YAN GUSTAVO ANRADE PISANI

RELATÓRIO DE ESTÁGIO:

MANUTENÇÃO E DESENVOLVIMENTO WEB E MOBILE NA EMPRESA GAFIT

LAVRAS – MG

2023

YAN GUSTAVO ANRADE PISANI

RELATÓRIO DE ESTÁGIO:
MANUTENÇÃO E DESENVOLVIMENTO WEB E MOBILE NA EMPRESA GAFIT

Relatório de estágio supervisionado apresentado à Universidade Federal de Lavras, como parte das exigências do Curso de Ciência da Computação, para obtenção do título de Bacharel.

Prof. DSc. Raphel Winckler de Bettio
Orientador

LAVRAS – MG

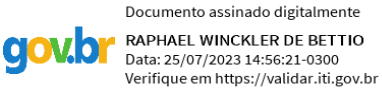
2023

YAN GUSTAVO ANRADE PISANI

RELATÓRIO DE ESTÁGIO: MANUTENÇÃO E DESENVOLVIMENTO WEB E MOBILE NA EMPRESA GAFIT

Relatório de estágio supervisionado apresentado à Universidade Federal de Lavras, como parte das exigências do Curso de Ciência da Computação, para obtenção do título de Bacharel.

APROVADA em 13 de Julho de 2023.



Prof. DSc. André Grützmann UFLA
Prof. DSc. Rafael Serapilha Durelli UFLA

Prof. DSc. Raphel Winckler de Bettio
Orientador

LAVRAS – MG
2023

AGRADECIMENTOS

Agradeço primeiramente a minha família, em especial meus pais, José Ricardo e Rita de Cássia, e minha irmã, Jéssica Alana pelo apoio incondicional, por todo amor, e por sempre estarem presentes durante toda minha vida e trajetória acadêmica.

Ao meu orientador Raphael, por instruir-me e pela paciência.

À Aparecida Consuelo e Demétrius, pelo abrigo e todo amor e carinho que me deram.

À empresa GAFIT e sua equipe pela oportunidade e por todo o conhecimento que pude adquirir.

Por fim, mas não menos importante, agradeço à todos meus amigos, antigos e novos.

Despite everything, it's still you.
(Robert F. Fox, "Undertale")

RESUMO

O trabalho busca apresentar as experiências e desafios do aluno durante seu estágio na empresa GAFIT SOLUÇÕES EM AUTOMAÇÃO LTDA, onde teve a oportunidade de trabalhar como um desenvolvedor *front-end* e *back-end* nos projetos de sistemas WEB e aplicações *mobile* da empresa. Devido à pandemia de SARS-CoV-2 (COVID-19), a companhia tornou todas as suas atividades remotas, e é discorrido no texto como a companhia – utilizando ferramentas como o Gather, para recriar o escritório em um mundo virtual – pôde superar este obstáculo. O relatório também apresenta as ferramentas utilizadas e atividades realizadas durante a trajetória profissional do aluno na empresa.

Palavras-chave: Desenvolvimento WEB. Desenvolvimento Mobile. Laravel. PHP. Java. Objective-C. Swift. Scrum. Metaverso.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

O trabalho a seguir apresenta as experiências do aluno na empresa GAFIT Soluções como um desenvolvedor *full-stack* (profissional capaz de trabalhar tanto no desenvolvimento do *front-end* quanto no *back-end* de um sistema ou aplicação). O relatório expõe a aplicação de conhecimentos aprendidos durante o curso de Ciência da Computação e sua aplicação em um ambiente profissional, conhecimentos adquiridos dentro da empresa, e vivências do aluno durante o período de estágio, discorrido no formato de um relatório de estágio. O estagiário atuou remotamente para a empresa, pois a mesma possui todos os seus recursos alocados virtualmente.

A empresa possui vários projetos, voltados principalmente para sistemas WEB e plataformas *mobile*, e o estudante em estágio teve a oportunidade de atuar em alguns deles. O primeiro projeto foi o desenvolvimento de um sistema WEB para o Estádio Governador Magalhães Pinto, mais conhecido como Mineirão, seguido pelo projeto de desenvolvimento e manuntenção de um aplicativo *mobile* para o grupo de empresas Alpha Transfers. A empresa também permitiu-o ter um contato mais próximo com clientes e indivíduos externos à empresa, refinando e ensinando-o conhecimentos além de apenas desenvolvimento de *softwares*, como melhorar habilidades sociais, organização de projetos e planilhas, e permitiu uma participação mais ativa dentro dos projetos.

Durante o período do estágio, o estágiário também pôde aplicar diversos conhecimentos adquiridos durante o curso, como conceitos de metodologia ágil, padrões de projeto, bancos de dados, modelagem, implementação, arquitetura e engenharia de *software*. Esses conhecimentos foram excepcionalmente necessários para que conseguisse exercer seu papel de desenvolvedor comfortavelmente e sem maiores dificuldades. As atividades realizadas foram relatadas neste documento, além de mostrar o ferramental utilizado durante o período de estágio, uma análise da importância do curso para realizar o estágio, e uma análise crítica da empresa.

2 REFERENCIAL TEÓRICO

Este capítulo tem como objetivo apresentar as técnicas e ferramentas utilizadas no decorrer do estágio. Ele serve como base para a compreensão das atividades desenvolvidas durante o estágio e o contexto em que elas estão inseridas.

2.1 Metodologias

2.1.1 Scrum

Scrum é uma estrutura leve que auxilia pessoas, times e organizações a gerar valor por meio de soluções adaptativas para problemas complexos. Destaca-se pela maior ênfase dada ao gerenciamento do projeto. Baseia-se em uma abordagem iterativa e incremental, e oficialmente existem três responsabilidades principais (SCHWABER; SUTHERLAND, 2020):

- Product Owner: representa os interesses de todos no projeto. É o proprietário do produto ou um representante da empresa que atua diretamente com os clientes e/ou usuários e orienta a equipe de desenvolvimento na construção do produto correto (validação).
- Scrum Team: desenvolve as funcionalidades do produto. São interdisciplinares (membros do time devem possuir todo conhecimento necessário para criar um incremento do software), auto-organizável e não há líder geral.
- Scrum Master: garante que todos sigam as regras e práticas do Scrum. Também é responsável por remover os impedimentos do projeto.

O principal evento realizado no Scrum é a *Sprint* (na tradução literal, uma "arrancada") onde no período de um mês ou menos é desenvolvida uma versão incremental e usável de um produto. A *Sprint* é oficialmente dividida em quatro etapas (SCHWABER; SUTHERLAND, 2020):

- *Sprint Planning*: *Product Owner* e *Scrum Team* decidem, em conjunto, o que deverá ser implementado.
- *Daily Meetings*: Reuniões diárias de aproximadamente 15 minutos entre os desenvolvedores do *Scrum Team*, onde se comunicam entre si para identificar impedimentos, informar os objetivos diários e ajustar seus planos.

- *Sprint Review*: *Scrum Team* e *stakeholders* revisam o progresso da *Sprint*, e com essa informação, em conjunto, decidir o que fazer subsequentemente.

- *Sprint Retrospective*: Tem a intenção de inspecionar a *Sprint* prévia, e procura mudanças para melhorar sua efetividade para próxima *Sprint*.

Existem também artefatos do Scrum, que, representam trabalho ou valor, e são projetadas para maximizar transparência de informações. São elas (SCHWABER; SUTHERLAND, 2020):

- *Product Backlog*: Uma lista ordenada, priorizada de requisitos do projeto, em constante mudança e adaptação. Dele, são selecionados items que podem ser desenvolvidos pelo *Scrum Team* em uma Sprint (*Sprint Backlog*).

- *Sprint Backlog*: Meta da *Sprint*, conjunto de items selecionados do *Product Backlog*, e um plano de ação para a entrega do incremento do projeto.

2.2 Linguagens e Frameworks

2.2.1 HTML/CSS

HyperText Markup Language (HTML), define o significado e a estrutura do conteúdo da web. Geralmente utilizada em conjunto com CSS (para aparência/apresentação) e JavaScript (para funcionalidade/comportamento) em uma página da web. Com HTML é possível conectar páginas da Web entre si, seja dentro de um único *site* ou entre *sites* (*Hypertext*). É utilizado marcação para realizar anotações em texto, imagens e outros conteúdos para exibição em um navegador web. (*Markup*) (MOZILLA FOUNDATION, 2023b).

