DESENVOLVIMENTO DE SOFTWARE MULTIPLATAFORMA

PROJETO INTEGRADOR III

SOLUÇÃO FULL STACK

Disciplinas envolvidas no projeto integrador III

1. BANCO DE DADOS NÃO RELACIONAL
2. GESTÃO ÁGIL DE PROJETOS DE SOFTWARE
3. DESENVOLVIMENTO WEB III
4. INTERAÇÃO HUMANO COMPUTADOR
5. TÉCNICAS DE PROGRAMAÇÃO II

Fatec Votorantim

FATEC - VOTORANTIM D.S.M.

DESENVOLVIMENTO DE SOFTWARE MULTIPLATAFORMA
FATEC VOTORANTIM

CENTRO PAULA SOUZA

FATEC VOTORANTIM

Curso de tecnologia em Desenvolvimento de Software Multiplataforma

Curso Superior de Desenvolvimento de Software Multiplataforma

Breno Jose Da Silva, 3011392413025

Gabriel Ribeiro Correa, 3011392413032

Herivelton Henrique Gonçalves, 3011392413011

Wendel Augusto Lopes Vasco, 3011392413035

Projeto Interdisciplinar III

Gestão Ágil de Projetos de Software

FaculRide – WI (WORLD INNOVATION)

Orientadores

Prof.º Esp. Cláudio Corredato

Profª Maria Janaína da Silva Ferreira

Prof.º Ricardo Roberto Leme

Prof.º Tiago Vanderlei de Arruda

Votorantim

Junho, 2025

Resumo

O FaculRide surge como uma inovadora plataforma de caronas destinada à comunidade acadêmica, com o nobre objetivo de mitigar a emissão de carbono e facilitar o transporte diário de alunos, professores e funcionários. Este projeto não é apenas um site; é uma iniciativa sustentável que reflete o compromisso ambiental da instituição.

Através do FaculRide, propomos um sistema de compartilhamento de viagens que não só alivia o estresse do trânsito e reduz custos operacionais, mas também promove a interação social e a colaboração dentro do ambiente da faculdade. Com funcionalidades pensadas para a segurança e conveniência dos usuários, como cadastro verificado e um sistema de agendamento flexível, o FaculRide está preparado para se tornar parte essencial da rotina da faculdade.

O FaculRide apresenta-se como uma plataforma de caronas inovadora, com foco na sustentabilidade, interação social, segurança e na melhoria do transporte diário para a comunidade acadêmica.

SUMÁRIO

1 INTRODUÇÃO À GESTÃO ÁGIL E METODOLOGIA SCRUM 5
2 METODOLOGIA ÁGIL NA PRÁTICA 5
3. ENTREGAS DAS SPRINTS 5
4. Apresentação Final 6
5. Backlog do Produto & User Stories 6
6. Protótipo 7
7. Documentação do Projeto 8
8. Levantamento de Requisitos 8
9. Requisitos Funcionais 8
10. Diagrama de Caso de Uso 9
11. Requisitos Não Funcionais 9
12. Projeto do Software 10
13. Tecnologias Utilizadas 10
14. Integração e Papéis da Equipe 10

1 INTRODUÇÃO À GESTÃO ÁGIL E METODOLOGIA SCRUM

O desenvolvimento do FaculRide proporcionou a vivência prática dos conceitos e práticas da Gestão Ágil de Projetos de Software, com foco total na metodologia Scrum. O Scrum foi escolhido por sua abordagem interativa e incremental, permitindo maior flexibilidade, adaptação rápida a mudanças e entrega contínua de valor ao cliente. O projeto foi dividido em sprints curtos, cada um com metas claras, backlog priorizado e entregáveis definidos.

2 METODOLOGIA ÁGIL NA PRÁTICA

O projeto foi dividido em 5 sprints, cada uma com metas e entregáveis específicos:

Sprint 1: Estruturação do sistema, cadastro e autenticação de usuários, início da acessibilidade no front-end, primeiros testes.

Sprint 2: Melhorias visuais e funcionais, integração com Google Maps, refinamento da interface, segundo ciclo de testes.

Sprint 3: Integração dos fluxos de dados entre os serviços Angular, back-end e o banco de dados PostgreSQL, testes de integração, correção de bugs.

Sprint 4: Testes aprofundados de acessibilidade e usabilidade, ajustes finais de desempenho.

Sprint 5: Documentação completa, manuais, instruções de uso, detalhamento técnico e publicação do sistema na plataforma Render.

Cada sprint foi documentada com apresentações detalhadas, incluindo metas, backlog, user stories, critérios de aceitação e resultados alcançados.

3. ENTREGAS DAS SPRINTS

O projeto foi dividido em 5 sprints, cada uma com metas e entregáveis específicos:

Sprint 1: Estruturação do sistema, cadastro e autenticação de usuários, início da acessibilidade no front-end, primeiros testes.

Sprint 2: Melhorias visuais e funcionais, integração com Google Maps, refinamento da interface, segundo ciclo de testes.

Sprint 3: Integração dos fluxos de dados entre os serviços Angular, back-end e o banco de dados PostgreSQL, testes de integração, correção de bugs.

Sprint 4: Testes aprofundados de acessibilidade e usabilidade, ajustes finais de desempenho.

Sprint 5: Documentação completa, manuais, instruções de uso, detalhamento técnico e publicação do sistema na plataforma Render.

Cada sprint foi documentada com apresentações detalhadas, incluindo metas, backlog, user stories, critérios de aceitação e resultados alcançados.

4. APRESENTAÇÃO FINAL

A apresentação final do FaculRide incluiu uma demonstração prática do sistema, com vídeos, capturas de tela e protótipo funcional no Figma. Foram destacados os fluxos principais (cadastro, login, busca e oferta de caronas, avaliações, painel do usuário), além de diferenciais como acessibilidade, segurança, integração com mapas, persistência real dos dados no PostgreSQL e a publicação do sistema na Render, tornando-o acessível online.

5. BACKLOG DO PRODUTO & USER STORIES

Backlog do Produto (ordenado por prioridade):

1. Cadastro e autenticação de usuários
2. Gerenciamento de usuários
3. Solicitação de carona

4. Oferta de carona
5. Busca de caronas disponíveis
6. Geolocalização e visualização no mapa
7. Notificações para usuários
8. Sistema de avaliação e comentários
9. Gerenciamento de conta
10. Histórico de caronas

User Stories (exemplos):

Como estudante, quero buscar caronas disponíveis para meu trajeto, para economizar tempo e custos. Critérios de aceitação: O estudante deve visualizar caronas com informações de motorista, horário e local.

Como motorista, quero oferecer caronas, para compartilhar despesas e ajudar colegas. Critérios de aceitação: O motorista pode cadastrar uma carona, definir horários, pontos de partida e número de vagas.

Como usuário, quero avaliar motoristas e passageiros, para garantir segurança e confiança na plataforma. Critérios de aceitação: Após cada carona, o usuário pode dar uma nota de 1 a 5 estrelas e deixar um comentário.

Ferramentas Utilizadas: Jira para gerenciamento do backlog, user stories, tarefas e acompanhamento das sprints. O Jira permitiu a visualização clara do progresso, a priorização das demandas e o registro de todas as atividades do time. Figma para prototipação e validação de interfaces.

6. PROTÓTIPO

Baixa fidelidade: Wireframes em papel e digital, validando fluxos e arquitetura da informação.

Alta fidelidade: Protótipo Figma, demonstrando todas as telas e interações principais.

Avaliação heurística: Checklist Nielsen, identificação e correção de problemas de usabilidade.

7. DOCUMENTAÇÃO DO PROJETO

Toda a documentação está organizada em uma pasta /docs no repositório, incluindo backlog, user stories, planejamento das sprints, revisões, lições aprendidas, requisitos, diagramas, código-fonte e relatórios.

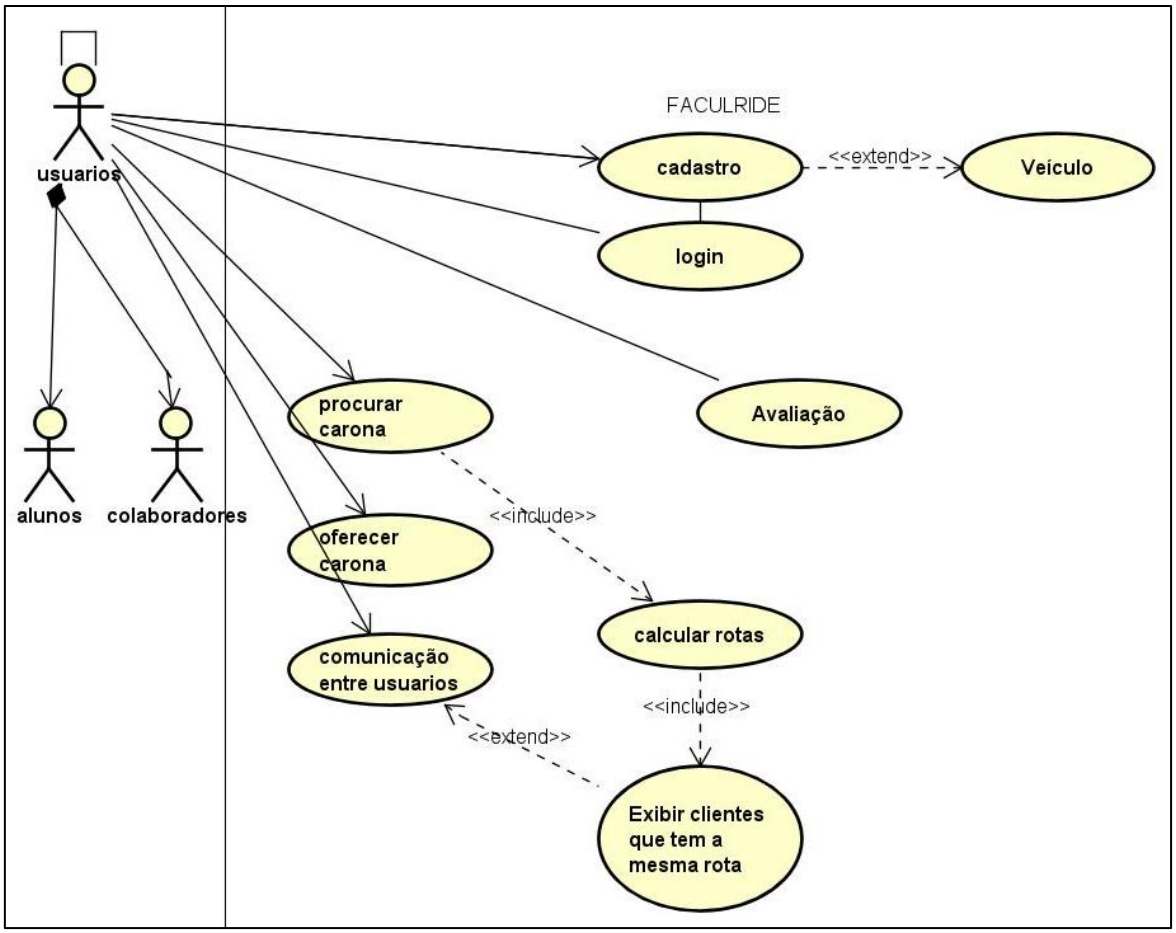
8. LEVANTAMENTO DE REQUISITOS

Técnicas utilizadas: Reuniões internas da equipe (brainstorm), pesquisa de mercado (Uber, BlaBlaCar, CaronaCar, Indrive) e análise de sistemas similares para identificar melhores práticas.

9. REQUISITOS FUNCIONAIS

N° Requisito Funcional	Nome	Descrição
RF001	Login	Fazer Login Intranet
RF002	Gerenciar Usuário	Consultar, cadastrar, alterar ou excluir usuário do site
RF003	Chamar Carona	Solicitar, passageiro solicita a sua carona ao motorista
RF004	Oferecer Carona	Oferecer carona, motorista disponibiliza a sua carona ao passageiro
RF005	Procurar Carona	Pesquisar e visualizar lista de motoristas disponíveis
RF006	Geolocalização	Rastrear localização dos passageiros e motoristas
RF007	Notificações	Enviar notificações via e-mail ou pelo site para informar os usuários
RF008	Avaliação	Implementar um sistema de avaliação de 1 a 5 para os usuários
RF009	Comentários	Comentários relacionados aos usuários

10. DIAGRAMA DE CASO DE USO



11. REQUISITOS NÃO FUNCIONAIS

N° Requisito Não Funcional	Nome	Descrição
RNF001	Banco de dados	Utilização de Banco de dados SQL Server
RNF002	HTML	Utilizar linguagem HTML
RNF003	Segurança	O sistema deve ser protegido contra acessos não autorizados
RNF004	Escalabilidade	O sistema deve ser capaz de aumentar ou diminuir sua capacidade
RNF005	Velocidade e Eficiência	O sistema deve ser capaz de aumentar ou diminuir sua capacidade de tráfego de dados conforme necessário, para acomodar picos de demanda.
RNF006	Disponibilidade	O sistema deve estar disponível 24 horas por dia, 7 dias por semana, para que os usuários possam solicitar viagens a qualquer momento.
RNF007	Usabilidade	A interface do usuário deve ser intuitiva e fácil de usar, mesmo para pessoas com pouca experiência em tecnologia.

RNF008	Confiabilidade	O sistema deve ser robusto e resistente a falhas.
RNF009	Portabilidade	O aplicativo deve funcionar em diferentes dispositivos (smartphones, tablets, etc.) e sistemas operacionais (iOS, Android, etc.).
RNF010	Manutenibilidade	O código-fonte deve ser bem organizado e documentado para facilitar futuras atualizações e correções.

12. PROJETO DO SOFTWARE

Arquitetura: SPA (Single Page Application) com Angular no front-end, back-end responsável pela lógica de negócio e integração com o banco de dados PostgreSQL, persistência real dos dados, e integração com Google Maps/OpenLayers para geolocalização. O sistema foi publicado e está disponível online por meio da plataforma de hospedagem Render, garantindo acesso remoto, escalabilidade e facilidade de manutenção.

Inserir diagrama de arquitetura mostrando front-end Angular, back-end, banco de dados PostgreSQL, Render e integrações externas.

13. TECNOLOGIAS UTILIZADAS

Front-end: Angular, Bootstrap, HTML5, CSS3. Back-end: Node.js (ou outra tecnologia utilizada), integração com PostgreSQL. Banco de Dados: PostgreSQL. APIs: Google Maps/OpenLayers. Hospedagem: Render. Controle de Versão: Git, GitHub. Prototipação: Figma. Gestão: Jira.

14. INTEGRAÇÃO E PAPÉIS DA EQUIPE

Breno Jose Da Silva: Gerente de Projeto e Documentação – responsável pelo gerenciamento do projeto, organização das tarefas, acompanhamento das sprints, elaboração e revisão da documentação.

Herivelton Henrique Gonçalves: Desenvolvedor Full-Stack – responsável pelo desenvolvimento do front-end (Angular) e pela implementação da lógica de negócio, integração com o banco de dados PostgreSQL e publicação na Render.

Gabriel Ribeiro Correa: Desenvolvedor – responsável pela implementação do back-end, lógica de dados, integração dos fluxos e funcionalidades, conexão com o PostgreSQL e suporte à publicação na Render.

Wendel Augusto Lopes Vasco: Gerente de Projeto e Documentação – responsável pelo gerenciamento do projeto, apoio na organização das tarefas, elaboração e revisão da documentação, testes e validação.