**Introdução**

O restaurante *Lar de Descanso da Dona Xinga*, situado no centro de Luanda, Angola, é um estabelecimento com elevada afluência, apesar de dispor de apenas 15 mesas. A grande procura resulta frequentemente em longos tempos de espera por um lugar, que podem variar entre 30 minutos a uma hora, mesmo em dias considerados "rápidos". Face a este cenário, a proprietária decidiu procurar uma solução tecnológica que permitisse aos clientes efetuar reservas antecipadas, com o intuito de reduzir os tempos de espera e melhorar a experiência global de atendimento.

**Situação atual do restaurante**

Atualmente, o restaurante enfrenta uma perda significativa de clientes, causada por dois fatores principais: a ausência de um sistema eficiente de reservas antecipadas e a má gestão das reservas existentes, que são efetuadas de forma manual e desorganizada. Esta situação tem levado a longos tempos de espera, frustração por parte dos clientes e, consequentemente, à diminuição da taxa de retorno ao estabelecimento. Face a este cenário, o restaurante entrou em contacto connosco com o objectivo de encontrar uma solução tecnológica que permita otimizar a gestão de reservas e melhorar a experiência de atendimento.

**Problemática**

Tendo em conta a situação atual do restaurante e os fatores que têm contribuído para a perda de clientes, levanta-se a questão: **que solução tecnológica poderá ser implementada para resolver estes problemas?** Uma das respostas possíveis passa pela **criação de uma plataforma online que permita efetuar reservas de forma simples e eficiente,** reduzindo assim os tempos de espera e melhorando a experiência dos clientes.

**Justificativa**

Porquê um site online? À partida, poderia considerar-se uma solução mais simples — como, por exemplo, a criação de uma linha telefónica exclusiva para a gestão de reservas. No entanto, este método apresenta várias limitações: a impossibilidade de atender múltiplas chamadas em simultâneo, a sobrecarga da equipa de atendimento, o risco de erros manuais e a ausência de um registo centralizado e fiável.

A implementação de um sistema online apresenta diversas vantagens:

* Organização centralizada das reservas;
* Redução do esforço manual e da margem de erro;
* Acesso imediato à informação relevante;
* Melhoria da experiência tanto para os clientes como para a equipa do restaurante.

**Objetivo geral**

Desenvolver um website funcional para o restaurante Lar de Descanso da Dona Xinga, com o objectivo de agilizar o processo de tratamento e gestão de reservas, melhor.

**Objetivo específicos**

O objectivo deste projeto é desenvolver uma página web funcional que permita:

* O registo de novos clientes e respective autenticação (login);
* A reserva de mesas com indicação da data, hora e número de pessoas;
* A consulta e atualização do estado das reservas, em tempo real, por parte do pessoal do restaurante.
* Funcionalidade de uso para os diferentes tipos de utilizadores do sistema, sejam clientes, recepcinista e administradores.

Adicionalmente, os administradores terão acesso ao sistema para gerir reservas, controlar a lotação do espaço e supervisionar a distribuição das mesas.

**Escopo e Limitações**

Este projeto, como desenvolvido no âmbito acadêmico da disciplina de Programação Web, decidimos que às características adequadas para nos como iniciantes para se focar:

* Utilização de tecnologias web fundamentais
* Implementação simplificada com uso mínimo de JSON
* Foco na aplicação prática de conceitos básicos
* Sistema funcional, mas com escopo limitado para fins educativos

**Referencial teórico**

**Sistema de reservas online**

Os sistemas de reservas online podem ser considerados como uma evolução natural na gestão de vários estabelecimentos comerciais, especialmente no setor de restauração. Estes sistemas permitem aos clientes efetuar reservas de forma autônoma, reduzindo a carga de trabalho dos funcionários e minimizando erros humanos.

Tecnologias de Desenvolvimento Web

Front End

HTML (HyperText Markup Language): Linguagm de marcação que estrutura o conteúdo da página, incluindo texto, imagens e links.

CSS (Cascading Style Sheets): Define o design visual, como cores, estilos e organização da página, tornado a interface mais apelativa para o usuário.

JavaScript: Linguagem de programação responsável por adicionar interação e dinamismo à página, como atualizações de conteúdo em tempo real.

Backend

O backend sera implementado em PHP, uma linguagem de programação executada no servidor, utilizada para processar pedidos, aceder a base de dados e gerar o conteúdo dinâmico em HTML

Banco de dados

A gestão dos dados será efetuada com MySQL, um sistema de gestão de bases de dados relacionais. Os dados serão organizados em tabelas compostas por colunas (atributos) e linhas (registos), permitindo armazenar, consultar, atualizar e eliminar informação de forma estruturada.

Padrões de Desenvolvimento

Arquitetura MCV (Model-View-Controller)

O padrão MVC é uma arquitetura de página web que separa a aplicação em três componentes interconectados que desempenham funções distintas:

* Model: Representa os dados e a lógica de negócio
* View: Interface de usuário e apresentação
* Controller: Intermediário entre os dois componentes mencionado, controlando o fluxo de dados

Metodologias de desenvolvimento de software

Modelo Cascata (Waterfall)

Modelo em que as etapas do projeto são executadas de forma sequencial, facilitando o planeamento e a documentação do nosso trabalho

Elementos Ágil

Incorporado os elementos do modelo ágil, nomeadamente o desenvolvimento em sprints ou fases incrementais, permitindo maior flexibilidade e capacidade de adaptação a imprevistos que podem surgir durante a criação desse projeto.

**Metodologia**

O nosso trabalho caracteriza-se como uma pesquisa aplicada de natureza experimental, com abordagem pratica voltada para a resolução de problemas reais através do desenvolvimento de uma solução tecnológica.

Abordagem de desenvolvimento

Optou-se por uma abordagem híbrida que conjuga dois modelos de desenvolvimento:

* Modelo Waterfall (em cascata): Etapas executadas de forma sequencial, facilitando o planeamento e a documentação
* Elementos do modelo Agile: Desenvolvimento em sprints (fases incrementais), como a implementação inicial do sistema de login antes da funcionalidade de reservas, permitindo maior flexibilidade

Ferramentas Utilizadas

Ambiente de Desenvolvimento

Para o desenvolvimento do site, foi utilizada a IDE Visual Studio Code. A simulação do servidor local foi efetuada através do ambiente XAMPP, o qual inclui o servidor Apache e o sistema de gestão de bases de dados MySQL, utilizado para o armazenamento e manipulação dos dados da aplicação.

