# ParkIO

Guilherme Gabriel S. Pereira, Henrique P. F. Monteiro, José Maurício G. França, Lucas Ângelo O. M. Rocha e Victor B. G. Campos.

1. **Apresentação do problema**

O Condomínio do Conjunto Habitacional Santa Terezinha possui 54 blocos e 2 portarias, com isso, contabilizando 918 apartamentos. Além disso, não possui uma vaga de estacionamento exclusivo para cada apartamento, totaliza-se aproximadamente apenas 800 vagas que são de acesso aos moradores e aos visitantes do condomínio.

Diante disso, as portarias possuem dificuldade de comunicação e controle de entrada e saída dos veículos visitantes. Logo, o condomínio possui uma demanda de segurança envolvendo o controle da entrada e saída de veículos que não possuem tag (Identificador de condômino) de entrada.

1. ***Stakeholders***

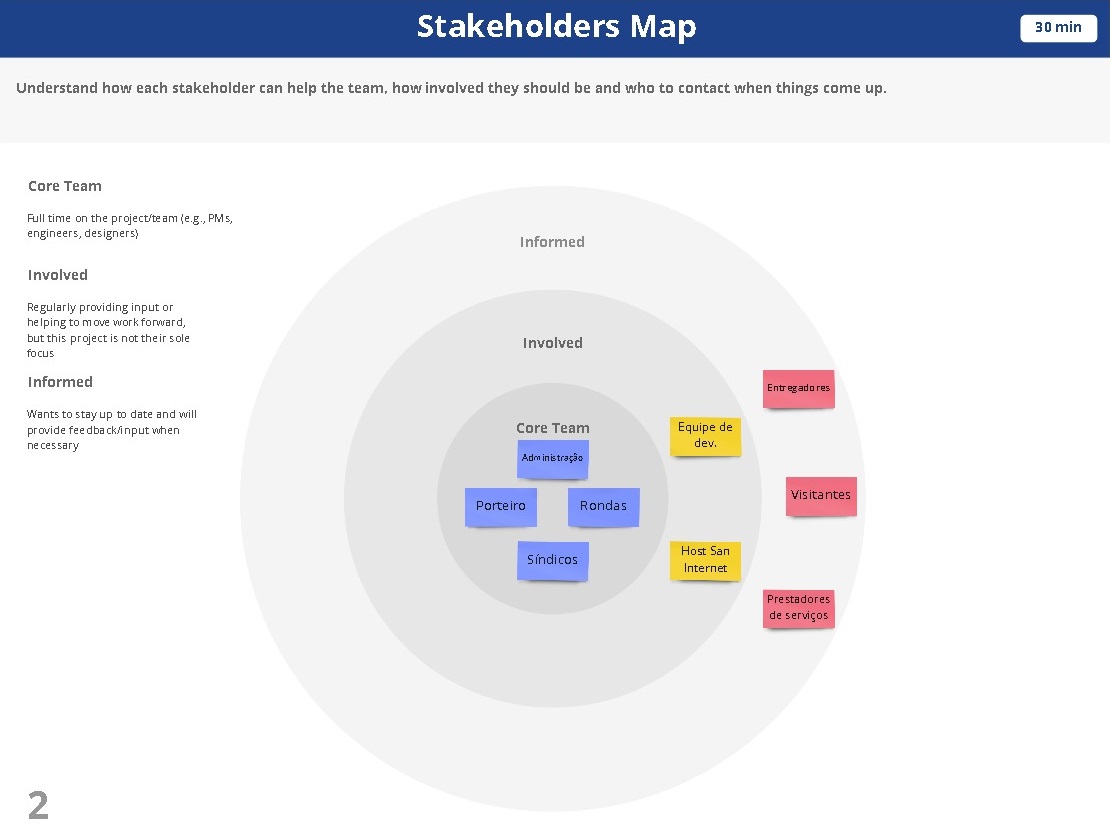


Imagem 1 – Mapa de Stakeholders ParkIO

Partes interessadas no projeto:

* Administração do condomínio: pode visualizar, cadastrar, alterar e remover veículos, além de poder criar usuários, visualizar os relatórios e alterar configurações do sistema.
* Síndicos(as) de cada bloco: podem visualizar somente os veículos de visitantes direcionados ao seu bloco, sem poder fazer alterações no registro.
* Porteiros(as): podem cadastrar a entrada e saída dos veículos para sua portaria que está no instante, e enviar alertas para administração.
* Rondas: podem visualizar e alterar dados de todos os veículos visitantes.
* Visitantes: pessoas que desejam entrar no condomínio, porém, não moram lá. (Exemplos: entregador de pizza, técnico de internet...).