Cascading Style Sheets (CSS), é uma linguagem *stylesheet* utilizada juntamente com HTML para descrever a apresentação de um documento. O CSS descreve como elementos são mostrados na tela (MOZILLA FOUNDATION, 2023a), incluindo cores, *layout* e fontes. Permite adaptar a apresentação para differente tipos de dispositivos, como telas grandes ou pequenas. CSS é indepentente do HTML e pode ser usado com qualquer linguagem XML baseada em marcação. A separação do HTML de CSS facilita a manutenção de websites, compartilhar estilos entre páginas, e preparar páginas para outros ambientes (W3C, 2010).

2.2.2 JavaScript

JavaScript é uma linguagem para scripts orientada à objeto de alto-nível, dinamicamente tipada, multiparadigma, multiplataforma e é uma das tecnologias principais, juntamente com HTML e CSS, para o desenvolvimento WEB (MOZILLA FOUNDATION, 2023d). Como informado por (FLANAGAN, 2020), a linguagem de JavaScript possui uma sintaxe vagamente baseada em Java, mas as duas linguagens não possuem nenhuma relação. JavaScript é utilizada para criar conteúdo *web* dinâmico e interativo, como aplicações e navegadores e tornar páginas *web* interativas (por exemplo, adicionar animações complexas, botões clicáveis e menus *popup*) (MOZILLA FOUNDATION, 2023d).

As capacidades dinâmicas de JavaScript incluem construção de objetos em tempo de execução, listas de parãmetros variáveis, variáveis de funções, criação dinâmica de scripts, introspecção de objetos e recuperação de código-fonte. JavaScript contém uma bibilioteca padrão de objetos como *Array*, *Date* e *Math*, e um conjunto de elementos de linguagem como operadores, estruturas de controle e declarações. JavaScript padrão pode ser utilizado para vários propósitos ao suplementá-la com objetos adicionais (por exemplo, *Client-side* JavaScript extende a linguagem ao suplementá-la com objetos para controlar o navegador e seu *Document Object Model* (DOM)) (MOZILLA FOUNDATION, 2023c).

2.2.3 PHP

Hypertext Preprocessor (PHP) é uma linguagem de *script* em código aberto de uso geral, adequada para o desenvolvimento web e que pode ser embutida dentro do HTML (THE PHP GROUP, 2023). De acordo com a sua documentação (THE PHP GROUP, 2023), é focado em *server-side scripts*, portanto, com a linguagem você pode coletar dados de formulários, gerar páginas com conteúdo dinâmico, enviar e receber cookies. PHP possui várias vantagens, como performance, escalabilidade, interfaces para vários sistemas de bancos de dados diferentes, bibliotecas pré-incorporadas para muitas tarefas comuns no desenvolvimento WEB, portabilidade e forte suporte à orientação a objetos (THOMSON; WELLING, 2017).

O que distingue o PHP de algo como o *client-side* JavaScript é que o código é executado no servidor, gerando o HTML que é então enviado para o navegador. O navegador recebe os resuldados da execução desse script, mas não sabe qual era o código fonte. Portanto, pode-se

configurar o servidor web para processar todos os arquivos HTML com o PHP, e os usuários não podem visualizar o código fonte. Além disso, PHP pode ser utilizado na maioria dos sistemas operacionais, incluindo Linux, várias variantes do Unix, Microsoft Windows, macOS, RISC OS (THE PHP GROUP, 2023).

2.2.4 Java

Como informado por (CHAGAS; BARUQUE; BARUQUE, 2010), Java é uma linguagem de programação orientada a objetos, estruturada, imperativa, genérica, funcional, reflexiva e concorrente, podendo ser utilizada em aplicações simples no *desktop*, até complexos aplicativos *web* e *mobile*. A linguagem possui algumas características importantes, como (CHAGAS; BARUQUE; BARUQUE, 2010):

- Java Virtual Machine (JVM), que provê especificações de plataforma de *hardware* na qual se compila todo código de tecnologia Java, permitindo que o *software* Java seja uma plataforma independente;
- Coletor de lixo, responsável pela liberação automática do espaço em memória;
- Java Runtime Environment (JRE), ambiente que roda os códigos compilados para a JVM, executa o carregamento de classes, verificação de código, e o código executável;

As vantagens principais da linguagem Java são sua portabilidade e segurança. Segurança pois ao isolar a aplicação dentro do ambiente de execução Java, previne o acesso à outras partes do dispositivo, impedindo o acesso de códigos prejudiciais (por exemplo, um *malware*), e portabilidade, pois códigos gerados em Java funcionam em qualquer máquina que possua JVM (SCHILDT, 2019).

2.2.5 Objective-C/Swift

Objective-C era a linguagem principal utilizada para desenvolver software para OS X e iOS. É um superconjunto da linguagem C, possui orientação a objetos e *runtime* dinâmico. Objective-C herda a sintaxe, tipos primitivos e instruções de controle de fluxo de C e adiciona sintaxe para definir classes e métodos (APPLE INC., 2014a).

Swift foi criado em 2014 como um substituto para o Objective-C, pois faltavam funcionalidades presentes em linguagens mais modernas, como polimorfismo, tipagem dinâmica e inferência de tipos. Além disso, as variáveis sempre são inicializadas antes do uso, indices de vetores são checados por erros *out-of-bounds*, inteiros são checados por *overflow* e memória é manuseada automaticamente (APPLE INC., 2022).

Assim como C, Swift faz uso de variáveis para armazenar e referenciar valores utilizando um nome identificador. Swift também faz uso extensivo de constantes. Além disso, Swift também introduz tipos avançados não encontrados em Objective-C, como tuplas e tipos opcionais, que lidam com a ausência de valor, similar à utilizar *nil* com ponteiros em Objective-C, mas que funcionam com qualquer tipo, não só classes. Opcionais são mais seguros e expressivos do que ponteiros *nil* em Objective-C, e um exemplo de que Swift é uma linguagem com segurança de tipos (*type-safe*) (APPLE INC., 2014b).

2.2.6 Laravel

Laravel é uma *framework* de aplicação *web* baseada em Hypertext Preprocessor (PHP), com uma sintaxe expressiva e elegante e altamente escalável. Laravel provém o usuário ferramentas robustas para injeção de dependência, testes unitários, filas, eventos em tempo real e mais. Por padrão, um projeto Laravel divide-se nos seguintes diretórios (LARAVEL, 2022):

- App: Contém o código núcleo da aplicação, quase todas as classes estarão localizadas nesse diretório;
- Bootstrap: Contém o arquivo que realiza o *bootstrap* da *framework*, e também possui um *cache* que contém os arquivos gerados pela *framework* para optimização como arquivos *cache* de rotas e serviços;
- Config: Contém todos os arquivos de configuração do aplicativo;
- Database: Contém as migrações do banco de dados, *model factories* e *seeds*.
- Lang: Contém todos os arquivos de linguagem do aplicativo;
- Public: Contém o arquivo de entrada de todas as requisições entrando no aplicativo e configura *autoloading*. Também contém todos os ativos, como imagens, JavaScript e CSS;

- Resources: Contém as *views* e ativos não-compilados do CSS ou JavaScript;
- Routes: Contém todas as definições e rotas da aplicação. Por padrão os arquivos de rota incluídos são o arquivo *web*, *api*, *console* e *channels*;
- Storage: Contém os *logs*, *templates* compiladas de Blades, *caches* de arquivos e outros arquivos gerados pela *framework*;
- Tests: Contém os testes automatizados;
- Vendor: Contém as dependências do Composer (um gerenciador de dependências para PHP);

Laravel também inclui uma ferramenta de *template* poderosa chamada Blade. Blades permitem que você utilize PHP puro em suas *templates*. Todas as *templates* Blade são compiladas em código PHP puro e colocadas em um *cache* até serem modificadas, o que significa que Blade adiciona quase zero *overhead* na aplicação. Também incluído em Laravel, temos o Eloquent (LARAVEL, 2022). Eloquent é um mapeador de objeto relacional (ou *Object Relational Mapper* (ORM)), software que facilita o manuseamento dos registros do banco de dados ao representar dados como objetos, funcionando como uma camada de abstração sobre o banco de dados utilizado para guardar os dados da aplicação (HAWB, 2020).