Ferramentas de IA

Durante a criação desse projeto foi utilizado diversas ferramentas de inteligência artificial como apoio ao desenvolvimento para cobrir a nossa falta de experiência e compressão de dificuldades encontradas:

* ChatGPT: Foi muito útil para criar elementos gráficos como o logótipo e as imagens usadas nas cartas de ocasiões. Também nos ajudou a esclarecer dúvidas rápidas e a gerar pequenos trechos de código quando ficávamos presos em algum detalhe.
* Claude Sonnet 4: Usado para explicar assuntos mais complicados e para nos ajudar a estruturar melhor partes do sistema, como os ficheiros JavaScript. Serviu como apoio na criação e revisão de código mais complexo.
* Grok: Foi uma das primeiras IAs que usámos. Ajudou-nos a pensar na estrutura geral do trabalho, tanto na parte prática como na documentação. Gerou os primeiros códigos e levantou questões que nos obrigaram a pensar melhor sobre o que estávamos a fazer, o que acabou por melhorar o nosso raciocínio lógico.
* QODO: Começámos a usar o QODO mais perto do final do projeto. A grande vantagem dele foi conseguir juntar e usar várias IAs ao mesmo tempo, analisando o projeto como um todo por meio de modelos de protocolo de contexto (MCP). Isso ajudou-nos a perceber onde podíamos melhorar e acelerou bastante a fase final do desenvolvimento.
* Creati: Foi usado principalmente para dar ideias de design e funcionalidade para os dashboards, tanto o do recepcinista como o do administrador. Ajudou-nos a imaginar uma interface mais funcional e amigável.

**Motivação**

Este projeto, além de corresponder a uma exigência académica no âmbito da disciplina de Programação Web, representa também uma oportunidade prática para aplicar conhecimentos teóricos e explorar, de forma concreta, o processo de criação de uma solução digital com impacto real.

Para muitos dos intervenientes, trata-se do primeiro contacto com o desenvolvimento web aplicado, o que reforça o valor pedagógico e profissional deste desafio.

**Análise e levantamento de requisitos**

Análise de Stakeholders

O sistema desenvolvido foi pensado para responder às necessidades de diferentes tipos de utilizadores (stakeholders), cada um com objectivo e benefícios específicos que justificam e incentivam a adoção da plataforma:

* Clientes: Necessitam de uma interface simples e intuitiva para efetuar reservas online. O principal benefício é a facilidade e rapidez no processo de marcação, o que pode contribuir para o aumento da frequência de visitas ao restaurante.
* Rececionistas: Precisam de uma interface eficaz para gerir o atendimento, controlar as chegadas dos clientes, tratar as reservas e libertar mesas. O sistema ajuda a otimizar estas tarefas, tornando o processo mais ágil, organizado e eficiente.
* Administrador: Requer controlo total sobre o sistema, incluindo o acesso a relatórios e gestão tanto da componente de clientes como da equipa interna. O principal benefício é a centralização da informação e o controlo completo da operação.
* Proprietária do restaurante: Necessita de um sistema funcional e fiável para garantir o bom funcionamento do negócio. Beneficia-se da melhoria na gestão operacional, bem como do acesso a análises e dados que ajudam na tomada de decisões estratégicas.

**Requisito Funcionais**

O website encontra-se dividido em três áreas funcionais distintas:

* Área do Cliente
* Registro e Login: Clientes criam contas com nome, e-mail, telemóvel e senha; login com e-mail e senha redireciona pra página do perfil onde pode observar as credenciais.
* Reservas: Reservar mesas com data, hora e número de pessoas (máximo 60), até 7 dias de antecedência.
* Edição/Cancelamento: Editar ou cancelar reservas até 2 horas antes, com pop-up: “Confirmar cancelamento da reserva?”
* Verificação de Disponibilidade: Botão “Verificar Disponibilidade” exibe pop-up com “X mesas disponíveis” ou “Nenhuma mesa disponível”.
* Histórico de Reservas: Tabela ordenada por data descendente, com colunas: Data, Hora, Nº de Pessoas, Status (Reservado, Confirmando, Concluído, Cancelado, Expirado).
* Área do Rececionista
* Visualização de Reservas: Filtros por nome, telemóvel ou email (Nome, Telemóvel, Email).
* Gestão de Mesas: Atualizar estado de mesas (Livre → Ocupada) com botão “Confirmar”.
* Gestão de Atrasos: Pop-ups ao carregar a tela pra atrasos de 20 minutos (“O cliente compareceu?”) e 30 minutos (confirma “Expirado”); status automático após 40 minutos.
* Clientes sem Reserva: Registrar ocupação manual de mesas (mínimo 1 pessoa)
* Área do Administrador
* **Gestão de Reservas**: Cancelar/editar o estado da reserva no caso de falha.
* **Exclusão de Contas**: Excluir clientes e reservas, com pop-up: “Excluir permanentemente?”, preservando relatórios.
* **Relatórios Diários**: Tabela com Data, Reservas, Mesas Ocupadas, Expirações, Cancelamentos, Percentual de Ocupação.

**Requisitos não-funcionais**

Como já mencionamos os requisitos funcionais especificam como o sistema deve funcionar em termos de tarefas, agora lado dos não-funcionais se refere no desempenho, segurança, usabilidade, compatibilidade, escalabilidade entre outros.

Desempenho

* Verificação de disponibilidade: < 2 segundos (otimizado pra *\*XAMPP\**, 8 GB RAM, 2.5 GHz).
* Histórico de reservas: < 3 segundos (100 registros).
* Relatórios diários: < 5 segundos.

Segurança

* Validação contra SQL injection (usando prepared statements) por meio de PDO no Backend, e JavaScript manter um forma especifica de interação como os campos.
* Senhas hasheadas com password\_hash() para criptografar as senhas e manter a sua proteção na base de dados.
* Dados sensíveis (ex.: e-mails, telemóveis) não exibidos publicamente.

Usabilidade

* Interface com fonte ≥ 16px, alto contraste (ex.: texto preto em fundo branco).
* Mensagens de erro claras (ex.: “E-mail inválido” se faltar “@”).
* Navegação acessível via teclado pra acessibilidade.
* Instruções em formulários (ex.: “DD/MM/AAAA”) pra usuários com pouca experiência.

Compatibilidade

* Suporta os navegadores comuns para ambos computadores e telemóveis.
* Responsiva (320px a 1920px).

Escalabilidade

* 5-8 usuários simultâneos com hardware atual (> 95% de sucesso).
* 1000 reservas/mês com índices no MySQL.

