**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 10 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.

**Objetivo**

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.

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

**Porquê um sistema online?**

Poder-se-ia considerar, à partida, uma solução mais simples — como, por exemplo, a criação de uma linha telefónica exclusiva para a gestão de reservas. No entanto, esse método apresenta várias limitações: a impossibilidade de lidar com chamadas simultâneas, a sobrecarga da equipa de atendimento, o risco de erros manuais e a inexistência 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.

**Tecnologias Utilizadas**

O desenvolvimento será realizado na IDE Visual Studio Code. O front-end (interface do utilizador) será construído com HTML, CSS e JavaScript, enquanto o backend (lógica de funcionamento) será desenvolvido em PHP, com armazenamento de dados em MySQL.

Adicionalmente, recorrer-se-á a ferramentas de inteligência artificial como ChatGPT, Grok e Creati, que servirão de apoio ao processo de desenvolvimento — nomeadamente na geração de código, esclarecimento de dúvidas e criação de conteúdos gráficos e textuais.

**Metodologia**

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

* Modelo Waterfall (em cascata): em que as etapas do projeto são executadas de forma sequencial, facilitando o planeamento e a documentação;
* Elementos do modelo Agile: nomeadamente o desenvolvimento em *sprints* (fases incrementais), como, por exemplo, a implementação inicial do sistema de login antes da funcionalidade de reservas, permitindo uma maior flexibilidade e capacidade de adaptação a imprevistos.

**Motivação**

Este projeto, além de corresponder a uma exigência académica no âmbito de uma unidade curricular, 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.

**Requisitos do Projeto**

A criação de um website com estas características exige uma visão detalhada das funcionalidades que o sistema será — e não será — capaz de realizar, algo apenas mencionado brevemente na introdução.

**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 de reservas.
* Reservas: Reservar mesas com data, hora (intervalos de 30 minutos, ex.: 19:00) 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, Concluído, Cancelado, Expirado; Expirado em vermelho).
* Alerta de Saída: Alerta “Tem certeza que quer sair sem salvar?” ao fechar aba com formulário incompleto.
* Área do Rececionista
* Visualização de Reservas: Filtros por nome, telemóvel ou data/hora, exibindo até 5 resultados (Nome, Telemóvel, Data, Hora, Nº de Pessoas).
* Gestão de Mesas: Atualizar estado de mesas (Livre → Ocupada) com botão “Confirmar Ocupação”.
* 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 30 minutos.
* Clientes sem Reserva: Registrar ocupação manual de mesas (mínimo 1 pessoa)
* Área do Administrador
* **Gestão de Reservas**: Cancelar/editar reservas pra corrigir *\*overbooking\**, com pop-up: “Confirmar edição?”
* **Alertas de Falhas**: Dashboard com até 5 alertas (ex.: “Mesa 3 reservada duas vezes”), com links pra edição.
* **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, salva em `relatorios\_diarios`.

**Comportamento de Erros**

O sistema reage a cenários de erro com mensagens claras pra evitar travamentos e orientar o usuário, usando pop-ups (*\*SweetAlert2\**) 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), um pop-up exibe: “Data inválida. Escolha uma data entre hoje e os próximos 7 dias.”
* **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 10 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.”

**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 10 mesas, cada uma com capacidade fixa de ~4 pessoas (máximo total de ~60 pessoas). Reservas devem respeitar a capacidade (ex.: grupo de 5 não pode usar uma mesa de 4).
* **Alocação de mesas**: O sistema aloca uma mesa com capacidade igual ou ligeiramente maior que o número de pessoas (ex.: grupo de 3 usa uma mesa de 4, não de 6, se disponível). A alocação usa uma query SQL pra verificar disponibilidade (ex.: `SELECT id FROM mesas WHERE capacidade >= ? AND id NOT IN (SELECT mesa\_id FROM reservas WHERE data = ? AND hora = ?)`).
* **Gestão de atrasos**: Reservas com 30 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 um pop-up: “Período de edição/cancelamento expirado. Entre em contato com o restaurante.”
* **Reservas simultâneas**: O sistema impede *\*overbooking\** verificando a disponibilidade de mesas antes de confirmar uma reserva, usando uma query SQL pra contar mesas ocupadas no horário.

**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. |

**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) e XSS (sanitização de inputs).
* Senhas hasheadas com password\_hash().
* Sessões expiram após 30 minutos de inatividade.
* 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

* 5-10 usuários simultâneos (> 95% de sucesso).
* 1000 reservas/mês com índices no MySQL.

**Limitações do Sistema**

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

* Capacidade para até 10 mesas.
* Ausência de sistema de pagamento funcional.
* Número limitado de utilizadores em simultâneo (entre 5 e 10).
* Sem aplicação móvel ou funcionalidades de entrega.
* Formato optimizador apenas para computadores e telemóveis.

Estas limitações refletem a natureza preliminar e de pequena escala 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.

**Esclarecimento das Tecnologias Utilizadas**

O desenvolvimento será realizado na IDE Visual Studio Code.

O front-end será construído com:

* HTML (Linguagem de Marcação de Hipertexto): estrutura o conteúdo da página, incluindo texto, imagens e hiperligações.
* CSS (Folhas de Estilo em Cascata): define o design visual, como cores, estilos e organização da página, tornando a interface mais apelativa.
* JavaScript: linguagem de programação responsável por adicionar interação e dinamismo à página, como atualizações de conteúdo em tempo real.

O backend será implementado em PHP, uma linguagem de script executada no servidor, utilizada para processar pedidos, aceder a bases de dados e gerar conteúdo dinâmico em HTML.

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.

Além disso, recorrer-se-á a ferramentas de inteligência artificial como ChatGPT, Grok e Creati para apoiar o processo de desenvolvimento:

* ChatGPT: para esclarecimento de dúvidas, correção gramatical e apoio na criação de conteúdos textuais e gráficos.
* Grok: para análise e explicação detalhada de código.
* Creati: para auxílio na criação de páginas web e estruturas de código com base nas tecnologias utilizadas.

**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.