Quando fazendo uso do Eloquent, cada tabela do banco de dados possui um *"Model"* correspondente, utilizado para interagir com aquela tabela. Além de ter a capacidade de recuperar registros da tabela do banco de dados, *models* do Eloquent permitem a realização de inserções, atualizações, e deleções de registros da tabela. Por fim, Laravel também possui a interface Laravel Sail, uma solução pré-incorporada na *framework* para rodar sua aplicação utilizando Docker (LARAVEL, 2022).

2.2.7 MySQL

MySQL é um sistema gerenciador de banco de dados relacional em código aberto que utiliza a linguagem SQL. Um banco de dados é uma coleção de dados estruturada. Para acessar, adicionar e processar dados armazenados em um banco de dados de um computador, é necessário um sistema gerenciador de banco de dados, como o servidor MySQL. Já bancos de dados relacionais armazenam dados em tabelas separadas. As estruturas dos bancos de dados são

organizadas em arquivos para otimizar a velocidade. O modelo lógico, com objetos como bancos de dados, tabelas, *views*, linhas e colunas, oferecem um ambiente de programação flexível (ORACLE, 2023a).

MySQL é rápido e escalável. Um servidor MySQL funciona tanto em sistemas *desktops* quanto em *laptops*, juntamente com outras aplicações e servidores WEB. Também é possível dedicar um dispositivo inteiro para o MySQL, permitindo ajustar suas configurações para fazer uso de toda a memória, CPU e capacidade de entrada e saída disponível. MySQL também pode escalar *clusters* de máquinas conectadas em rede. Além disso, o *software* de banco de dados MySQL é um sistema cliente/servidor que possui um servidor SQL *multithreaded* que suporta *back ends* diferentes, vários programas e bibliotecas cliente diferentes, ferramentas administrativas e várias *Application Programming Interfaces* (APIs) (ORACLE, 2023a).

2.3 Softwares de Apoio

2.3.1 Vagrant

Vagrant é uma ferramenta para construção e manuseamento de ambientes de máquinas virtuais. Fornece ambientes de trabalho fáceis de configurar, reproduzíveis e portáteis. É uma ferramenta para construir ambientes de desenvolvimento completos. Vagrant isola dependências e suas configurações em um ambiente descartável, sem sacrificar nenhuma das ferramentas que estão sendo utilizadas (editores, *browsers*, *debuggers*) (HASHICORP, 2023).

Quando é criado um *Vagrantfile* (arquivo que descreve o tipo de máquina necessária para um projeto, como configurá-la e provisioná-la. É necessário apenas um *Vagrantfile* por projeto (HASHICORP, 2023)) pelo desenvolvedor ou organizador do projeto, é necessário apenas um comando (*vagrant up*), e Vagrant realizará todas as instalações e configurações necessárias para deixar o projeto pronto para ser trabalhado, ou seja, Vagrant auxilia na maximização de produtividade e flexibilidade de times de desenvolvimento ao preparar ambientes e reduzir o tempo de configuração de projetos para os desenvolvedores (HASHICORP, 2023).

2.3.2 MySQL Workbench

MySQL Workbench é uma ferramenta visual para *design* de bancos de dados e servidores MySQL. Desenvolvimento SQL, modelagem (design) de dados, administração de servidores e migração de dados são as funcionalidades da ferramenta (MYSQL, 2023). MySQL Workbench permite o usuário criar e gerenciar conexões com servidores de bancos de dados, além de permitir a configuração de parâmetros de conexão, e também fornece a capacidade de executar consultas SQL nas conexões dos bancos de dados utilizando o editor SQL embutido (ORACLE, 2023b).

Além disso, MySQL Workbench também permite o usuário criar modelos do esquema de bancos de dados graficamente, realizar engenharia reversa e engenharia progressiva entre um esquema e um banco de dados ativo, editar todos os aspectos do banco utilizando um editor de tabelas. O editor de tableas fornece ferramentas para editar tabelas, colunas, índices, gatilhos, particionamento, opções, inserções, privilégios, rotinas e *views*. Também permite o usuário administrar instâncias de servidores MySQL, administrar usuários, realizar *backups* e recuperações, inspecionar dados de auditorias, verificar saúde do banco, e monitorar o desempenho dos servidores (ORACLE, 2023b).

2.3.3 WSL2

WSL (*Windows Subsystem for Linux*) é uma funcionalidade do Windows que permite desenvolvedores rodarem um ambiente Linux sem a necessidade de uma máquina virtual ou *dual booting*. WSL 2 trouxe mudanças importantes como a utilização de um *kernel* Linux verdadeiro, utilizando um subconjunto de funcionalidades Hyper-V. A diferença entre o WSL 2 e 1 é que WSL 2 roda dentro de uma máquina virtual manuseada que implementa um *kernel* Linux real, e por conta disso, WSL 2 tem compatibilidade com mais binários Linux do que o WSL 1, que não possuia todas as chamadas de sistema implementadas (MICROSOFT, 2022a).

A funcionalidade do Windows requer menos recursos (CPU, memória, armazenamento) do que uma máquina virtual completa. WSL também permite o usuário fazer uso de ferramentas das linhas de comando do Linux juntamente com as linhas de comando do Windows, aplicativos, e acessar seus arquivos do Windows dentro de um Linux (MICROSOFT, 2022b). Outra adição do WSL2 sobre o WSL1 são as operações onde há uso intenso de arquivos (como

git clone, *npm install* e *apt update/upgrade*) ocorrem com maior velocidade. (MICROSOFT, 2022a).

2.3.4 Docker

Docker é uma ferramenta para rodar aplicações e serviços em *containers* (agrupamento de código de uma aplicação com todos os arquivos e bibliotecas necessárias para ser executado em qualquer infraestrutura) leves que não interferem com a configuração ou software instalado localmente, utilizando virtualização de nível de sistema operacional. Todos os contêiners são executados por um único kernel do sistema operacional (DOCKER INC., 2023a). Docker otimiza o ciclo de vida do desenvolvimento ao permitir desenvolvedores trabalharem em ambientes padronizados utilizando *containers* locais que fornecem suas aplicações e serviços. São úteis para fluxos de trabalho com integração e entrega contínua (DOCKER INC., 2023b).

A plataforma baseada em *containers* do Docker permite cargas de trabalho altamente portáteis. Sua natureza portátil e leve também permite o gerenciamento dinâmico de cargas de trabalho e escalar aplicações em quase tempo real. Docker fornece uma alternativa viável e de maior custo-benefício à máquinas virtuais baseadas em *hypervisor*. Docker é uma ferramenta ótima para ambientes de alta densidade, e para implantações pequenas e médias. Docker utiliza uma arquitetura cliente-servidor, onde o cliente Docker se comunica com o Docker *daemon*, que realiza as construções e distribuições dos *containers* Docker, e se comunicam utilizando uma API REST ou uma interface de rede (DOCKER INC., 2023b).

2.4 Ambientes de Desenvolvimento

2.4.1 PhpStorm

PhpStorm é um ambiente de desenvolvimento integrado voltado para a linguagem de programação PHP, HTML, CSS e JavaScript, publicado pela empresa JetBrains (JETBRAINS, 2022). O editor possui várias ferramentas pré-incorporadas, como preenchimento de código inteligente, realce de sintaxe e verificação de erros dinâmica para facilitar o desenvolvimento em PHP. Além disso, também possui uma interface de usuário unificada para controle de versão (funcionando para ferramentas como Git, Mercurial, Subversion e Perforce), ferramentas de

teste como *PHPUnit*, permite a visualização da arquitetura de bancos de dados (e.g MySQL e SQLite), e por fim, permite a integração de ferramentas para operações de servidor como *Vagrant e Docker* (JETBRAINS, 2023).

2.4.2 Android Studio

Android Studio é um ambiente de desenvolvimento integrado para desenvolver para a plataforma Android, baseado no software IntelliJ de JetBrains, que suporta várias linguagens de programação como Java, HTML, CSS, JavaScript, Kotlin e Groovy. Para permitir desenvolvimento no sistema operacional Android, Android Studio faz uso de um sistema de *build* baseado em Gradle (sistema de automação de compilação de código aberto que se baseia nos conceitos de Apache Ant e Apache Maven), um emulador Android, *templates* de código, e integração com Git. Além de seu editor oferecer compleção de código, análise e refração (GOOGLE DEVELOPERS, 2022). Por fim, Android Studio também possui uma funcionalidade pré-incorporada de aplicar mudanças de código e recursos em uma aplicação rodando.