**Regras de Negócio**

As regras de negócio governam a lógica do sistema, garantindo que ações como reservas e alocações sejam consistentes e evitem *\*bugs\** lógicos, como *\*overbooking\** ou alocações erradas. Elas são implementadas via PHP (lógica do servidor) e SQL (consultas ao MySQL):

* **Capacidade das mesas**: O restaurante tem 15 mesas, cada uma com capacidade fixa de ~4 pessoas (máximo total de ~60 pessoas). Reservas devem respeitar a capacidade (ex.: grupo de 5 vai usar duas mesas um com 3 pessoas e outra com 2 pessoas).
* **Alocação de mesas**: O sistema aloca uma ou mais mesas necessárias para dependendo do número de pessoas que fazem parte do grupo. A alocação usa uma consulta SQL pra verificar disponibilidade das mesas.
* **Gestão de atrasos**: Reservas com 40 minutos de atraso são marcadas como “Expirado” automaticamente via PHP, salvo se o rececionista marcar “Concluído” antes (ex.: cliente chega aos 25 minutos). Um *\*timestamp\** no banco (`data`, `hora`) rastreia o tempo.
* **Edição/Cancelamento**: Edição ou cancelamento de reservas é bloqueado dentro de 2 horas do horário reservado, com os botões sendo desabilitado em vez aparece a mensagem. “Não é possivel editar (menos de 2h)”
* **Reservas simultâneas**: O sistema impede *\*overbooking\** verificando a disponibilidade de mesas antes de confirmar uma reserva, usando uma consulta SQL pra contar mesas ocupadas no horário e à atribuição específica das mesas apenas no momento da confirmação da chegada do cliente.

**1.6 Prioridade das Funcionalidades**

|  |  |  |
| --- | --- | --- |
| Funcionalidade | Prioridade | Justificativa |
| Registro e Login | Alta | Essencial para autenticação e acesso ao sistema. |
| Reservas (Criar/Editar/Cancelar) | Alta | Função principal do sistema, exigida por clientes. |
| Verificação de Disponibilidade | Alta | Impede *overbooking*, crítica pra operação. |
| Histórico de Reservas | Alta | Melhora a experiência do cliente, exigida pelo projeto. |
| Área do Rececionista | Alta | Necessária para gerir reservas no restaurante. |
| Área do Administrador (Gestão) | Média | Útil para corrigir falhas, mas menos frequente. |
| Relatórios Diários | Baixa | Agrega valor analítico, mas não é essencial. |

**Comportamento de Erros**

O sistema reage a cenários de erro com mensagens claras pra evitar travamentos e orientar o usuário, usando pop-ups pra interações amigáveis. Validações ocorrem no front-end (JavaScript) pra feedback instantâneo e no back-end (PHP) pra segurança. Exemplos:

* **Data inválida nas reservas**: Se o cliente selecionar uma data anterior ao dia atual ou além de 7 dias (ex.: 28/04/2025 pra hoje 20/04/2025), não sera possível porque o JavaScript foi configurado para apenas aceitar o dia atual e até 7 dias após.
* **Número de pessoas inválido**: Se o cliente inserir um número menor que 1 (ex.: -1, 0) ou maior que a capacidade do restaurante (60 pessoas, com 14 mesas de ~4 pessoas cada), o formulário mostra: “Número inválido. Insira entre 1 e 60 pessoas.”
* **E-mail inválido no registro**: Se o e-mail não contiver “@” ou for inválido (ex.: “joao.gmail”), o formulário exibe: “E-mail inválido. Insira um e-mail válido (ex.: joao@gmail.com).”
* **Login com credenciais incorretas**: Se o e-mail ou senha estiverem errados, um pop-up mostra: “E-mail ou senha incorretos. Tente novamente.”
* **Sem mesas disponíveis**: Se não houver mesas para o horário solicitado, um pop-up avisa: “Nenhuma mesa disponível para o horário selecionado. Tente outro horário.”
* **Falha no banco de dados**: Em caso de erro no MySQL (ex.: servidor offline), a mensagem é: “Serviço temporariamente indisponível. Tente novamente mais tarde.”
* **Busca sem resultados (rececionista)**: Se a busca por nome ou telemóvel não encontrar um cliente, a tabela exibe: “Nenhum cliente encontrado.”

**Limitações do Sistema**

Devido à limitação de experiência e recursos técnicos, o sistema apresentará as seguintes restrições:

* Ausência de sistema de pagamento funcional.
* Número limitado de utilizadores em simultâneo (entre 5 e 8).
* Sem aplicação móvel ou funcionalidades de entrega.
* Formato optimizador apenas para computadores e telemóveis.
* Sem sistema de Backup.
* Sem sistema de notificações por via de massagem ou email.

Estas limitações refletem a natureza preliminar e de pequena escala do estado atual do sistema.

**Público-Alvo**

O público-alvo deste sistema são os clientes do restaurante, frequentes ou ocasionais, que tenham acesso a um computador ou telemóvel e pretendam efetuar reservas remotamente.

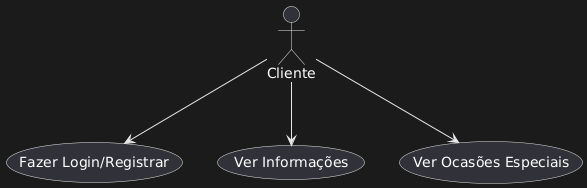
Modelagem e design do sistema

Modelagem de casos de uso

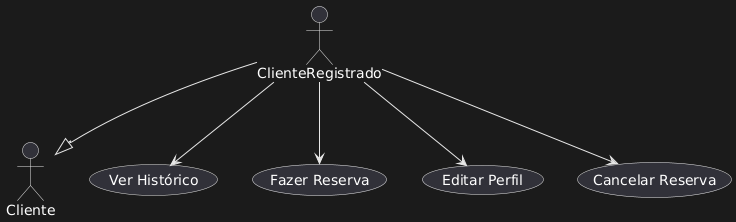
Diagrama de casos de uso

O diagrama de casos de uso ilustra as interações entre os diferentes atores (Cliente, Recepcionista, Administrador) e as funcionalidades do sistema, demonstrando claramente as permissões e responsabilidades de cada tipo de usuário.

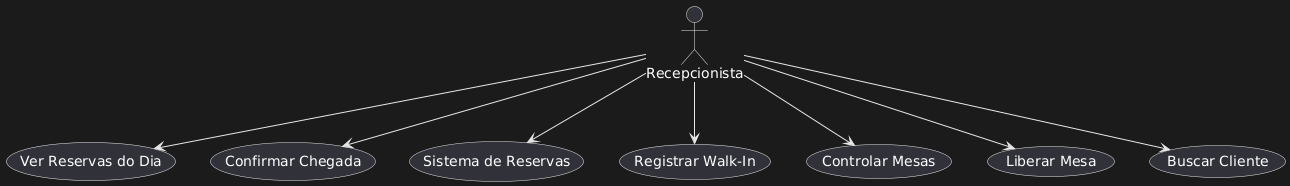
1.Cliente (Visitante)