Personas:

Imagem 2 – Persona de ‘Porteiro’

Imagem 3 – Persona de ‘Administradora’

Imagem 4 – Persona de ‘Síndico’

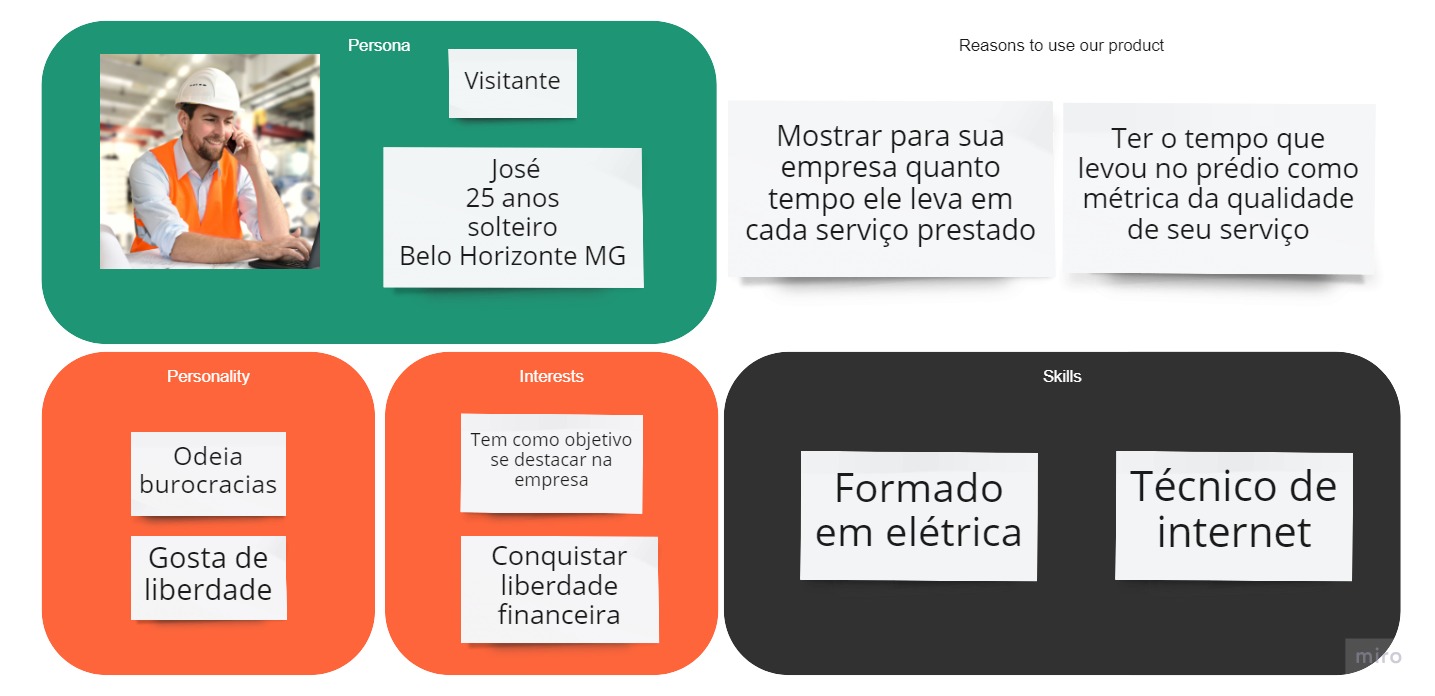
Imagem 5 – Persona de ‘Visitante’

Imagem 6 – Persona de ‘Ronda’

As imagens 2, 3, 4, 5 e 6 representam as personas dos principais usuários do sistema. Foram definidas, para que fosse possível notas as necessidades que fariam os usuários utilizarem o sistema, seus interesses e personalidades, para auxiliar na adaptação da interface do sistema.

1. **Proposta da solução**

Requisitos do produto:

|  |  |
| --- | --- |
| Nº | Descrição do Requisito Funcionais |
| 1 | O porteiro deve cadastrar os veículos no sistema. |
| 2 | O porteiro deve remover o veículo quando o visitante sair. |
| 3 | O administrador do sistema deve cadastrar tipos de visitantes. |
| 4 | O porteiro poderá aumentar o tempo de permanência do visitante. |
| 5 | O sistema deve emitir um alerta para um grupo no Telegram quando algum veículo ultrapassar o tempo máximo no condomínio. |
| 6 | O administrador do sistema poderá criar usuários com diferentes permissões. |
| 7 | O administrador do sistema poderá criar diferentes portarias dentro do sistema. |
| 8 | Os usuários de qualquer tipo devem ser capazes de fazer login. |
| 9 | Os usuários de qualquer tipo devem selecionar uma portaria para utilizar o sistema. |
| 10 | Os usuários devem poder consultar a lista de veículos. |
| 11 | A administração deverá possuir um relatório delimitado por um filtro de período de tempo, porteiro e/ou placa do veículo. |
| 12 | O porteiro deverá reportar um veículo visitante, caso ocorra algum problema. |
| 13 | O ronda poderá editar placa, modelo e cor de veículos que ainda estão dentro do condomínio. |
| 14 | O síndico poderá visualizar os veículos designados para o seu bloco. |