2.4.3 Xcode

Xcode é um ambiente de desenvolvimento integrado e software livre da empresa Apple para o macOS, utilizado para desenvolver software para macOS, iOS, iPadOS, watchOS e tvOS. Suporta principalmente as linguagens de programação Objective-C, Swift e Apple-Script. Além disso, também suporta C/C++, Java, Python, Ruby e ResEdit. Por padrão, Xcode inclui a ferramenta Instruments, um analisador e visualizador de *performance* do aplicativo. Também integrado com o sistema controle de versões Git (APPLE INC., 2023).

2.5 Outras Tecnologias

2.5.1 Gather

Gather é uma plataforma Metaverso criada em 2020 buscando tornar interações virtuais mais humanas. Gather pode ser utilizado para conversar ou colaborar com outras pessoas, como transferir arquivos e links, além de permitir apresentações utilizando compartilhamento de telas e utilização de um quadro-branco presente no universo virtual. De acordo com (FITRIA, 2021), salas no Gather são chamadas de espaços, e espaços são um mapa 2D com um display 8-bit e, assim como uma conferência no mundo real ou outras reuniões, o usuário pode facilmente iniciar conversas paralelas, compartilhar documentos fazendo uso do pacote de aplicativos disponibilizado pelo Google (Google Docs) para colaborar em trabalhos ou tarefas, e até inserir vídeos do YouTube para serem assistidos juntos.

Dentro da ferramenta Gather, o usuário têm um personagem onde pode personalizá-lo da maneira que desejar, e também têm a permissão de criar áreas que podem ser visitadas por outros usuários (a área também pode ser modificada da maneira que o usuário preferir). Dentro dessas áreas virtuais, podem ser inseridos blocos que servem como "isolamento acústico", ou seja, usuários não presentes dentro destes blocos não podem escutar (ou visualizar o *chat* de texto) essas áreas à não ser que movimentem o seus personagens para dentro desta área demarcada. Essa é uma ferramenta muito importante para organização de ambientes de trabalho onde existem vários projetos em desenvolvimento simultaneamente, pois reuniões e conversas estão limitadas apenas às áreas designadas.

2.5.2 ClickUp

ClickUp é uma ferramenta de gerenciamento de tarefas e colaboração baseada em nuvem. Possui funcionalidades como ferramentas de colaboração e comunicação, atribuição de tarefas e *status*, alertas e uma barra de ferramentas de tarefas (MUDRAKOLA, 2022). Usuários podem atribuir comentários e tarefas a integrantes do time específicos ou grupos de integrantes. Comentários e tarefas podem ser marcados como terminados ou em progresso, ou podem ser criados *status* customizados. Projetos podem ser vistos em uma *dashboard* ágil ou organizados

por um integrante. Usuários também podem configurar notificações para serem enviadas apenas em itens específicos, e também possui integrações com Slack e GitHub.

ClickUp é uma plataforma que pode ser modificada para qualquer caso de uso, de simples à complexo. O usuário consegue criar *views* personalizadas para ter uma visão ampla de todos os processos da organização (CLICKUP, 2023). Com a ferramenta, gerenciadores de projetos podem delegar tarefas aos integrantes do projeto, traçar linhas do tempo e monitorar funcionários. Podem ser atribuidas tarefas e subtarefas para integrantes do time, que são notificados no momento que são atribuídos à tarefa. Além disso, também pode-se monitorar o progresso dos participantes do projeto e criar, visualizar e modificar gráficos de Gantt (CLICKUP, 2020).

2.5.3 Kanban

Kanban é um método para gerenciar todos os tipos de serviços profissionais, também conhecidos como trabalho do conhecimento. Kanban é fortemente baseado no *lean*: Foco no fluxo de trabalho, limitar o trabalho em andamento para estabelecer sistemas puxados, foco na otimização do sistema como um todo; ao invés de gerenciar o desempenho individual, tomada de decisões baseadas em dados e melhoria contínua de maneira evolutiva. Quadros Kanban são o meio mais comum de visualizar um sistema Kanban. (KANBAN UNIVERSITY, 2021).

Com a utilização do Kanban, um entendimento avançado do fluxo de trabalho é desenvolvido. Kanban pode auxiliar, por exemplo, em descobrir uma estimativa de quando uma tarefa específica será completada, permitindo tomadas de decisão mais rápidas. Também pode-se medir o serviço providenciado, julgar o impacto de mudanças introduzidas, e utilzar esses dados para prever as capacidades de entrega (DAVID J ANDERSON SCHOOL OF MANAGEMENT, 2021).

2.5.4 GitLab

GitLab é um software *open source* para colaboração de códigos. Além de ferramentas de versionamento de códigos, também possui várias ferramentas de planejamento, criação, verificação (e.g Integração Contínua), segurança, e monitoramento (GITLAB INC., 2023b). GitLab fornece capacidades DevSecOps (Desenvolvimento, Segurança e Operações) em uma aplicação

com armazenamento de dados unificado, para ter a habilidade de acessá-los em um local único. GitLab também fornece ferramentas de automação para substituir processos manuais (GITLAB INC., 2023c).

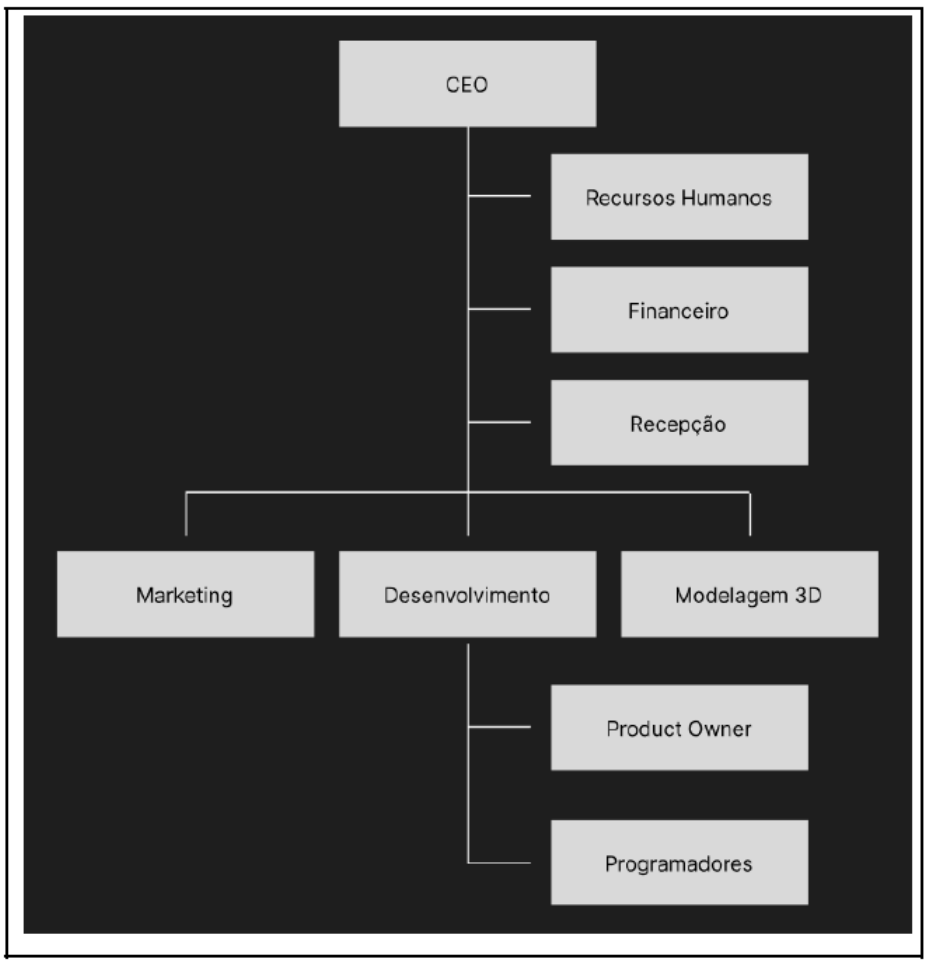
Enquanto GitHub é uma plataforma de colaboração para auxiliar revisão e gerenciamento de códigos remotamente, GitLab tem um foco maior em integração e entrega contínua e ferramentas DevSecOps. GitLab fornece ferramentas como rastreio e quadros de *issues*, gerenciamento de portifólio e código fonte, revisão de código, repositório binário, *service desk*, configuração de infraestrutura, integração e entrega contínua, testes de segurança e monitoramento de desempenho de aplicativos (GITLAB INC., 2023a). Portanto, GitLab é uma ferramenta mais recomendada caso o usuário esteja trabalhando em um projeto privado, pois o GitLab fornece mais ferramentas e um controle maior sobre o acesso de usuários.