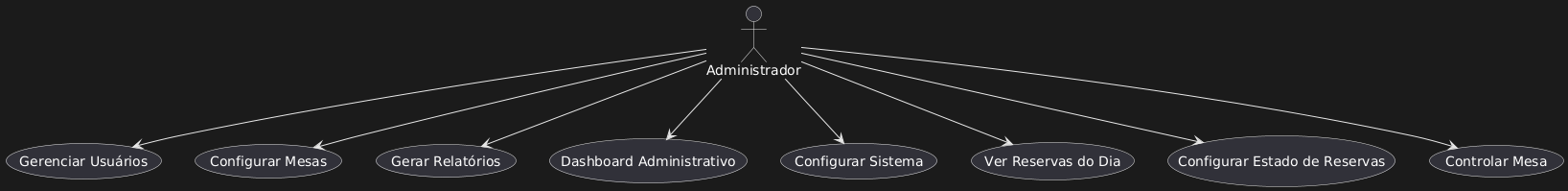
**2.Cliente Registado**

****

**3.Funcionalidades do Rececionista**

****

**4. Funcionalidades do Administrador**

****

**5.Relacionamentos Include/Extend**

****

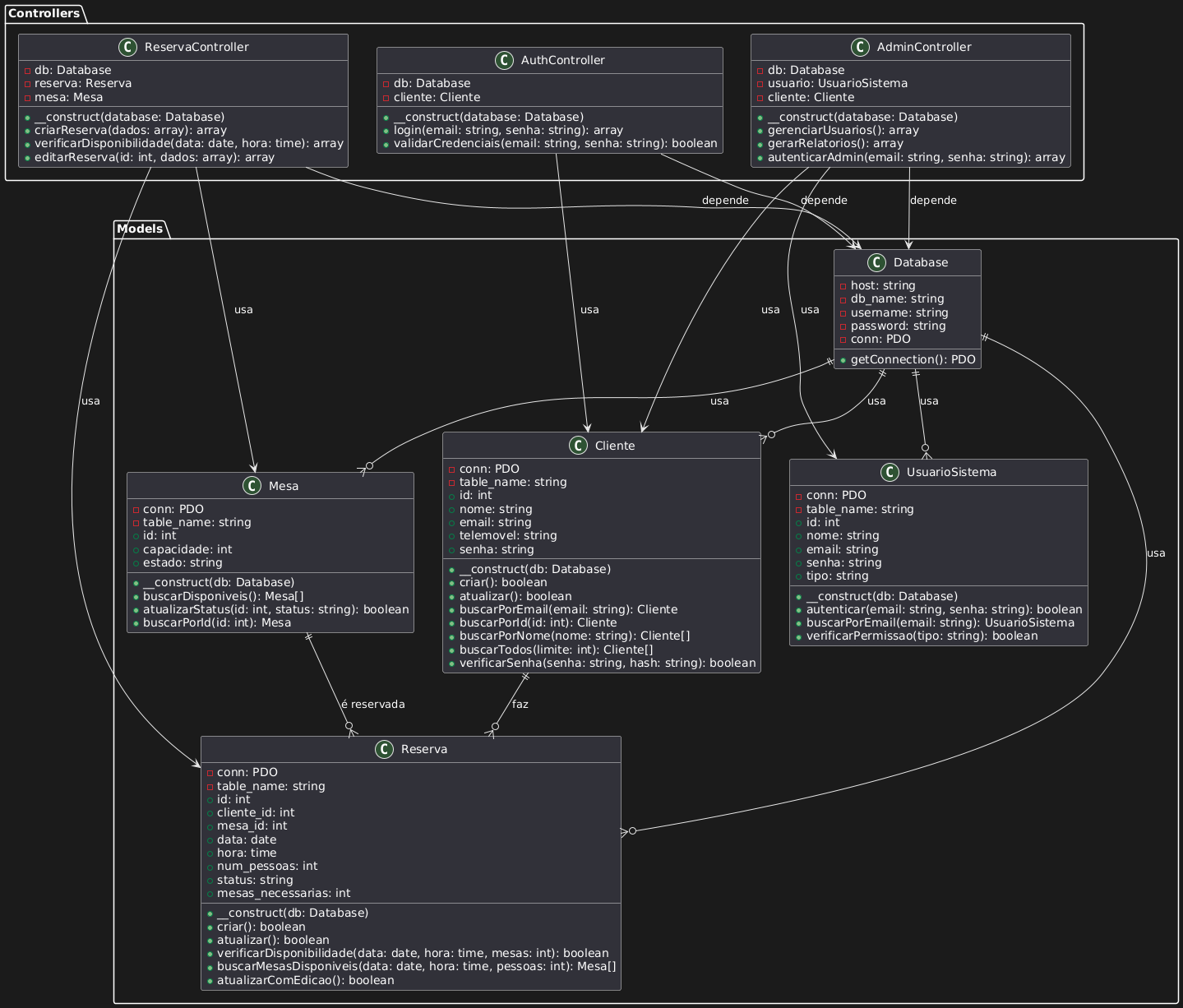
**Resumo dos atores**

|  |  |  |
| --- | --- | --- |
| **Ator** | **Descrição** | **Principais Funcionalidades** |
| **Cliente** | Usuário público do site | Ver informações |
| **Cliente Registrado** | Cliente com conta | Todas do Cliente + histórico e perfil |
| **Recepcionista** | Funcionário operacional | Gerenciar reservas diárias e mesas |
| **Administrador** | Gestor do sistema | Todas as anteriores + configurações |