|  |  |
| --- | --- |
| Nº | Descrição dos Requisitos Não Funcionais |
| 1 | O sistema web deve ser responsiva para proporcionar o uso de todas as funcionalidades providas pelos requisitos funcionais em resoluções de 576px até 1080px. |
| 2 | O sistema deve processar requisições do usuário em no máximo 3 segundos. |
| 3 | O sistema deve ser aprovado nos testes unitários de requisições HTTP e autenticação. |
| 4 | O sistema deve possuir uma interface web que seja objetiva para o usuário, com no máximo três funcionalidades por página. |
| 5 | O sistema deve ser compatível com sistema operacional Linux, com o objetivo de proporcionar a disponibilidade em nuvem de pelo menos 98% do tempo de atividade (uptime). |
| 6 | O sistema deve ser dimensionado para suportar até 20 usuários conectados ao mesmo tempo. |
| 7 | O sistema deve garantir a segurança das senhas dos usuários, criptografando-as ao serem inseridas no banco de dados. |
| 8 | O sistema deve garantir a integridade do registro simultâneo de veículos por mais de um usuário ao mesmo tempo, por meio do padrão de Int com auto increment como chave primária na tabela do banco de dado que recebe os veículos visitantes cadastrados. |
| 9 | A aplicação deve estar disponível nos modos claro e escuro, possibilitando maior conforto para o usuário. |

1. **Projeto da solução**

Os wireframes foram desenvolvidos utilizando o programa Adobe XD. O mapa de stakeholder, personas e histórias de usuário foram feitos na plataforma Miro.

Como linguagem de marcação de hipertexto foi escolhido o HTML5 para construção estrutural do site, CSS3 para estilização, com auxílio do framework Bootstrap 5.0.

Para programação será utilizado JavaScript no frontend e no backend PHP com auxílio do framework Lumen, e o SGBD selecionado foi o MySQL por ser relacional e gratuito.

A IDE de desenvolvimento escolhida foi o Visual Studio Code.

Infraestrutura para repositório do código online foi definido o GitHub. Foram utilizadas as ‘Issues’ do GitHub para se dividir as tarefas de implementação.

A empresa de hospedagem para a aplicação foi escolhida a Saninternet.com, para hospedar a aplicação. A hospedagem utilizada será com o gerenciamento da ferramenta cPanel.

Integração contínua para deploy de versões do software foi feita utilizando o GitHub Actions para fazer o upload para a hospedagem do San Internet automaticamente.

Diante disso, as tecnologias definidas foram:

* Frontend: HTML, CSS, JavaScript e Bootstrap.
* Backend: PHP e Lumen.
* SGBD: MySQL.
* IDE: Visual Studio Code.
* Wireframes: Adobe XD.
* Nuvem de Hospedagem: San internet utilizando Apache.
  + Cpanel: como ferramenta de administração da hospedagem.
* Miro: para criação do mapa de stakeholder, personas e histórias de usuário.
* MySQL Workbench: para criar o DER e modelo lógico de SQL.
* GitHub: repositório de código online.
* Markdown e docx para documentação e textos no GitHub.
* GitHub Actions: para deploy automático de versões.

1. **Artefatos principais**

Modelagem:

* Diagrama de caso de uso:

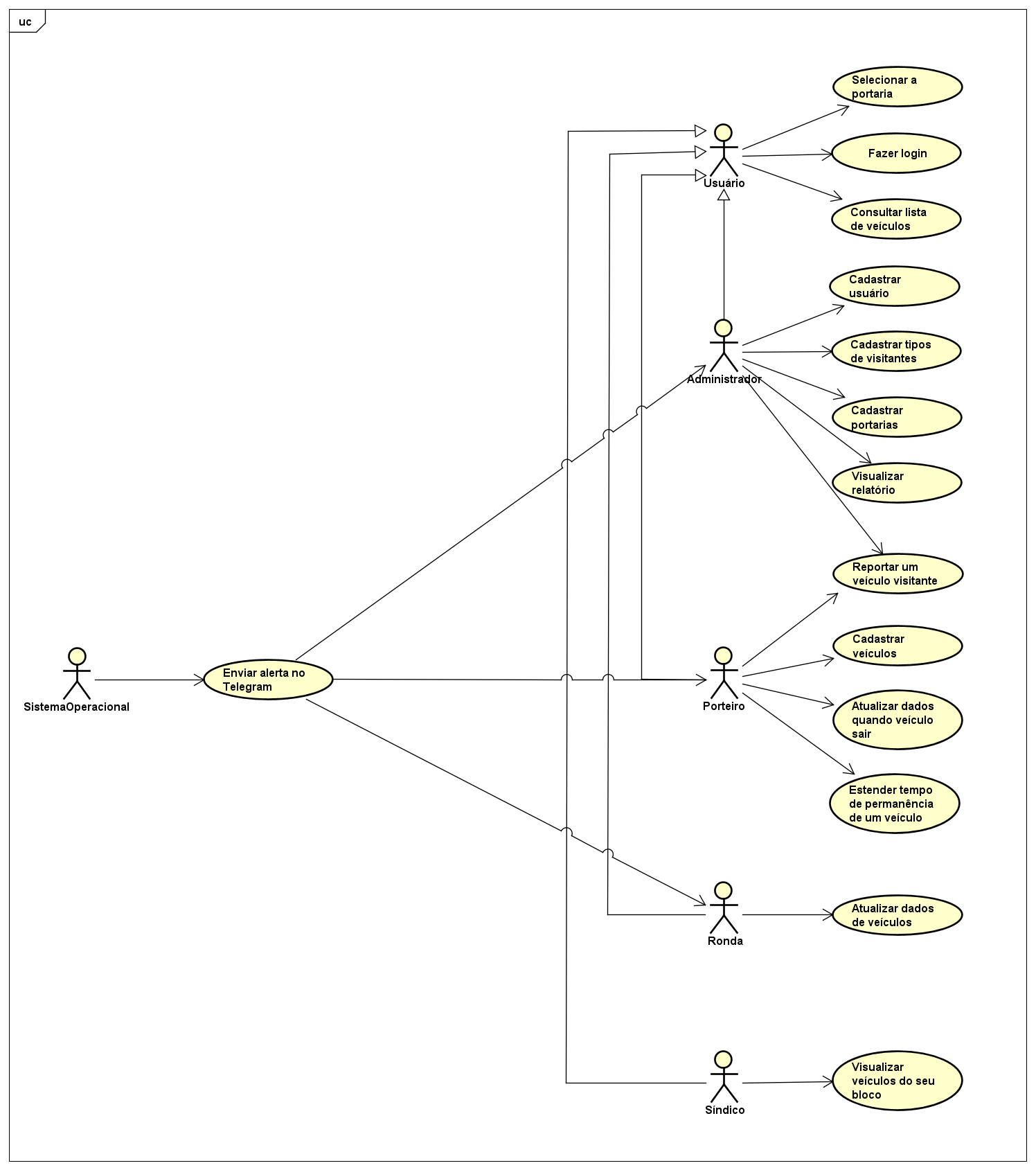


Imagem 7– Diagrama de Caso de Uso

A imagem 7 é o diagrama de caso de uso, que representa todos os atores e casos de uso. O ator ‘SistemaOperacional’ é o único ator que efetua o “Enviar alerta no Telegram”, para alertar os outros usuários “Administrador”, “Porteiro” e “Ronda” que há um veículo dentro do condomínio que ultrapassou o limite de tempo no estacionamento. O ator “Síndico” apenas pode “Visualizar veículos do seu bloco”, sem poder ver para qual apartamento o visitante do veículo foi. Já o ator “Ronda” pode “Atualizar dados de veículos” que eventualmente foram cadastrados errados ou não cadastrados pelo “Porteiro”. Além de “Cadastrar veículos”, o “Porteiro” pode “Atualizar dados quando veículo sair”, alterando o status dele para fora do condomínio, “Reportar um veículo visitante” caso ocorra algum problema ou “Estender tempo de permanência de um veículo” se o visitante necessite de ficar por mais tempo. O “Administrador” possui permissão para “Cadastrar usuário” de qualquer tipo, de menos “SistemaOperacional”, “Cadastrar tipos de visitantes” com o tempo limite para cada tipo, “Cadastrar portarias” do condomínio, e por fim, “Visualizar o Relatório” com os gráficos de desempenho e estatísticas do sistema. Ademais, todos os atores, de menos “SistemaOperacional”, podem “Selecionar a portaria” que irão utilizar, “Fazer login” no sistema e “Consultar a lista de veículos”.

* Diagrama de classes:

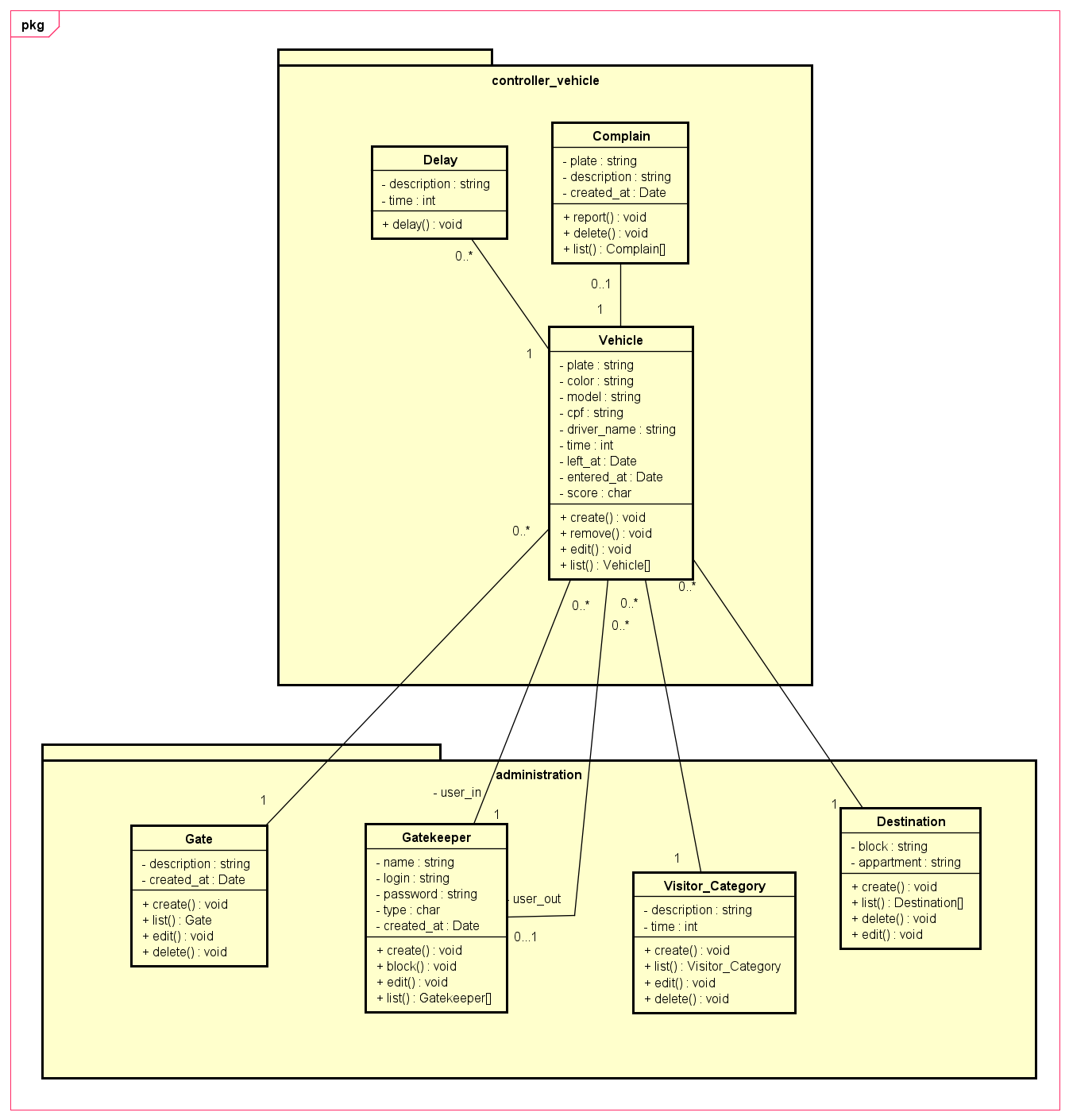


Imagem 8 – Diagrama de Caso de Classe

A imagem 8 representa a visão do sistema por meio do diagrama de classes, separado em dois pacotes: “controller\_vehcile” que possuí a classe “Delay” representando os adiamentos de tempos para cada veículo de um visitante que necessitou de ficar por mais tempo dentro do condomínio; também possui a classe “Complain” onde é guarda as denúncias efetuadas por porteiros para veículos que se comportaram mau dentro do condomínio; ademais há a classe "Vehicle” que guarda as informações do veículo, condutor e porteiro quem deixou entrar/sair que é a classe principal do diagrama possuindo relacionamento com todas as demais classes. O segundo pacote é chamado “Administration”: este pacote contém as portarias “Gates”, os porteiros “GateKeeper” , as categorias dos visitantes “Visitor\_category” e os destinos “Destination” possíveis de cada veículo.

