**UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL**
**FACULDADE DE COMPUTAÇÃO**
**CONSTRUÇÃO DE SOFTWARE - 2021/2**
**PROFESSOR MÁRCIO SILVA**

# Introdução

O objetivo principal do trabalho é construir um sistema com Frontend e Backend escrito com a Linguagem Javascript. Cada Release/Sprint (versão do sistema) será definido pelo Professor. A Linguagem TypeScript também pode ser utilizada se o aluno já domina o Javascript e deseja adquirir novos conhecimentos. Porém, durante as aulas estudaremos apenas o Javascript.

# Descrição do Trabalho

O sistema consiste em uma aplicação para gestão de uma imobiliária. É muito importante que a aplicação atenda os requisitos mínimos listados a seguir.

Este trabalho pode ser desenvolvido com até 3(três) alunos. **Após a segunda entrega, nenhum grupo poderá receber novos membros. Ou seja, grupos com 1 ou 2 pessoas ainda poderão receber novos membros caso seja do interesse dos envolvidos.**

# Requisitos do Sistema

Sistema de Imobiliária

1. Para a imobiliária é muito importante manter um cadastro dos imóveis que estão à venda e dos imóveis que estão disponíveis para

locação.

2. Para cada imóvel registrado é importante saber se o imóvel está disponível para venda ou locação ou se ele já foi vendido ou locado. Isso ajudará a filtrar a pesquisa por imóveis.

3. Para cada imóvel é importante armazenar um ou mais fotos. O armazenamento pode ser, por exemplo, o nome do arquivo da foto do imóvel. Também é importante armazenar a data da construção do imóvel.

4. Os imóveis da imobiliária podem pertencer a uma dentre quatro categorias: casa, apartamento, terreno ou sala comercial.

5. Para os imóveis da categoria casa, as seguintes informações devem ser armazenadas: quantidade de quartos, quantidade de suítes, quantidade de salas de estar, quantidade de salas de jantar, número de vagas na garagem, área (em metros quadrados), se possui armário embutido, descrição (algum detalhe a mais que se deseja informar sobre a casa).

6. Para os imóveis da categoria apartamento, as seguintes informações devem ser armazenadas: quantidade de quartos, quantidade de suítes, quantidade de salas de estar, quantidade de salas de jantar, número de vagas na garagem, área (em metros quadrados), se possui armário embutido, descrição (algum detalhe a mais que se deseja informar sobre o apartamento), andar, valor do condomínio, indicativo se o condomínio possui portaria 24hs.

7. Para os imóveis da categoria sala comercial, as seguintes informações devem ser armazenadas: área em metros quadrados, quantidade de banheiros, quantidade de cômodos.

8. Para os imóveis da categoria terreno, as seguintes informações

devem ser armazenadas: área (em metros quadrados), largura, comprimento, se o terreno possui aclive/declive.

9. Para cada imóvel disponível para locação ou venda é importante saber o seu endereço completo, sendo muito importante cadastrar o bairro no qual o imóvel está localizado, pois este campo ajudará a filtrar várias pesquisas feitas por cliente.

10. É também importante armazenar o valor de venda ou de aluguel de cada imóvel. Este é o valor sugerido pelo cliente.

11. Após o imóvel ser alugado/vendido é importante registrar o valor real de aluguel e venda, indicando quanto desse valor será destinado à imobiliária.
12. Cada imóvel disponível para venda ou locação deve estar associado a um cliente da imobiliária, nomeado cliente proprietário. Um imóvel pode pertencer a mais de um cliente proprietário e um cliente proprietário pode ter mais de um imóvel na imobiliária.

13. Os clientes proprietários da imobiliária devem ser cadastrados no sistema e as seguintes informações devem ser cadastradas: cpf, nome, endereço, telefones de contato, email, sexo, estado civil, profissão.

14. Cada imóvel vendido ou alugado deve estar associado a um cliente da imobiliária, nomeado cliente usuário.

15. Os clientes usuários da imobiliária devem ser cadastrados no sistema e as seguintes informações devem ser cadastradas: cpf, nome, endereço, telefones de contato, email, sexo, estado civil, profissão. Caso o cliente usuário vá alugar um imóvel também devem ser armazenados: fiador (pelo menos 1) e indicações (pelo menos 2).