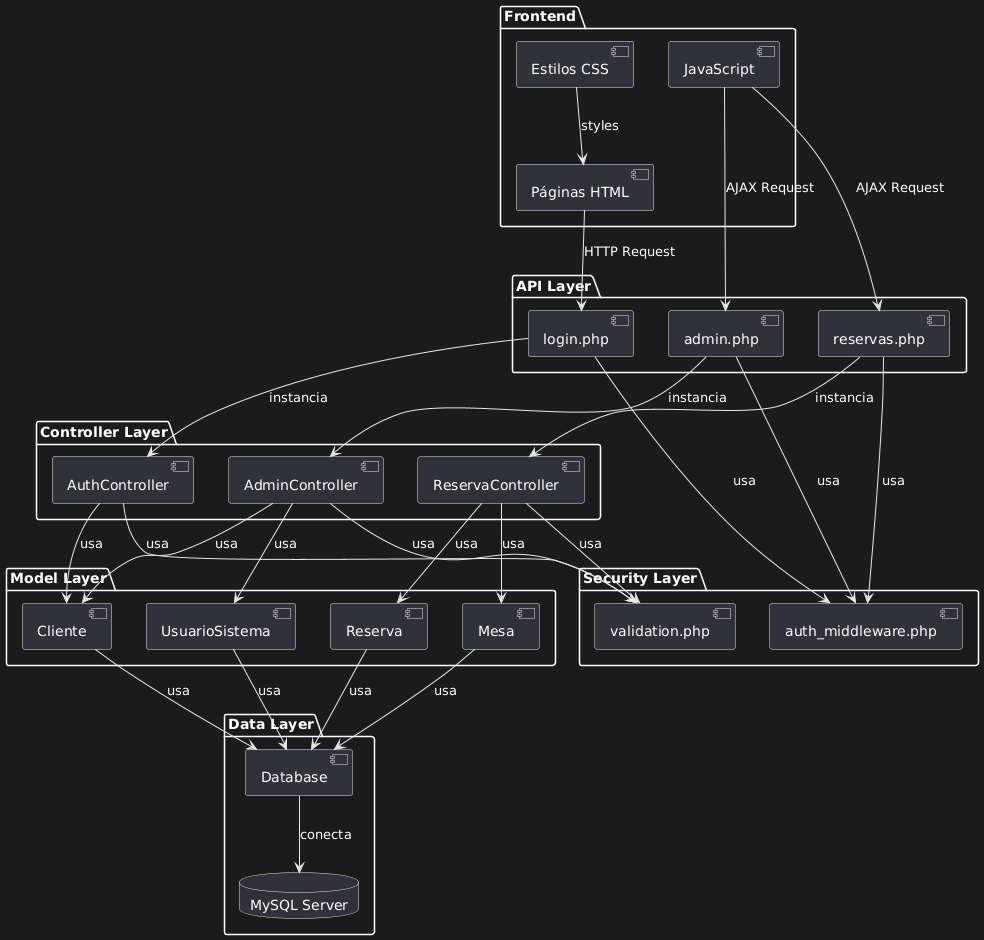
**Modelagem Estrutural**

A estrutura de classes reflete a implementação do padrão MVC, com separação clara entre modelos de dados, controladores de lógica e views de apresentação.