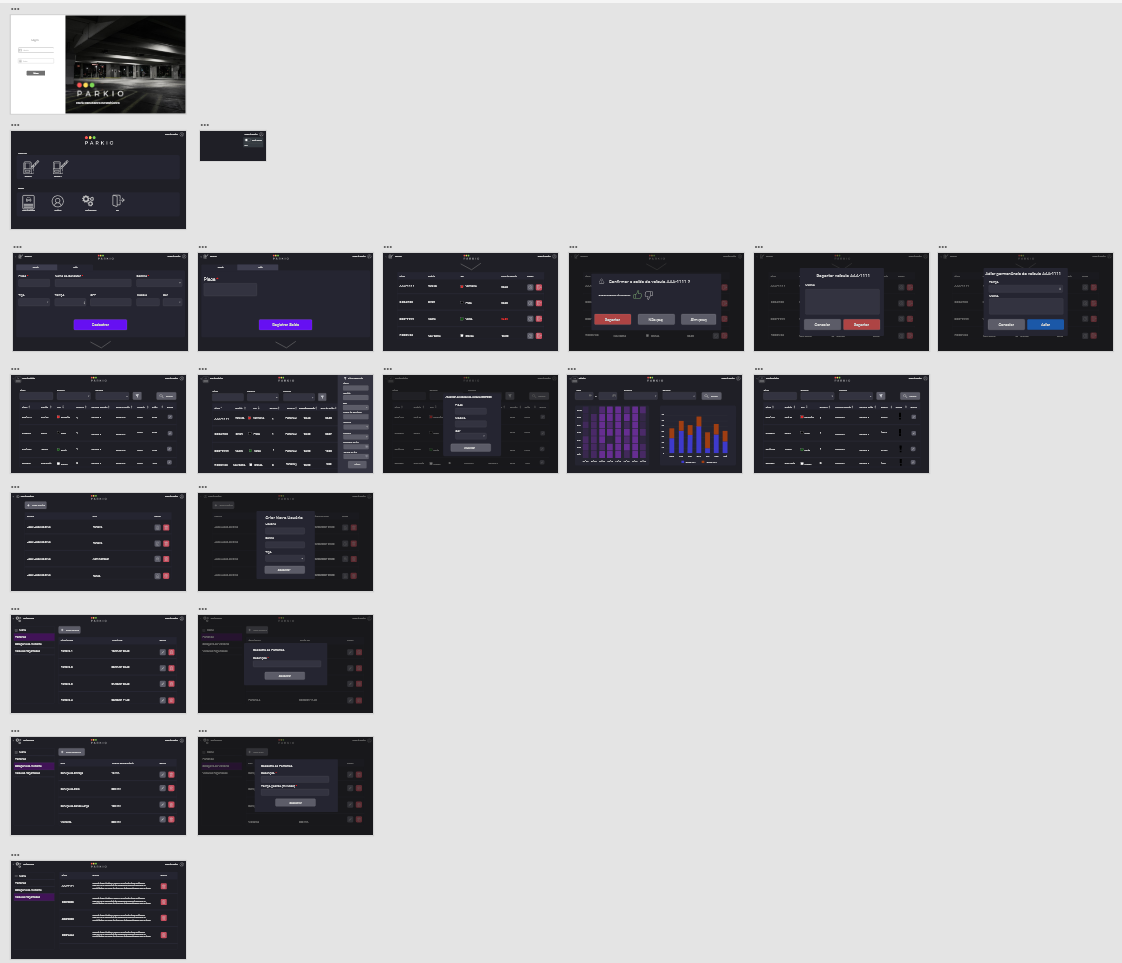


Imagem 9– Visão Geral do Protótipo

A imagem 11 representa a visão geral dos wireframes criados. No início eram somente os wireframes de cadastro de veículo e de listagem, porém durante o desenvolvimento do projeto tornou-se necessário a melhoria do design e a incrementação do restante das páginas que foram eventualmente desenvolvidas.

Software:

O ParkIO é um software que contém um conjunto de rotinas para auxiliar na organização da entrada e saída dos visitantes de um condomínio que adentraram em um veículo.

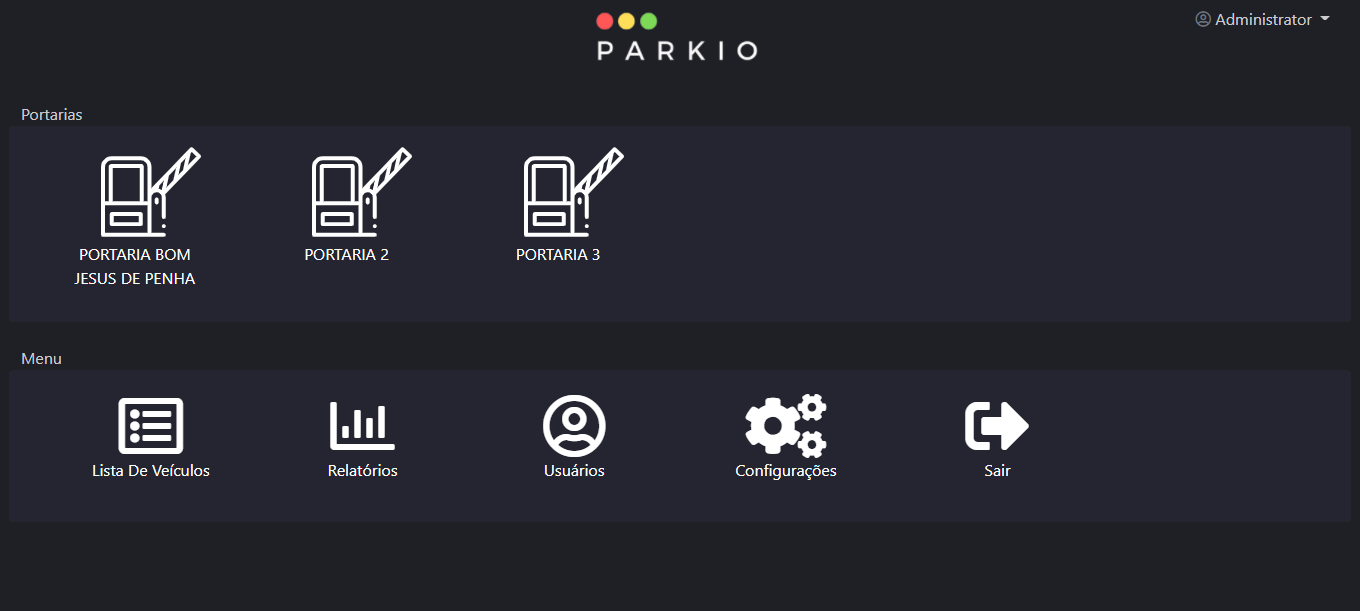


Imagem10 – Tela inicial na visão do administrador

Ao entrar dentro do sistema ele realiza se realiza uma separação dos tipos de usuário em que cada tipo possui alguns menus equivalentes à sua permissão, dos quais o usuário do tipo administrador tem acesso completo.



Imagem13 – Tela de cadastrar o visitante

A primeira rotina (conjunto de funcionalidades) e uma das principais, é a de cadastro e atualização do status do visitante, a qual em uma aba da tela são cadastradas informações sobre o visitante e seu veículo, como nome, placa, destino, tempo de estadia entre outros e na outra aba apresenta um campo de placa para o porteiro informar que o porteiro saiu do condomínio. Na parte inferior da tela, há uma lista com todos os veículos que ainda não saíram do condomínio e entraram pela mesma portaria. Nesta lista há possibilidade de adiar a estadia, exigindo uma justificativa do tempo adicional, que o morador deve informar ao porteiro. Ao informar que o visitante está saindo do condomínio, deve se fazer uma avaliação de como foi o comportamento do visitante, tendo as opções: bom e ruim, também há a possibilidade de reportar o comportamento do visitante com uma justificativa, para que caso o visitante volte futuramente apareça um aviso sobre o ocorrido anteriormente.

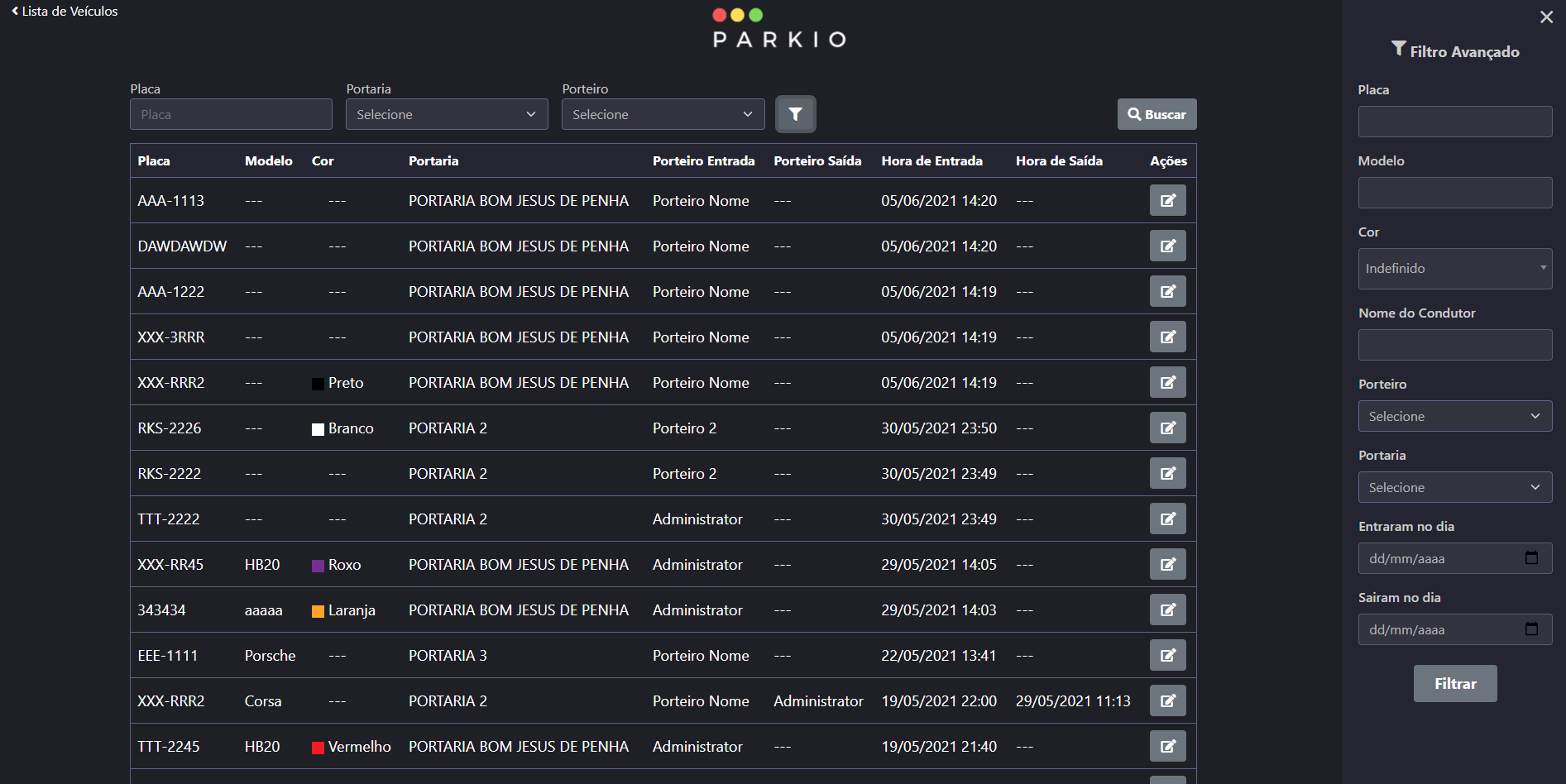


Imagem 14 – Tela de listagem dos visitantes

A segunda rotina é onde o usuário consegue visualizar informações sobre todos os veículos cadastrados que estão no condomínio e também os que já saíram do condomínio. Na parte superior possuí filtros de placa, portaria e porteiro, no menu ao lado, há um filtro avançado que permite a busca por modelo do veículo, nome de condutor, cor, data que entrou e data que saiu. Ademais, caso o usuário seja um administrador ou ronda é possível editar ou complementar dados do veículo como placa, modelo ou cor. Nesta lista, quando o usuário for um sindico, os veículos mostrados serão somente os que possuem um destino vinculado ao sindico, garantindo a segurança de não vazar dados desnecessários.



Imagem 15 – Tela de relatório dos visitantes

Já a terceira rotina, se trata da geração de relatórios que têm como objetivo, por meio de um filtro de data, portaria e porteiro, informar a quantidade de visitantes por hora através de um gráfico de calor. Também há a possibilidade de visualizar um gráfico de colunas, que expõe a quantidade de visitantes por porteiro nos respectivos dias da semana. Sendo assim, através da análise dos gráficos é possível realizar o gerenciamento do Condomínio, com o intuito de otimizar o serviço fornecido.

Uma rotina simples, está relacionada ao cadastro de usuários, no qual apenas os usuários administradores terão permissão de efetuar. Nela, é possível cadastrar usuários do tipo ‘administrador’, ‘porteiro’, ‘síndico’ e ‘ronda’, informando também um nome, login e senha para cada novo usuário. Também, é possível alterar a senha de usuários já cadastrados, por meio da lista de todos os usuários. Cada usuário possui suas permissões e limitações pré-definidas, como o porteiro pode cadastrar e remover veículos, mas, não pode visualizar veículo que já saíram; síndicos podem visualizar apenas quais veículos foram designados para seu bloco (sem pode ver o apartamento); a ronda pode alterar os dados apenas de veículos dentro do condomínio.