16. É importante registrar a data que um imóvel foi colocado à venda ou para alugar. Também é importante registrar quando o imóvel foi alugado ou vendido.

17. Alguns imóveis, embora sejam anunciados para venda ou aluguel, acabam sendo retirados da imobiliária por decisão do cliente proprietário. É preciso armazenar no banco a informação de que o imóvel não foi alugado/vendido, mas não está mais disponível para locação/venda.

18. Para cada venda ou aluguel é necessário registrar o funcionário que fez a transação, bem como o valor da comissão que será destinado a esse funcionário. Um funcionário pode realizar mais de uma transação.

19. Para cada um dos funcionários da imobiliária é importante armazenar: nome, telefone, cpf, endereço, telefone contato, telefone celular, data ingresso imobiliária, cargo e salário, o qual é calculado a partir do seu salário base mais a porcentagem das comissões que ele recebe. Também é importante armazenar os dados para login no sistema, como usuário e senha. Para a senha é importante ter um campo que consiga armazenar pelo menos 128 caracteres. Você pode supor que a criptografia da senha será feita por software.

20. Os cargos disponíveis na imobiliária devem ser previamente cadastrados. Cada cargo possui um salário base.

21. Para cada transação (venda ou aluguel) é necessário realizar um registro no banco que contenha além dos dados do imóvel, do cliente e do funcionário, a data que foi realizada a transação, o número do contrato (que é um número sequencial que identifica cada um dos contratos da empresa) e a forma de pagamento.

22. As formas de pagamento disponíveis na empresa devem ser

previamente cadastradas.

23. Se um mesmo imóvel é colocado mais de uma vez para venda ou aluguel, somente a informação mais recente deve ser armazenada. As informações mais antigas a respeito do imóvel vão para uma tabela de histórico.

## Regras

- Não serão aceitos trabalhos atrasados. Se o grupo não entregar o trabalho no dia combinado, ele receberá nota zero.
- Em caso de projetos copiados de colegas, todos os envolvidos recebem nota zero. Lembre-se, é muito improvável que haja trabalhos totalmente iguais.
- O professor não ajudará os grupos na construção do trabalho. Porém, fornecerá uma aplicação exemplo (nicho e requisitos diferentes) que será desenvolvida durante as aulas.
- O professor poderá tirar dúvidas conceituais em horário de aula ou horário de atendimento (grupo do telegram, ava ou e-mail).
- O professor pode, a qualquer momento, solicitar uma apresentação dos grupos sobre o andamento dos trabalhos.
- O professor poderá questionar cada um dos integrantes do grupo no momento da apresentação.
- A nota dos integrantes não necessariamente será a mesma. Se durante a apresentação o professor detectar que algum integrante do grupo não tem domínio sobre o projeto, ele poderá receber uma nota menor que os demais integrantes.
- Cada integrante **deve possuir uma conta do github** para versionar o código do projeto. O professor fornecerá um repositório privado no github a cada grupo para desenvolver o projeto. Não crie *forks* deste projeto, caso seja necessário crie *branches* para organizar o desenvolvimento. (23:59hs GTM-04).

**RELEASES:**

**Sprint 1 (Entrega: 29/08/2021, 23:59hs GMT-04):**

Sprint 1 será necessário a entrega da estrutura do banco de dados da aplicação (sem dados) contendo:

- **Diagrama Físico** contendo todas as tabelas e relacionamentos do banco de dados relacional.
- **Arquivo ESTRUTURA.SQL** contendo a estrutura do banco de dados baseado no diagrama anterior.
- O banco de dados pode ser implementado no **MySQL (5.6+), PostgreSQL (9.1+), MariaDB (10.1+) ou MongoDB (4.0+).**
- No caso de desenvolvimento do banco dados NoSQL (MongoDB), o grupo deve entregar um JSON (ao invés de um arquivo .sql) que represente as *collections* do banco de dados bem como a estrutura interna de cada *collection*. O diagrama físico deve ser substituído por um diagrama de classe.

