOs e as páginas JSP trabalham em conjunto para fornecer as funcionalidades solicitadas pelo cliente, garantindo a correta execução da lógica de negócios e a integração com a base de dados.

10. Avaliação crítica

10.1 Implementação do servidor aplicativo

No desenvolvimento da aplicação, o servidor aplicativo foi implementado utilizando ferramentas *Java Enterprise Edition* (JEE). O uso destas ferramentas permitiu criar uma plataforma robusta e escalável para hospedar esta aplicação web. No entanto, é importante mencionar que, existem maneira de melhorar ainda mais esta implementação do sistema de gestão de salas. Algumas funcionalidades não implementadas, como o balanceamento de carga através do *mod_proxy* e a disponibilização do serviço através do *Web Services Description Language* (WSDL), permitiam um aumento da qualidade final da aplicação.

Com o balanceamento da carga o tráfego da aplicação iria ser distribuído entre vários servidores. Isto poderia evitar sobrecargas num único servidor, melhorando o desempenho, a disponibilidade e a escalabilidade do sistema. O WSDL é uma especificação que descreve a interface pública de um serviço web. Com isto, é possível fornecer informações detalhadas sobre os métodos disponíveis, os formatos de dados aceites e as operações que podem ser executadas no serviço.

10.2 Padrões de design na interface

Ao desenvolver a interface da aplicação, foi reconhecida a importância de fornecer aos utilizadores uma experiência intuitiva e eficiente. Para alcançar esse objetivo, optamos por incorporar alguns padrões de design específicos, como por exemplo: *breadcrumbs*, *calendar picker* e *alternating row colors*.

Criar Docente

[Menu Principal](#) > [Gestão de Docentes](#) > Criar Docente

Figura 10.1: Padrão de design *breadcrumbs*

Codigo	Nome	Dia/Hora de Inicio	Dia/Hora de Fim	Sala	E
AA_SIC	Apresentac	23/06/202	23/06/2023 12:00	cp2-0.10	
AA_SIC_Projeto	Projeto de AA	19/06/202	junho de 2023	12 00	
AssembG22/23	Assembleia C	26/06/202	D S T Q Q S S	13	
CeslumSCamp22/23	Ceslum Sumi	23/06/202	26 29 30 31 1 2 3	14	01
TM	Teses de Met	26/06/202	4 5 6 7 8 9 10	15	02
			11 12 13 14 15 16 17	16	03
			18 19 20 21 22 23 24	17	04
			25 26 27 28 29 30 1	18	05
			2 3 4 5 6 7 8		
			Limpar Hoje		

Figura 10.2: Padrão de design *calendar picker*

Código	Nome	Dia/Hora de Início	Dia/Hora de Fim	Sala	Entrada Livre	Descrição	Orador	Tema	Ações
AA_SIC	Apresentação	23/06/2023	23/06/2023	cp2-0.10	<input type="checkbox"/>	Apresentação	Antonio Nestor	Projetos Práticos	Edit Delete
AA_SIC_Projeto	Projeto de AA	19/06/2023	19/06/2023	cp2-0.02	<input type="checkbox"/>	Desenvolvimento	Francisco Reis	Java Web Dev	Delete
AssembG22/23	Assembleia Ge	26/06/2023	26/06/2023	cp2-1.05	<input type="checkbox"/>	Assembleia Ge	Manuel Pinto V	Discussão de li	Edit Delete
CesiumSCamp22/23	Cesium Summ	23/06/2023	23/06/2023	cp2-0.09	<input checked="" type="checkbox"/>	Summer Camp	Carlos Felipe A	Disponibilização	Edit Delete
TM	Teses de Mestr	26/06/2023	26/06/2023	cp2-2.01	<input checked="" type="checkbox"/>	Discussão sobre	Antonio Luis SI	Reuniao	Edit Delete

Figura 10.3: Padrão de design *alternating row colors*

Com a integração destes padrões de design na interface, é garantida uma experiência mais intuitiva, orientada e visualmente agradável aos utilizadores na interação com o nosso sistema. Isto contribui para a usabilidade geral da aplicação e para uma interação mais eficiente e agradável.

10.3 Avaliação heurística

Durante o processo de desenvolvimento da nossa aplicação, foi realizada uma avaliação heurística da interface do sistema. Esta avaliação teve como objetivo identificar possíveis problemas de usabilidade e fornecer recomendações para melhorar a experiência do utilizador. Nesta secção, são discutidos os detalhes dessa avaliação e as principais conclusões obtidas.

Heurística	Dificuldades	Oportunidades de melhoria
Visibilidade do estado do sistema	O estado do sistema é sempre comunicado ao utilizador.	Nenhuma
Correspondência entre o sistema e o mundo real	O sistema utiliza linguagem básica e amigável ao utilizador.	Nenhuma
Controlo e liberdade do utilizador	O sistema apresenta sempre botões de retorno.	Nenhuma
Consistência e normas	É mantida a consistência nas ações e na linguagem no sistema.	Nenhuma
Prevenção de erros	O sistema previne erros, através de mecanismos como opções de seleção, entre outros (menus dropdown).	Nenhuma
Reconhecimento em vez de recordação	O utilizador não necessita de decorar informações relativas ao sistema, pois este utiliza opções de seleção.	Nenhuma
Flexibilidade e eficiência de uso	Por exemplo eliminar várias aulas de um vez. As aulas só podem ser eliminadas uma a uma.	Poder seleccionar várias aulas e eliminá-las ao mesmo tempo.
Design estético e minimalista	O sistema apresenta um design minimalista com informação bem compactada.	Nenhuma
Ajudar os utilizadores a reconhecer, diagnosticar e recuperar de erros	O sistema apresenta mensagens de erro claras, com a indicação do problema a acontecer.	Nenhuma
Ajuda e documentação	O sistema não possui documentação.	Criar uma página de documentação a explicar as várias ações possíveis.

Tabela 10.1: Avaliação heurística da interface do sistema

De relembrar que esta tabela apenas é relativa à versão final da aplicação, sendo que ao longo do desenvolvimento da interface as heurísticas apresentadas eram tidas em conta, sendo efetuadas melhorias progressivas na medida de satisfazer estas questões.

11. Conclusões e Trabalho Futuro

Neste projeto, foi desenvolvido um sistema de gestão de salas para instituições de ensino, visando aprimorar a organização e a utilização eficiente dos espaços disponíveis. Ao longo do desenvolvimento, foram aplicados diversos conceitos e ferramentas abordados durante as aulas, resultando numa solução robusta e funcional.

A arquitetura do sistema, baseada numa arquitetura multi-camada, permitiu uma separação clara das responsabilidades e facilitou a manutenção e a extensibilidade do código. O uso do *Hibernate* como framework de mapeamento objeto-relacional simplificou a persistência dos dados na base de dados.

A implementação do servidor aplicativo com *Wildfly* proporcionou um ambiente confiável para a execução do sistema, enquanto os *servlets* e *beans* foram responsáveis por processar as solicitações dos utilizadores e executar a lógica de negócios. As páginas JSP forneceram uma interface de *front-end* dinâmica e interativa para os utilizadores.

Para trabalhos futuros, há diversas possibilidades de expansão e aprimoramento do sistema. Uma área de interesse é a implementação de balanceadores de carga, como o *mod_proxy*, para distribuir os pedidos dos clientes entre servidores e garantir alta disponibilidade e escalabilidade do sistema. Isto permitiria lidar com um maior número de utilizadores e eventos simultaneamente.

Outra oportunidade de aprimoramento seria realizar uma avaliação analítica da interface do sistema por parte de entidades que não estivessem envolvidas no desenvolvimento deste projeto, na procura de identificar possíveis melhorias de usabilidade e da experiência do utilizador. Isto pode ser feito por meio de técnicas de avaliação de interfaces, como testes de usabilidade, análise heurística e coleta de feedback dos utilizadores.

Concluindo, este projeto de sistema de gestão de salas demonstrou a aplicação prática dos conceitos e ferramentas lecionados nas aulas. A arquitetura modular e as tecnologias utilizadas forneceram uma base sólida para a construção de um sistema eficiente e funcional. Com trabalho futuro, como a implementação de balanceadores de carga e a avaliação analítica da interface, é possível aprimorar ainda mais a solução, atendendo às necessidades das instituições de ensino e proporcionando uma melhor experiência para os utilizadores.