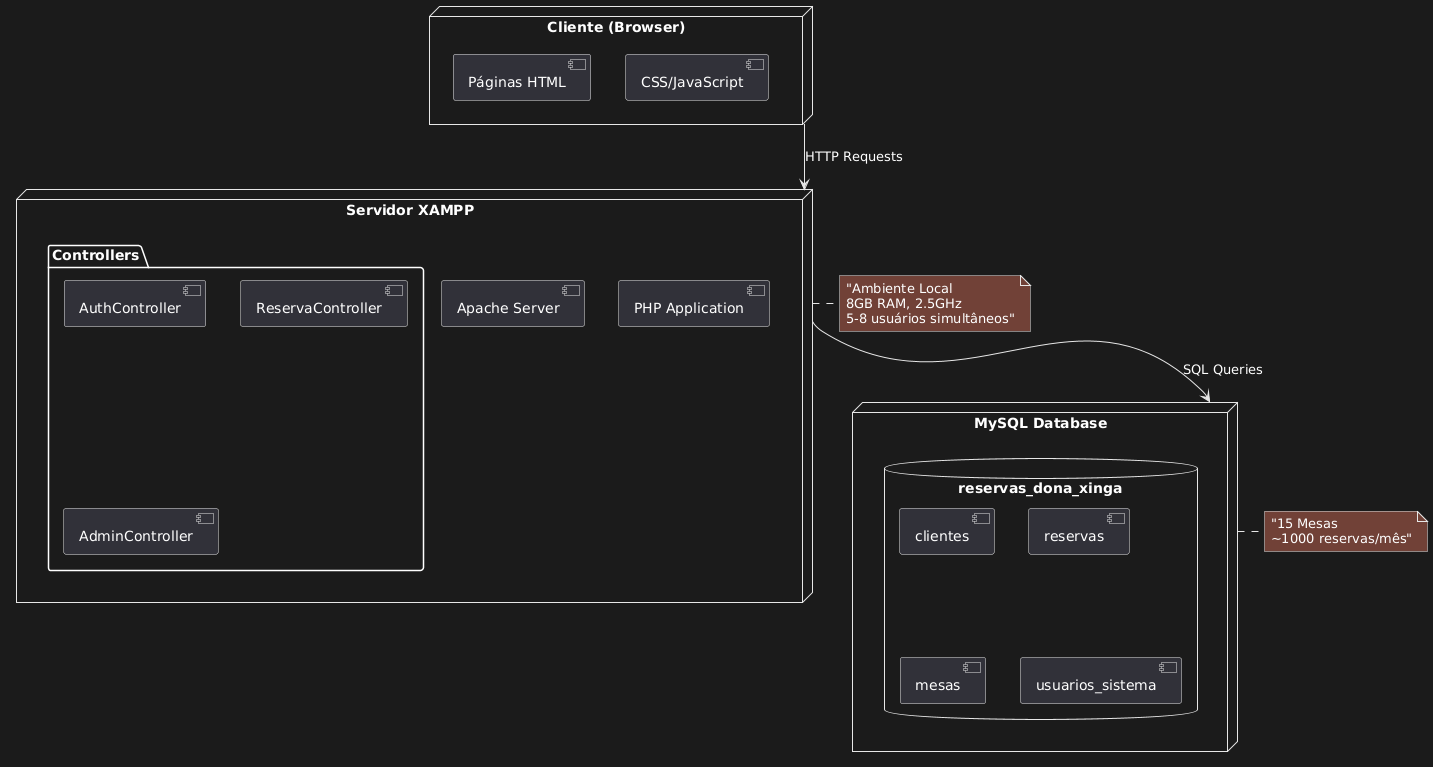
1.Diagrama de classes

****

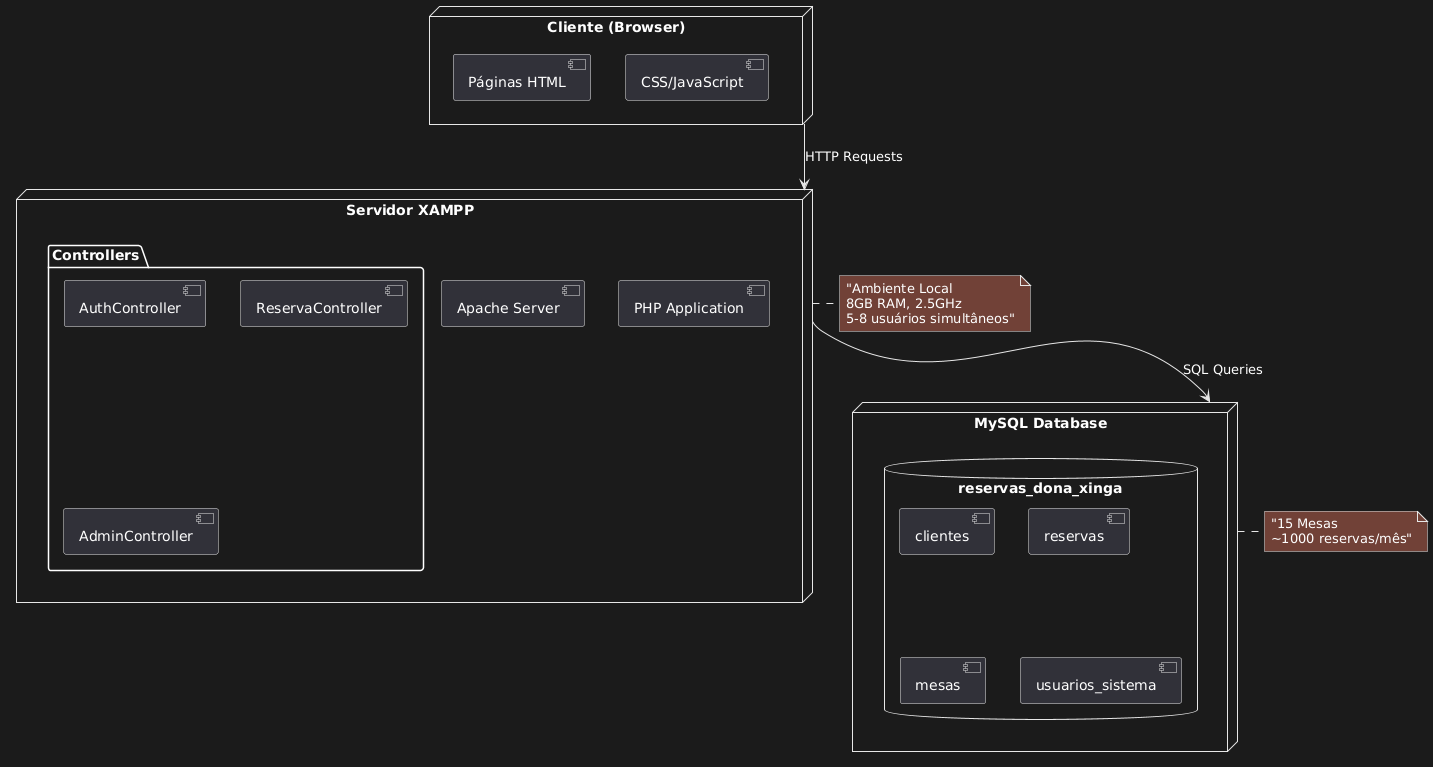
2.Diagrama de componentes



3. Diagrama de Implementação



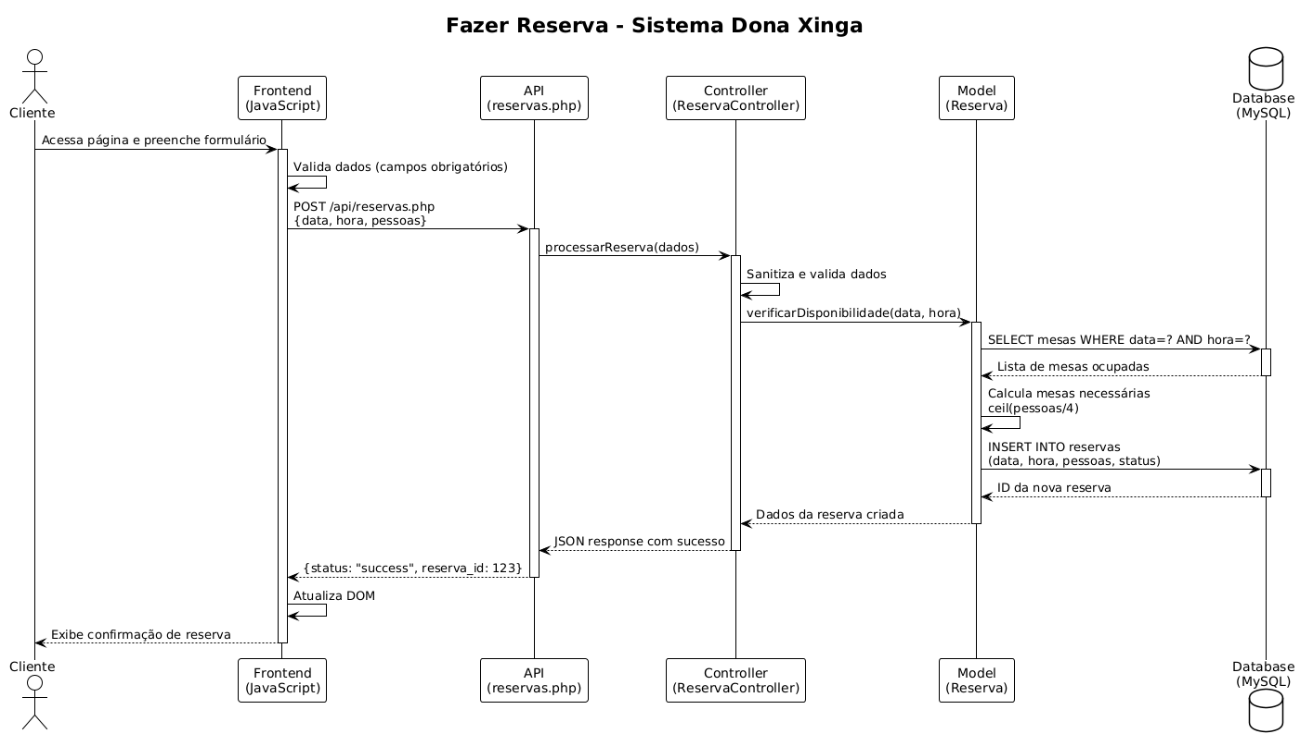
4. Diagrama de Pacotes



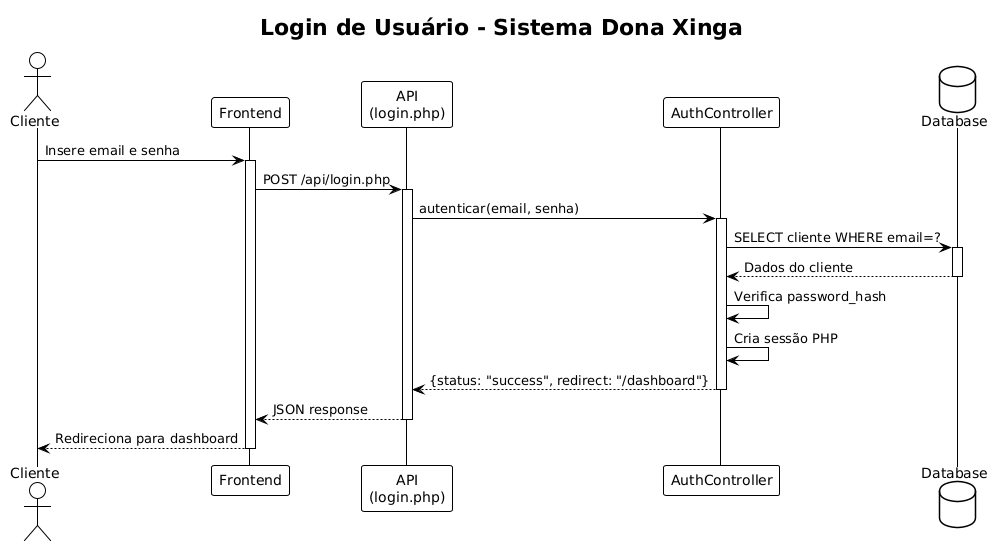
Modelagem Comportamental

Diagrama de sequência

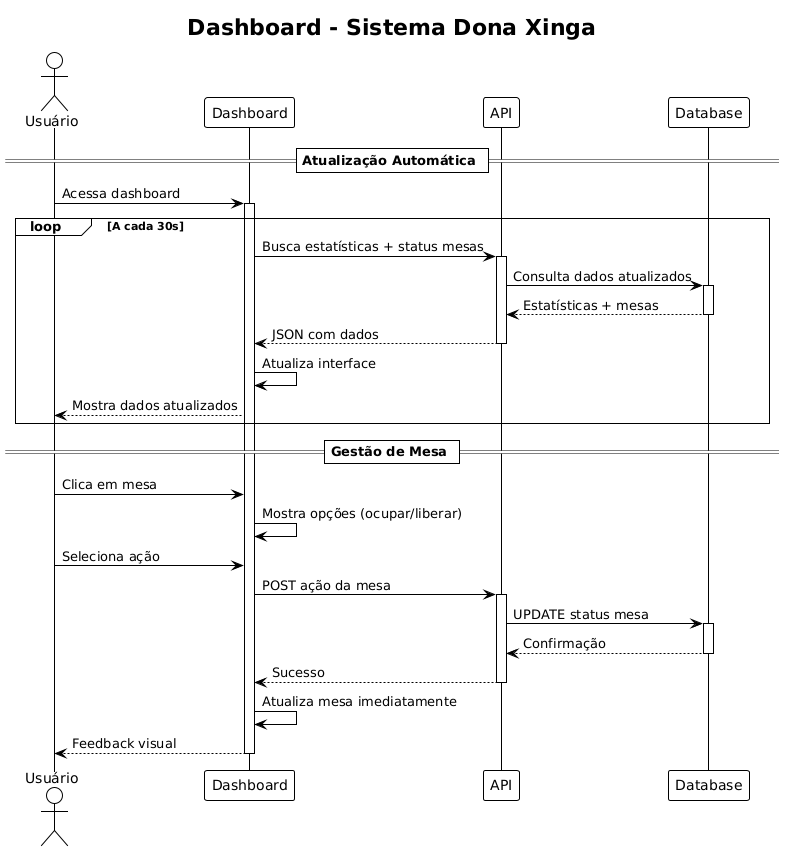
1. Diagrama de Sequência - Fazer Reserva



2. Login de Usuário (Autenticação)

****

**3.** **Diagrama de Sequência - Dashboard em Tempo Real**

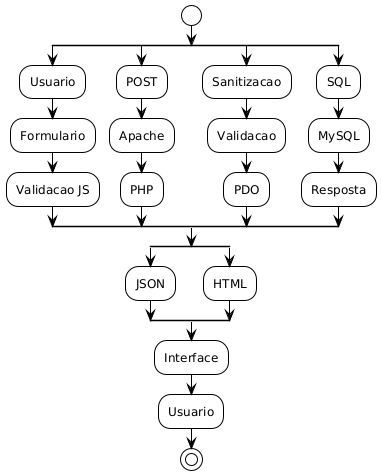
****

**Tabela de Resumo - Diagramas de Sequência**

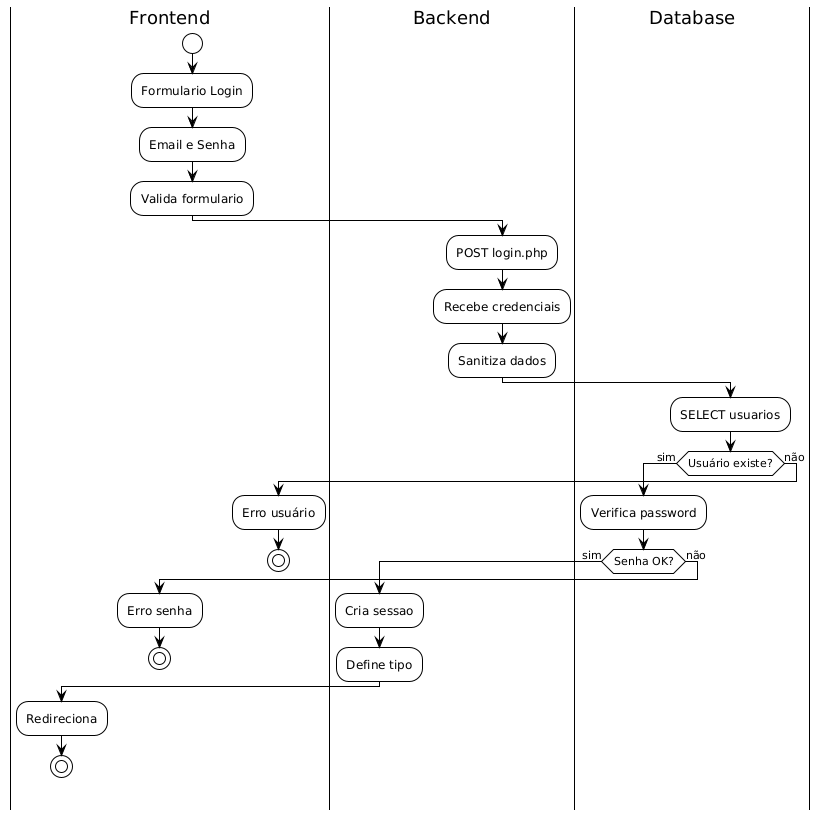
|  |  |  |  |
| --- | --- | --- | --- |
| Aspecto | Fazer Reserva | Login de Usuário | Dashboard |
| Objetivo | Criar nova reserva | Autenticar usuário | Monitorar e gerenciar mesas |
| Usuário | Cliente registado | Cliente registrado | Recepcinista/Gerente |
| Tipo | Síncrono | Síncrono | Assíncrono (tempo real) |
| Operações DB | SELECT + INSERT | SELECT + verificação hash | SELECT + UPDATE |
| Validações | Disponibilidade, dados | Credenciais | Permissões |
| Frequência | Sob demanda | Início de sessão | Contínuo (30s) |
| Complexidade | Média | Baixa | Alta |
| Resultado | Confirmação reserva | Redireccionamento | Interface atualizada |