**Sprint 2 (Entrega: 09/09/2021, 23:59hs GMT-04):**

Cada grupo deve desenvolver todas as ***migrations*** para o banco de dados proposto na **Sprint 1**. A aula completa sobre migrations está disponível no ava.ufms.br e foi ministrada em **25 de Agosto de 2021.**

**Sprint 3 (Entrega: 19/09/2021, 23:59hs GMT-04):**

CRUD - (Criar, Listar, Atualizar e Deletar) de Pessoas (Funcionários, Clientes).

LOGIN e LOGOUT.

**Sprint 4 (Entrega: 30/09/2021, 23:59hs GMT-04):**

CRUD - (Criar, Listar, Atualizar e Deletar) de Imóveis.

**Sprint 5 (Entrega: 10/10/2021, 23:59hs GMT-04):**

CRUD - (Criar, Listar, Atualizar e Deletar) de Métodos/Formas de Pagamento, Cargos dos funcionários da Imobiliária e outros dados/metadados/tabelas que são necessários para realizar uma Transação de VENDA ou ALUGUEL.

**Sprint 6 (Entrega: 24/10/2021, 23:59hs GMT-04):**

Implementar a **TRANSAÇÃO de venda e Aluguel de um imóvel.** Nesta etapa, o seu backend deve ser capaz de receber os dados necessários para realizar uma transação de aluguel ou venda de um imóvel. Cuide do histórico do imóvel, bem como comissões dos funcionários envolvidos no processo.

**Sprint 7 (Entrega: 14/11/2021, 23:59hs GMT-04):**

Implementar uma tela de login em ReactJS. A tela de login deve ficar alinhada ao centro. São necessários os campos de e-mail e senha, seguido de um botão “entrar”.

Validar se os campos estão preenchidos antes de enviar ao servidor. Se os campos obrigatórios estiverem preenchidos, envie as informações ao backend. Se o login ocorreu com sucesso, redirecione para a rota “/home” e salve o token Json Web Token no Browser. Caso contrário, emita um alerta na tela. Este alerta pode ser feito com o *alert* padrão do navegador ou utilizando outro feedback visual, como *Toasts*, *Modais* ou mensagens no corpo da página.

É preciso integrar o seu layout com algum Framework CSS.

Sugestões de Frameworks CSS integrados com ReactJS para ajudar no design:

- https://react.semantic-ui.com/

- https://mui.com/pt/

Depois de implementado, grave um vídeo de máximo 5 minutos mostrando tudo que foi desenvolvido. **O link para o vídeo deve estar no readme.md.**

**Sprint 8 (Entrega: 28/11/2021, 23:59hs GMT-04):**

Implementar a rota "/home" no frontend (ReactJS). Nesta página devem ser listados todos os imóveis disponíveis para aluguel e venda. Mostre informações como endereço, características dos imóveis, tipo e dados do proprietário (nome e dados de contato).

Adicionalmente, crie duas novas rotas "/home/aluguel" para mostrar imóveis disponíveis apenas para aluguel e "/home/venda" para mostrar apenas imóveis disponíveis para venda.

Grave um vídeo de máximo 5 minutos mostrando as funcionalidades implementadas nesta release. **O link para o vídeo deve estar no readme.md.**

**Sprint 9 (Entrega: 28/11/2021, 23:59hs GMT-04):**

Implementar a interface de transação (rota: "/transacao"). Nesta interface, você enviará ao backend as informações que ele espera desenvolvidos na Sprint 6. Basicamente precisamos saber qual o imóvel a ser alugado/vendido, quem é o comprador ou inquilino, bem como outras informações relacionadas à transação que você julgou necessário na Sprint 6.

Grave um vídeo de máximo 5 minutos mostrando as funcionalidades implementadas nesta release. **O link para o vídeo deve estar no readme.md.**

**Sprint 10 (Entrega: 28/11/2021, 23:59hs GMT-04):**

Implementar a interface (frontend) que lista todo o histórico de transações da imobiliária (rota "/historico"). Se for necessário, crie novas rotas no backend para prover os dados necessários.

Grave um vídeo de máximo 5 minutos mostrando as funcionalidades implementadas nesta release. **O link para o vídeo deve estar no readme.md.**

EXEMPLO DE MODELO FÍSICO: