

## Projeto Final da UC

### A. Objetivo

O objetivo deste projeto é desenvolver uma **plataforma web para gestão e marcação de serviços numa oficina automóvel**, integrando *Frontend* e *Backend*, com persistência de dados. A plataforma permitirá que múltiplas oficinas configurem os seus serviços (ex.: revisão, troca de óleo, diagnóstico), colaboradores técnicos, horários de atendimento e vagas. Os clientes poderão agendar marcações para os seus veículos, consultar histórico e gerir marcações.

Esta aplicação deve ser feita em **React** e deve comunicar com um servidor desenvolvido em **ExpressJS**. O servidor deverá disponibilizar os **endpoints** necessários para a aplicação **Frontend**. Os dados devem ser guardados numa base de dados em **MongoDB**. A aplicação deve ser alojada num servidor web, para que possa ser acedida pelos docentes para posterior avaliação.

O projeto será realizado em grupos de três alunos, devendo a entrega incluir o código-fonte e um relatório detalhado.

### B. Descrição

A plataforma consiste num sistema multi-oficina onde cada oficina pode gerir a sua oferta de serviços automóveis, configurar horários, mecânicos disponíveis e número de vagas por turno. Os clientes registam-se na plataforma, associam um ou vários veículos e podem agendar serviços específicos.

A plataforma deve incluir **diferentes perfis de utilizador**, com permissões distintas:

- **Admin Oficina** – Gere a oficina, serviços, mecânicos, turnos e configurações.
- **Mecânico/Staff Oficina** – Gere vagas, visualiza marcações do seu dia e atualiza estados

das intervenções.

- **Cliente** – Regista veículos, consulta serviços e agenda marcações num calendário interativo.

Também devem existir regras de negócio como limites de marcações, regras de cancelamento e tempos mínimos entre marcação e início do serviço.

O foco do projeto é criar uma **aplicação funcional, intuitiva e completa**, cobrindo:

- Frontend com boa experiência de utilizador
- Backend com API organizada e segura
- Gestão de dados consistente
- Autenticação e autorização
- Experiência de marcação simples para o cliente

Funcionalidades extra são bem-vindas e valorizadas.

## C. Funcionalidades Principais

### 1. Oficinas e Serviços

- **A plataforma deve suportar múltiplas oficinas, cada uma com:**
  - Nome, localização, contacto
  - Serviços que oferece (e.g., revisão, troca de travões, diagnóstico)
  - Mecânicos associados
  - Turnos/horários de atendimento
  - Número de vagas por turno
- **Cada oficina define os seus próprios serviços com:**
  - Tempo estimado
  - Preço
  - Tipo de serviço
  - Descrição pública (visível aos clientes)

- Descrição privada (visível apenas ao staff)

## 2. Configuração de Serviços

- Para cada serviço devem ser configuráveis:
  - Duração (minutos)
  - Vagas por turno
  - Mecânicos que podem executar este serviço
  - Regras de agendamento (ex.: antecedência mínima)
  - Descrição pública e privada

## 3. Gestão de Utilizadores

- A plataforma terá três tipos de utilizadores:
  - **Admin Oficina:** Permissões administrativas para configurar a oficina.
    1. Gerir a oficina
    2. Criar/editar/apagar serviços
    3. Criar turnos e gerir agendas
    4. Gerir mecânicos e permissões
  - **Mecânico / Staff:**
    1. Visualizar agenda diária/semana
    2. Gerir o estado das marcações (p.e. “Em curso”, “Concluída”)
    3. Atualizar vagas quando necessário
  - **Cliente:**
    1. Gerir o seu perfil
    2. Associar um ou vários veículos (marca, modelo, matrícula)
    3. Ver serviços disponíveis por oficina
    4. Agendar marcações num calendário interativo
    5. Cancelar ou reagendar (consoante regras)

## 4. Marcações e Calendário:

5. Os clientes podem:

- Escolher oficina
- Selecionar um veículo
- Escolher serviço
- Ver turnos disponíveis
- Agendar em calendário
- Gerir marcações ativas

6. A plataforma deve:

- Verificar vagas
- Validar regras (ex.: marcação até X horas antes)
- Impedir conflitos de horários para mecânicos

## D. Funcionalidades Bónus

- Experiência de Utilizador (UX/UI) aprimorada
  - Interfaces visuais para configuração de turnos, agendas e mecânicos.
- Login com Google ou outro serviço externo
- Sistema de notificações (email ou UI):
  - Confirmação de marcação
  - Lembrete antes da intervenção
  - Notificação de estado do serviço
- Histórico detalhado do veículo:
  - Serviços anteriores
  - Recomendações automáticas (opcional)
- Gestão de faturas / comprovativos (simulado)

## E. Requisitos e Funcionalidades Detalhadas

### 1. Registo e Login

- Registo com nome, email, password e tipo de utilizador
- Possibilidade de login externo (opcional)
- Associação do cliente aos seus veículos
- Associação de mecânicos a oficinas

### 2. Gestão de Oficinas e Serviços

- Admin Oficina:
  - Criar, editar e remover serviços
  - Criar e editar turnos e horários
  - Gerir staff
  - Definir vagas e disponibilidade
- Staff Oficina:
  - Gerir vagas
  - Consultar agenda
  - Atualizar estado das intervenções

### 3. Gestão de Veículos dos Clientes

- Cada cliente pode ter 1 ou mais veículos
- Campos: marca, modelo, matrícula, ano

### 4. Agendamento

- Cliente escolhe oficina → serviço → veículo
- Seleciona data/hora num calendário

- Sistema valida vagas, tempos e conflitos
- Marcações incluídas na agenda do mecânico atribuível

## **5. Visualização e Gestão de Marcações**

- Cliente:
  - Calendário (semanal/mensal)
  - Histórico
  - Cancelamento/reagendamento conforme regras
- Oficina:
  - Agenda diária/semanal por mecânico
  - Listagem de marcações
  - Atualização de estados

## **F. Estrutura de Dados Sugerida**

- Oficinas
  - Nome, localização, contacto
  - Serviços
  - Staff/mecânicos
  - Turnos e vagas
- Serviços
  - Nome tipo
  - Preço
  - Duração
  - Mecânicos autorizados
  - Descrição pública/privada
- Mecânicos
  - Nome, email
  - Oficina

- Especialidades (opcional)
- Veículos
  - Marca, modelo, matrícula, ano
  - Cliente associado
- Turnos
  - Data e hora
  - Vagas
  - Serviço associado
  - Mecânico (ou lista)
- Utilizadores (todos)
  - Nome, email, role
  - Oficina associada (se staff)

## G. Avaliação

- O projeto será avaliado com base nos seguintes critérios:
- Funcionalidade e Completude:
  - Cumprimento dos requisitos e funcionalidades definidas.
  - Implementação das funcionalidades bónus (se aplicável).
- Qualidade da Interface e Experiência do Utilizador (UX/UI):
  - Interface intuitiva e atrativa.
  - Foco na experiência do utilizador, especialmente para a visualização e gestão de marcações.
- Qualidade do Código:
  - Organização, modularidade e clareza do código.
  - Boas práticas de desenvolvimento em Node.js, Express e React.
- Complexidade e Originalidade:
  - Funcionalidades adicionais ou melhorias além do mínimo exigido.
- Documentação e Entrega:
  - Incluir documentação para configuração e uso da plataforma.
  - Relatório que explique a estrutura da aplicação, funcionalidades implementadas, limitações e melhorias possíveis.



## H. Grupos

- O projeto deve ser realizado em grupos de 3 alunos. Cada grupo deve apresentar a sua plataforma e defender o seu trabalho.
- A avaliação será individual, considerando a contribuição de cada membro para o trabalho final.

## I. Entrega do Trabalho

### Entrega de Código:

- O código deve ser organizado em pastas (frontend, backend) e submetido num único ficheiro compactado (ZIP/RAR).
- O backend deve estar configurado para uma base de dados MongoDB e deve ser possível testá-lo localmente.
- O ficheiro será submetido obrigatoriamente na plataforma Moodle, nas duas unidades curriculares.

### Relatório:

- **Introdução:** Descrição geral da plataforma e objetivos.
- **Arquitetura:** Explicação da estrutura do código (pasta de rotas, controladores, etc.).
- **Implementação:** Explicação das principais funcionalidades e APIs desenvolvidas.
- **Conclusão:** Limitações, melhorias e funcionalidades que poderiam ser incluídas no futuro.

### Data de Entrega:

- A entrega deve ser feita até a véspera do exame da Época Normal de **PSW** (para a época normal) e na véspera do exame da Época de Recurso de **PSW** (para a época de recurso)
- A Defesa dos trabalhos decorrerá no dia da prova escrita de **PSW**.