**Fluxogramas de Processos**

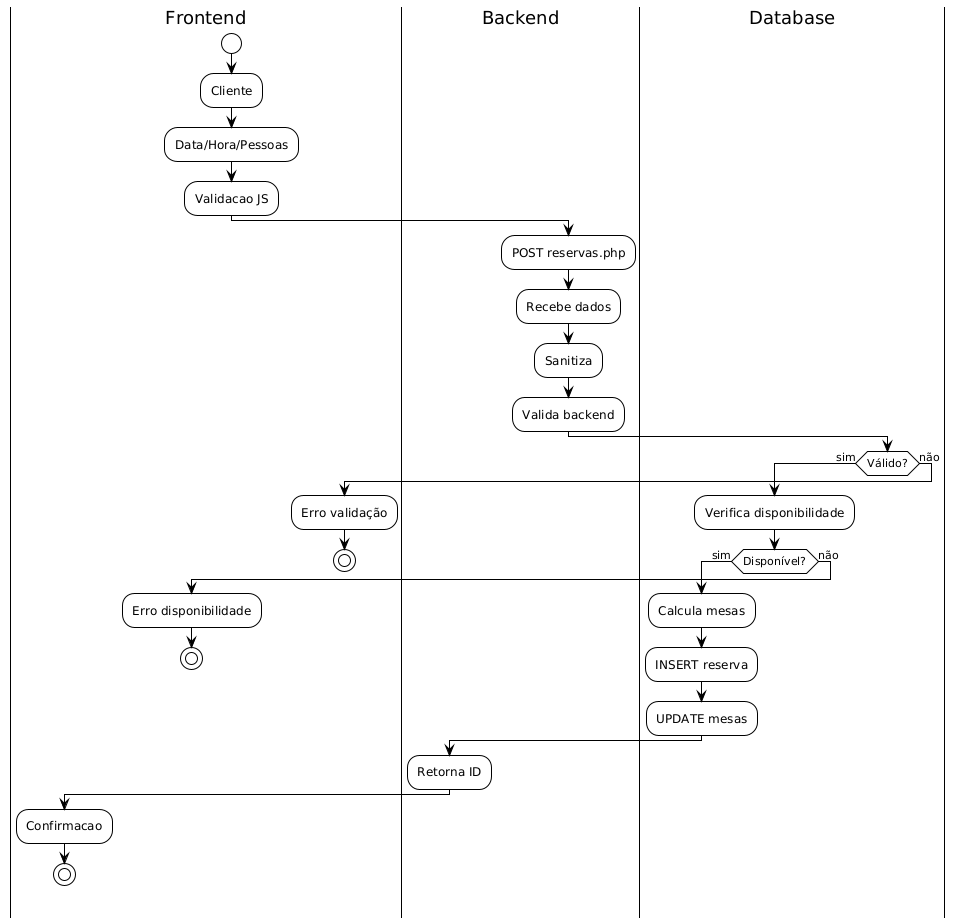
**Fluxo de Geral**

****

**Fluxo de Login**

****

**Fluxo de reserva**

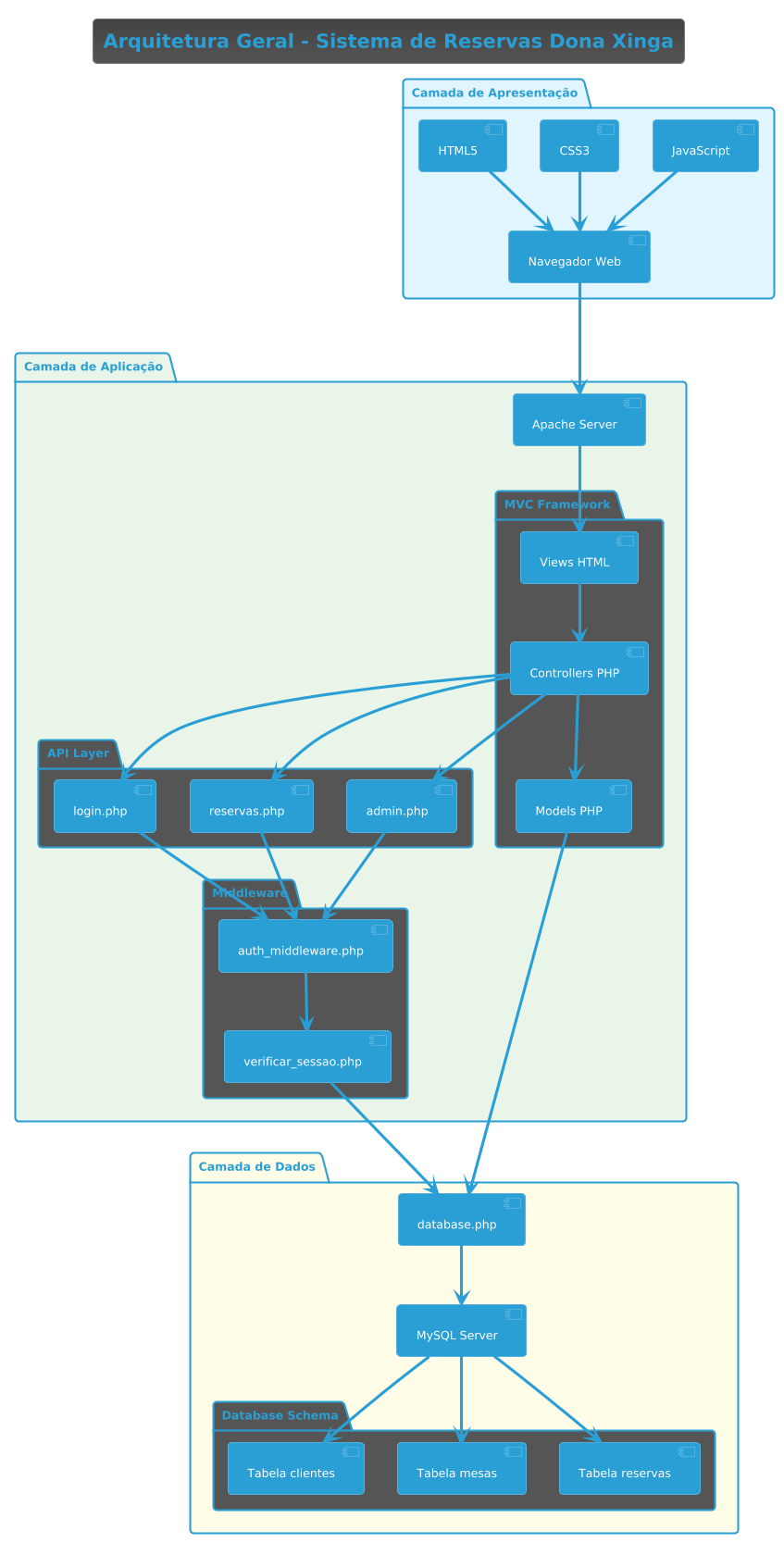
****

**Resumo dos Fluxos Principais**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fluxo | Entrada | Processamento | Database | Saída | Complexidade |
| Geral | Formulário HTML | Sanitização + Validação | PDO + SQL | JSON/HTML | Média |
| Login | Email + Senha | Verificação credenciais | SELECT usuarios | Redireccionamento | Baixa |
| Reserva | Data + Hora + Pessoas | Validação + Disponibilidade | INSERT + UPDATE | Confirmação | Alta |

**Arquitetura do sistema**

**Diagrama de arquitetura**

****

**Cabimento Académico**

Este projeto, proposto no âmbito da unidade curricular de Programação Web, lecionada pelo Eng. Manuel Bessa, representa não só uma avaliação académica, mas também uma oportunidade de desenvolvimento pessoal e profissional. Serve para testar os nossos conhecimentos e competências enquanto futuros engenheiros informáticos, num dos possíveis ramos de especialização após a conclusão do curso.