Adicionamos uma rotina de configurações com as opções de cadastro e edição e remoção de portarias, categorias de visitantes e destinos que são usados como categorizações na hora de registrar o visitante. Também possui uma lista de reclamações de mal comportamento dos visitantes, para que o admin verifique e caso precise, pode remover uma das reclamações.

1. **Conclusões**

O ParkIO foi uma aplicação web desenvolvida para possibilitar a administração do Condomínio do Conjunto Habitacional Santa Terezinha efetuar um controle da entrada e saída dos veículos de visitantes do condomínio.

Diante disso, foram traçados alguns objetivos específicos baseados nesse problema, dentre eles, permitir que o usuário Administrador cadastrar os porteiros, síndicos e rondas, para poderem utilizarem o sistema; os Porteiros podem efetuar o cadastro da entrada e saída dos veículos no sistema; os Síndicos podem apenas visualizar os visitantes designados para seus blocos; já os Rondas podem visualizar e alterar os dados dos veículos que já entraram.

Dessa forma, foi possível implementar todos os requisitos solicitados pelos clientes, possibilitando o condomínio fazer rodízio dos porteiros entre suas duas portarias, intercomunicando quais veículos de visitantes já estão no estacionamento e quais já saíram. Ademais, permitindo a administração verificar pelos gráficos, o fluxo da quantidade de veículos que entrou em cada horário dos dias de uma semana e quantos veículos foram cadastrados por cada porteiro.

Dando ênfase nos requisitos principais que foram os de " O porteiro deve cadastrar os veículos no sistema." e " Os usuários devem poder consultar a lista de veículos.". O sistema foi desenvolvido inicialmente por meio da criação de um Diagrama de Caso de Uso, para assim ser possível modelar o Diagrama de Classes do software, posteriormente, gerando os Casos de Usos Descritivos para os requisitos mais complexos. Com isso, foi criado o Diagrama de Entidade-Relacionamento, para tratar todas as entidades, atributos e relações, possibilitando a criação do banco de dados para a aplicação. Desse modo, o sistema foi instalado na plataforma San Internet, onde foram hospedados o banco de dados relacional MySQL e a aplicação criada em HTML, CSS e JavaScript no frontend com o framework Bootstrap, já no backend para conexão com o banco de dados foi utilizado o PHP com o framework Lumen, para ser possível a criação de uma API Restful.

Diante das informações supracitadas, conclui-se que todos os requisitos foram entregues. A aplicação encontra-se online e em uso pelos funcionários do condomínio.