3 ESTÁGIO

3.1 Estrutura da Empresa

A empresa é liderada pelo CEO, que também atua como Recursos Humanos, Financeiro e Recepção. Não há departamentos específicos, pois ele é o único responsável pelas três funções. A empresa também possui três times, um de marketing, um para modelagem 3D, e vários times de desenvolvimento, um para cada projeto. Cada time de desenvolvimento possui programadores; a quantidade de integrantes depende da demanda e/ou tamanho do projeto. (Figura 3.1)

Figura 3.1 – Organograma da Empresa



Como cada projeto possui uma equipe de desenvolvimento, os estagiários são alocados para uma dessas equipes (projeto) e ocasionalmente rotacionados dependendo da demanda dos projetos. A grande maioria dos integrantes da empresa trabalha como *full-stack*. O contato inicial com o cliente para financiamento do projeto é realizado pelo CEO da empresa, a partir desse

momento, os desenvolvedores mantém contato direto com os clientes para desenvolver, realizar mudanças, alterar prazos e sanar qualquer dúvidas relacionadas ao projeto. Caso haja algum problema que os desenvolvedores não sejam capazes de solucionar, ou que seja necessário o cancelamento do projeto, é novamente acionado o CEO.

A empresa faz uso de uma metodologia ágil baseada no Scrum (o *framework* Scrum é imutável. Embora a implementação de apenas partes do Scrum seja possível, o resultado não é Scrum (SCHWABER; SUTHERLAND, 2020)), na qual são realizadas *Daily Meetings* com todos os integrantes da empresa (diferentemente do Scrum comum, onde as reuniões são realizadas com o time Scrum), e todos os desenvolvedores têm contato direto com os clientes (ao invés de todas as informações passarem por um *Product Owner* e então para o *Scrum Team*). Além disso, o *Product Backlog* e *Sprint Backlog* eram gerados pelos próprios desenvolvedores e não havia um *Scrum Master* ou *Product Owner* designado. (Figura 3.2)

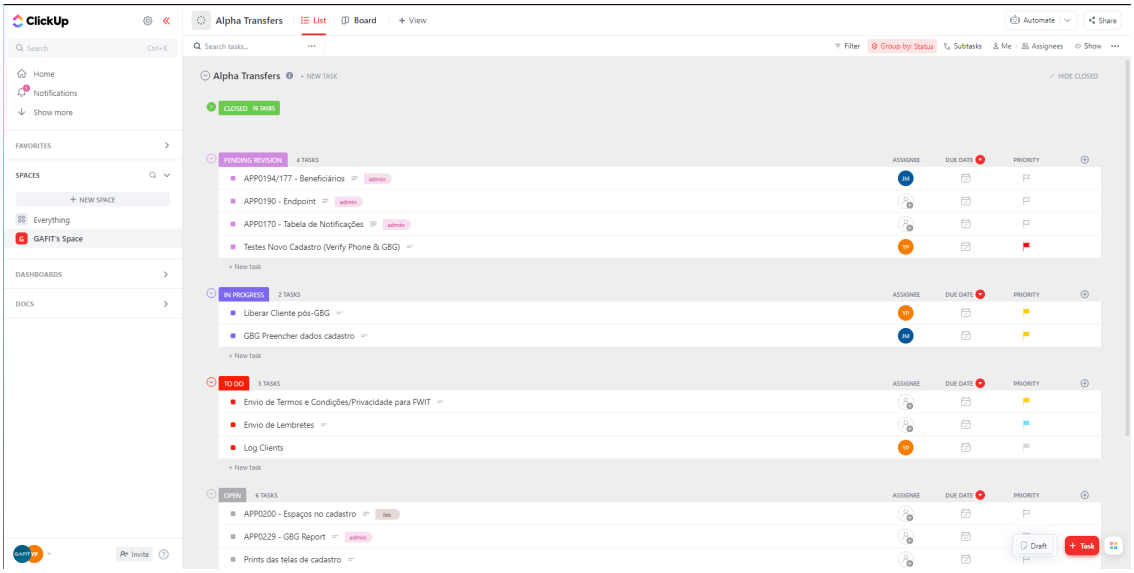
Figura 3.2 – Comparação entre o Scrum aplicado na GAFIT x Scrum de acordo com o Scrum Guide

Scrum na Teoria de acordo com o Scrum Guide (SCHWABER; SUTHERLAND, 2020)	Scrum na GAFIT Soluções
Trabalha com três responsabilidades principais: Product Owner, Scrum Team e Scrum Master.	Não há um Scrum Master ou Product Owner designado e todos os integrantes do Scrum Team também têm contato direto com o cliente.
No Sprint Planning, Product Owner e Scrum Team decidem, em conjunto, o que deverá ser implementado.	Sprints são definidas juntamente com os clientes, em reuniões entre o Scrum Team e os requisitores. Lá é produzido o Product Backlog e os Sprint Backlogs.
Daily Meetings são reuniões diárias que duram aproximadamente 15 minutos entre os desenvolvedores do Scrum Team, comunicando entre si para identificar impedimentos, informar os objetivos diários e ajustar seus planos.	Todos os integrantes da empresa participam das reuniões diárias, durando aproximadamente 40 minutos cada, onde todos comunicam o que foi feito, o que será realizado, e se há algum impedimento.
Sprint Reviews são realizadas para o Scrum Team e stakeholders revisarem o progresso da Sprint, e com essa informação decidir o que fazer subsequentemente.	As Sprint Reviews são regularmente realizadas entre o Scrum Team e os stakeholders para revisar o progresso da Sprint.
Sprint Retrospective têm a intenção de inspecionar a Sprint prévia, e procura mudanças para melhorar sua efetividade para próxima Sprint.	Não são realizadas Sprint Retrospectives.

A Figura 3.3 mostra a ferramenta ClickUp, onde é produzido o Kanban de cada projeto, com todas as tarefas, divididas em tarefas finalizadas (*Closed*), esperando revisão (*Pending Revision*), em desenvolvimento (*In Progress*), pendentes (*To-Do*), e abertas (*Open*). Além da divisão, também mostra à quem está atribuída a tarefa (*Assignee*), a data limite (*Due Date*), e a

prioridade da tarefa (*Priority*), separadas por urgente, prioridade alta, normal e baixa (determinadas pelas bandeiras vermelhas, amarelas, azuis e cinzas, respectivamente.)

Figura 3.3 – ClickUp do Projeto Alpha Transfers



O estágio foi completamente remoto, no entanto, todas as reuniões, confraternizações e comunicações da equipe foram realizadas através de um escritório localizado no Metaverso. A Figura 3.4 mostra a ferramenta Gather, onde foi criado um escritório virtual da empresa, nela foram realizadas todas as atividades relacionadas a GAFIT. O escritório possuia uma sala de reuniões, e cada integrante da empresa possuia seu próprio cubículo, isolado acusticamente dos outros, dessa forma conversas paralelas não atrapalhariam os desenvolvedores. Uma das principais vantagens da ferramenta Gather foi tornar as interações entre desenvolvedores muito mais rápidas e pessoais, como se o desenvolvedor realmente estivesse se levantando de sua mesa, e se aproximando de seu colega de trabalho. Essas interações tornaram o trabalho muito interativo, e sociável, permitindo todos os desenvolvedores conversarem entre si, seja discutindo sobre projetos ou até mesmo conversas não relacionados ao trabalho.

Figura 3.4 – Escritório Virtual da GAFIT

3.2 Projetos

Inicialmente, durante as duas semanas iniciais do estágio, o estagiário foi guiado para realizar estudos sobre a *framework* Laravel 9, por meio da documentação do mesmo, e de vídeo aulas recomendadas pela própria empresa. Sugeriram realizar os estudos de assuntos como a camada HTTP, *Views* e *Templates*, JavaScript e CSS, mas, principalmente, do Eloquent, incluído por padrão na estrutura do Laravel.

Além disso, essas semanas de estudo possibilitaram o estagiário conhecer a equipe por meio das *daily meetings* realizadas no Gather, além de configurar as ferramentas necessárias para poder contribuir com os projetos, como o MySQL Workbench (para a visualização dos bancos de dados), o Windows Linux Subsystem (WSL2) e o Docker (para execução e conteinerização dos projetos).

3.2.1 Mineirão

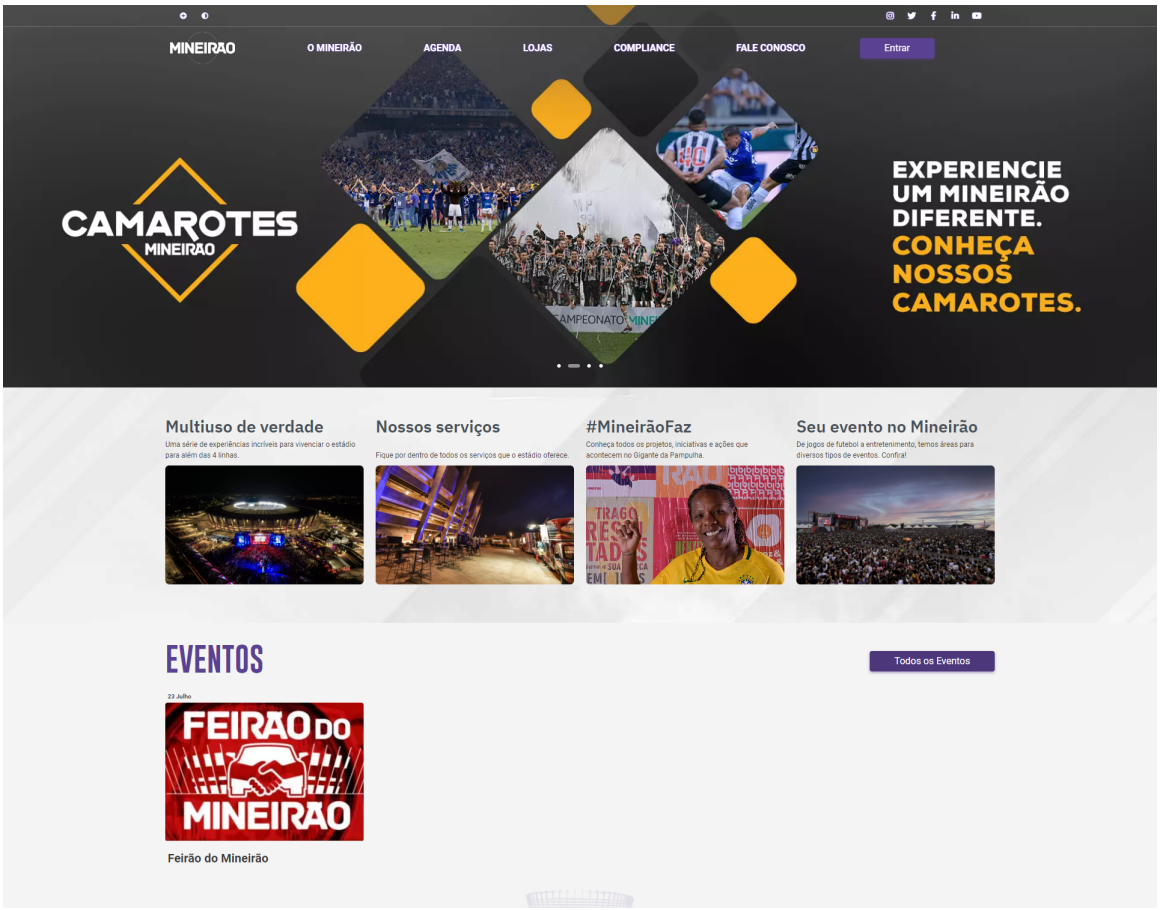
Após as semanas iniciais, o estagiário foi alocado primeiramente no projeto de desenvolvimento de um sistema WEB para o Estádio Governador Magalhães Pinto, mais conhecido como Mineirão. O projeto já estava em desenvolvimento, portanto, os integrantes da equipe o introduziram ao sistema e o informaram sobre o padrão de código e nomeclatura utilizado para

a criação de *branches* no GitLab da empresa, para versionamento de código. O sistema fazia uso do *framework* Laravel 9, Composer para gerenciar dependências, MySQL Workbench para gerenciar o banco de dados, e do WSL2 e Docker para virtualização.

O sistema WEB é dividido em duas partes:

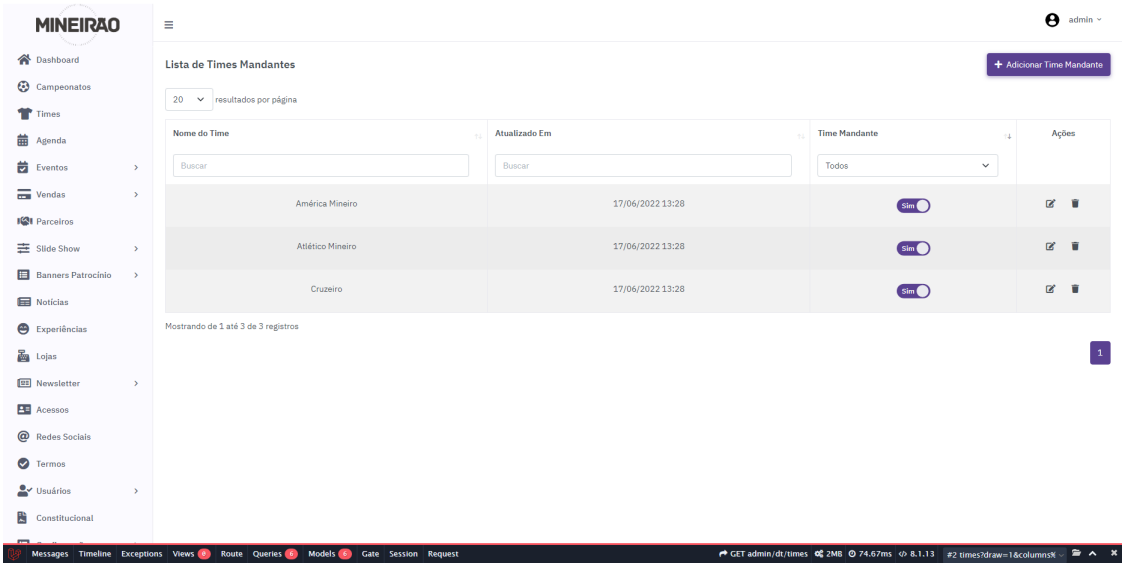
- Uma de acesso público, como mostrado na Figura 3.5, frente que permite os usuários visualizarem os eventos agendados do Mineirão, acessar notícias, projetos, lojas, e também realizar cadastro e compra de ingressos para os eventos.

Figura 3.5 – *Homepage* do Mineirão



- Uma de acesso privado, uma *dashboard*, como mostrado na Figura 3.6, que permite administradores manusearem o sistema; ver registros de vendas de ingressos/camarotes, usuários, agenda, adicionar notícias, campeonatos, times, lojas, *banners*.

Figura 3.6 – *Dashboard Admin* do Mineirão



3.2.1.1 Visão Geral das Atividades Realizadas

Figura 3.7 – Gráfico de Gantt do Projeto Mineirão

PROJECT TITLE	MINEIRÃO						COMPANY NAME	GAFIT																						
PROJECT MANAGER	GAFIT						DATE	01/11/22 - 25/11/22																						
							PHASE ONE																								
WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	PCT OF TASK COMPLETE	WEEK 1					WEEK 2					WEEK 3					WEEK 4								
							M	T	W	R	F	M	T	W	R	F	M	T	W	R	F	M	T	W	R	F				
	Mineirão																													
1	Corrigindo Botões Ticket-Select	Yan Pisani	11/1/22	11/1/22	1	100%																								
2	Corrigindo Banner Ticket-Select	Yan Pisani	11/2/22	11/4/22	4	100%																								
3	Corrigindo Links Eventos e Select-Ticket	Yan Pisani	11/7/22	11/8/22	2	100%																								
4	Tornando Eventos Indepententes de Agenda	Yan Pisani	11/8/22	11/11/22	4	100%																								
5	Melhorando Mapa Ticket-Select	Yan Pisani	11/11/2022	11/14/2022	2	100%																								
6	Testando/Corrigindo Agenda	Yan Pisani	11/16/22	11/17/22	2	100%																								
7	Bloquear Adição Eventos Datas Passadas	Yan Pisani	11/21/22	11/22/22	2	100%																								
8	Implementando Click & Drag Mapa Ticket-Select	Yan Pisani	11/23/22	11/24/22	2	100%																								
9	Corrigindo Compatiblidade	Yan Pisani	11/24/22	11/25/22	2	100%																								

Na Figura 3.7, pode-se observar um gráfico de Gantt onde são mostradas todas as atividades no qual o estagiário fez parte durante sua participação no projeto Mineirão. Primeiramente temos as correções mencionadas anteriormente, primeiramente focadas no *front-end* do projeto, onde foram feitas as correções estéticas dos botões seguindo um padrão de cor diferente e do banner na página *Ticket-Select*. (Linhas 1 e 2). Temos também as correções feitas aos links para remover a possibilidade de conflito caso haja eventos com mesmo nome. (Linha 3).

Também foi pedido para o estagiário realizar algumas correções na página de Agenda. Todo evento criado não era colocado na agenda automaticamente, e vice-versa. No entanto, Evento e Agenda são independentes, porém podem se relacionar. Além dessa correção, foi também requisitado para que não permitisse a adição eventos em datas que já passaram. Também foram feitos testes com as correções. (Linhas 4, 6 e 7). Por fim, as últimas tarefas que o estagiário trabalhou no projeto foram relacionadas as correções do mapa na compra de ingressos, que estava tendo problemas de compatibilidade, *hovers* desalinhados, e a implementação de *Click e Drag*. (Linhas 5, 8 e 9). Na seção abaixo será discorrido em detalhes algumas das tarefas realizadas.

3.2.1.2 Atividades Realizadas

Inicialmente o estágiário foi apresentado ao cliente, indivíduos que estavam representando o Mineirão, por meio de reuniões online (realizadas fora do Gather), onde foram discutidos *bugs* e inconsistências encontradas pelo cliente, e como o estagiário estava iniciando recentemente no projeto, não pôde contribuir muito para as reuniões iniciais, e utilizou-as para aprender o processo de interação da empresa com o cliente, entender mais sobre o projeto e funcionalidades novas que seriam inseridas. Após a reunião, o estagiário recebeu algumas tarefas pequenas voltadas ao *front-end* como, por exemplo, mostrado na Figura 3.8, onde os botões da tela de compra de ingressos estava seguindo o padrão de cores incorretas, além do botão de finalizar a compra estar pequeno demais, afetando a experiência do usuário. Portanto, o estagiário foi requisitado realizar essas correções, e foram feitas com sucesso, como mostrado na Figura 3.9.

Figura 3.8 – Botões com o Padrão de Coloração Incorretas

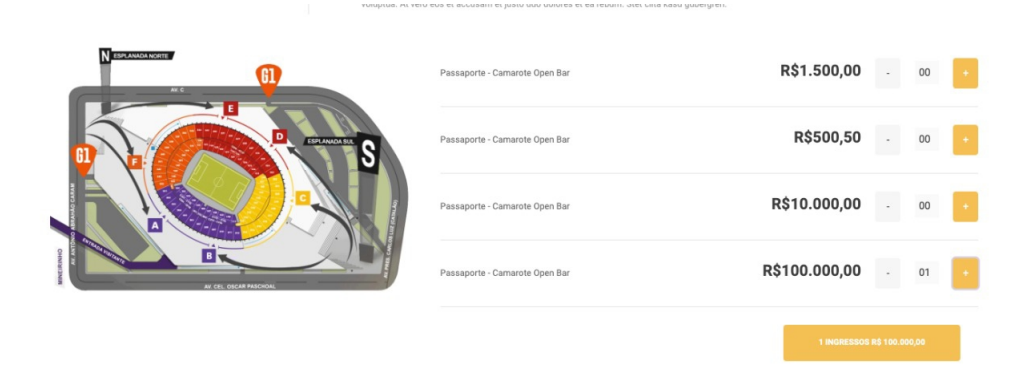
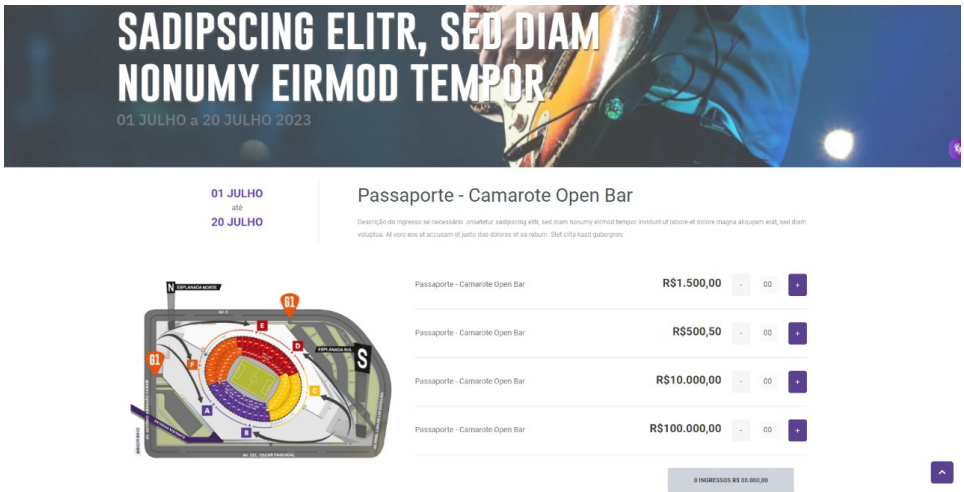


Figura 3.9 – Coloração dos Botões Corrigida



Também foi solicitado para o estagiário realizar tarefas voltadas para o *back-end* como, por exemplo, criar uma maneira de tornar as URLs para os eventos únicas. Previamente, a URL funcionava da forma apresentada na Figura 3.10:

Figura 3.10 – Como as Rotas eram Tratadas Inicialmente

```
Route::get('/selecionar-ingresso', 'getTicketSelect')->name('getTicketSelect');
Route::get('/evento/{p_Name}', 'getEventDetails')->name('getEventDetails');
```

Dessa maneira, havia o risco de conflito caso houvesse mais de um evento com nome idêntico em dias diferentes, situação muito provável de ocorrer. Portanto, primeiramente o estagiário padronizou as duas rotas para iniciarem com */evento*, passou a enviar o nome do evento na rota, assim como a rota *getEventDetails*, além de transformar o nome do evento em *lowercase* para seguir o padrão minúsculo da URL. Como todo evento possui uma data, e um evento com o mesmo nome não aconteceria mais que uma vez no mesmo dia, o estagiário adicionou a data do evento à frente do nome, tratando assim, o possível conflito de rotas.

No entanto, por padrão do sistema, o nome do evento é separado por espaços e não possui uma data à sua frente; outra situação que o estagiário precisou tratar. Quando o sistema recebia o nome do evento (com hífens e a data, e.g *cruzeiro-x-america-2022-01-21*), o estagiário fez uso de duas funções padrões do PHP: a função *preg_replace* e a função *str_replace* (Figura 3.11).

- *preg_replace* realiza uma pesquisa por uma expressão regular e a substitui.
- *str_replace* substitui todas as ocorrências da *string* de procura com a *string* de substituição.

Figura 3.11 – Utilização do *preg replace* e *str replace*

```
$removeDate = preg_replace('/(-\d{4}[-][01]\d[-][0-3]\d)/', '', $p_Name);
$v_Name = str_replace('-', ' ',$removeDate);
```

Logo, ao receber o nome do evento do link do domínio (*pName*), é utilizado a expressão regular apresentada na Figura 3.12:

A expressão regular funciona da maneira representada pela Figura 3.13.

Figura 3.12 – Expressão regular utilizada para retirar a data da URL

```
/(-\d{4}[-][01]\d[-][0-3]\d)/
```

Figura 3.13 – Pseudo-Código explicando Expressão Regular da Figura 3.12

```
1:  i ← 0
2:  while dentro da string do
3:      avançar posições na string até encontrar hífen seguido de um dígito
4:      if primeiro hífen encontrado then
5:          while i < 4 do
6:              busca dígitos inteiros entre 0 e 9
7:              i ← i + 1
8:          end while
9:          substitui dígitos encontrados por vazio
10:     end if
11:     if segundo hífen encontrado then
12:         busca dígito inteiro 0 ou 1
13:         busca dígito inteiro entre 0 e 9
14:         substitui dígitos encontrados por vazio
15:     end if
16:     if terceiro hífen encontrado then
17:         busca dígito inteiro entre 0 e 3
18:         busca dígito inteiro entre 0 e 9
19:         substitui dígitos encontrados por vazio
20:     end if
21: end while
22: substitui hífens por espaços
```

Figura 3.14 – Nome do Evento Antes e Depois do Tratamento da URL

```
antes: cruzeiro-x-america-2022-01-21
depois: cruzeiro x america
```

Outra tarefa também realizada pelo estagiário foi a correção da Agenda e Eventos. Após uma reunião com os representantes do Mineirão, foi apontado que as tabelas de Agenda e Eventos teriam que ser independentes, mas que pudessem se relacionar. Tarefa que foi designada ao estagiário, que inicialmente percebeu que a tabela de Eventos e Agenda eram isoladas (a página do evento criado somente aparecia na página pública caso este evento fosse criado na agenda. Portanto, eventos criados na tabela Eventos eram completamente isolados). A solução implementada pelo estagiário foi tornar a construção da página de eventos dependente da tabela de Eventos ao invés da tabela de Agenda (o estagiário refatorou o código no qual era criado a página, realizando as modificações necessárias e alterando o nome da classe para atender à tabela

de Eventos. Inicialmente a construção era realizada na classe Agenda), e feito isso, o estagiário realizava uma adição à Agenda caso fosse criado um Evento. Então, para que a Agenda pudesse se relacionar à tabela de Eventos, o estagiário permitia os administradores criarem um *pré-evento* na Agenda (apenas uma instância na agenda, e não realmente um evento), e este pré-evento pode ser utilizado diretamente pela tabela de Eventos para criar um evento baseado nas informações inseridas na agenda, sem ser necessário "recriar"aquele evento.

Realizada as tarefas, o estagiário reportava aos integrantes da equipe, os quais revisavam a *branch* da tarefa realizada e realizavam o *merge* para a *branch* principal (*master*) em seguida. Além disso, regularmente eram marcadas as datas que seriam realizados os *deploys* (implantação de uma iteração da aplicação que teve seu desenvolvimento concluído), com os clientes do projeto.

3.2.2 Alpha Transfers

O segundo projeto no qual o estagiário participou foi no desenvolvimento e manuntenção de um aplicativo *mobile* para um grupo de empresas especializadas em serviços financeiros de pagamento e câmbio que formam o grupo Alpha Transfers. O aplicativo do grupo oferece transferências internas entre usuários, câmbio entre diversas moedas e uma carteira digital. Suas transações podem ser feitas para qualquer país que use Real, Libras Esterlinas, Euro ou Franco Suíço. Diferentemente do projeto mencionado anteriormente, o sistema do Alpha Transfers faz uso de uma versão mais antiga do Laravel, o Laravel 5, Vagrant para virtualização do sistema e Android Studio e Xcode como ambientes de desenvolvimento para os sistemas Android e iOS, respectivamente. Além disso, não era mantido apenas um aplicativo, e sim, três:

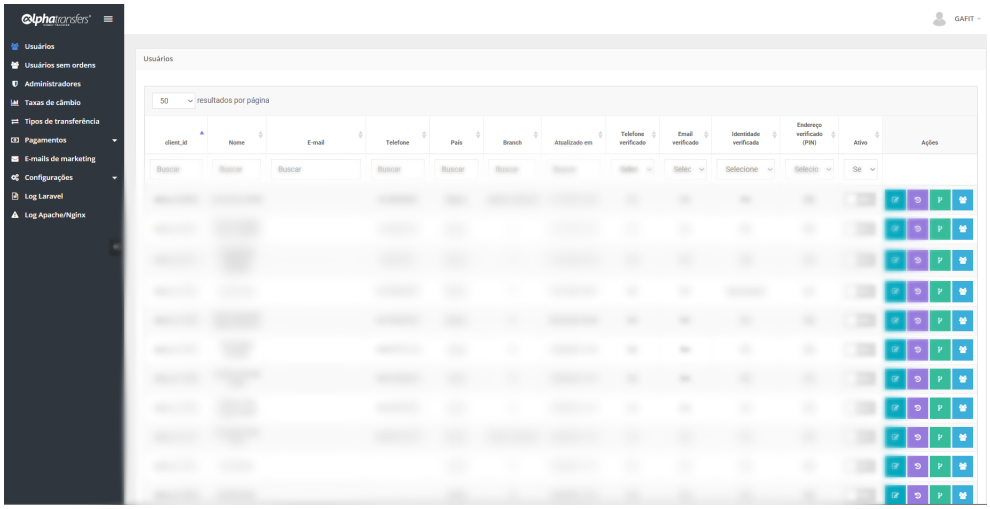
- Alpha Transfers: Aplicativo com foco em transações no Reino Unido.
- Omega Transfers: Aplicativo com foco em transações na Suíça.
- Jubilee Services: Aplicativo com foco em transações na Bélgica.

O motivo da realocação do estagiário foi que o time de desenvolvedores dos aplicativos estava pequeno demais (havia apenas um desenvolvedor ativo), e o projeto estava tendo problemas, portanto, o estagiário foi realocado para dar apoio ao desenvolvedor ativo naquele momento.

Assim como no sistema WEB do Mineirão, o projeto possuia duas frentes:

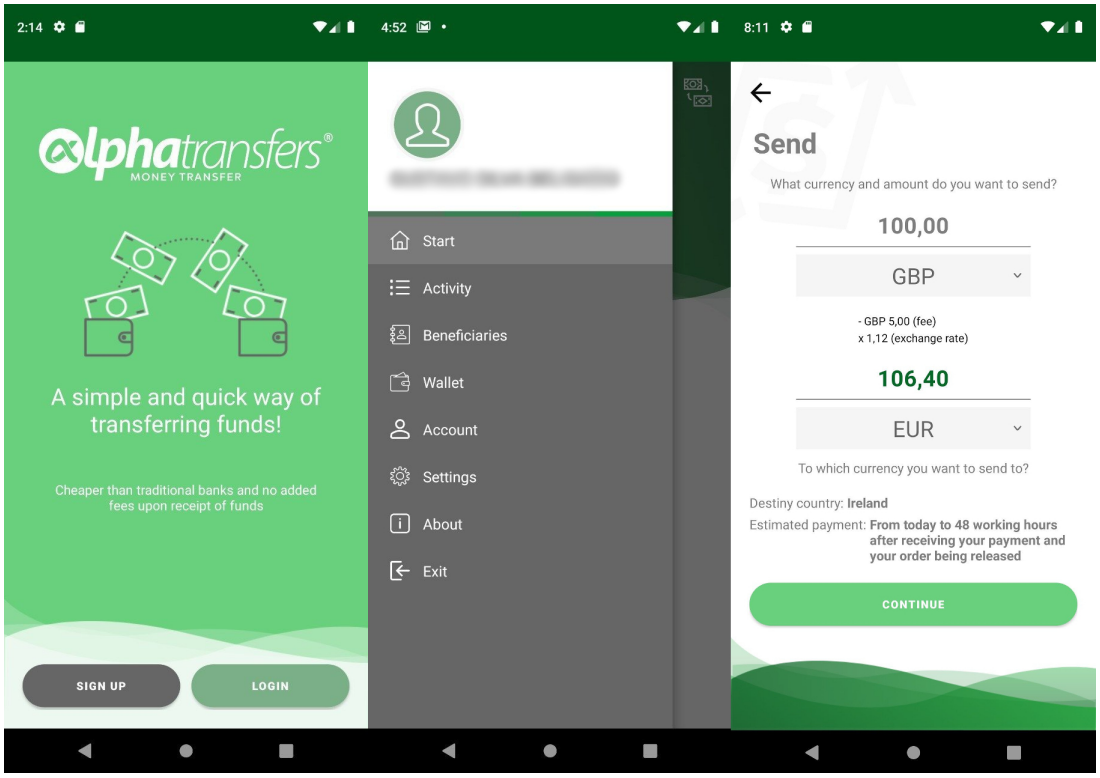
- Uma de acesso exclusivo dos administradores (Figura 3.15), uma dashboard que permite os responsáveis manusear os seus usuários (como ver informações sobre o usuário, pendências de cadastro, transações feitas, beneficiários), taxas de câmbio e pagamentos feitos.

Figura 3.15 – Dashboard Alpha Transfers



- Uma de acesso público (Figura 3.16), um aplicativo para dispositivos móveis (Android e iOS), onde os usuários podem realizar transferências, câmbios e têm acesso à uma carteira digital pessoal.

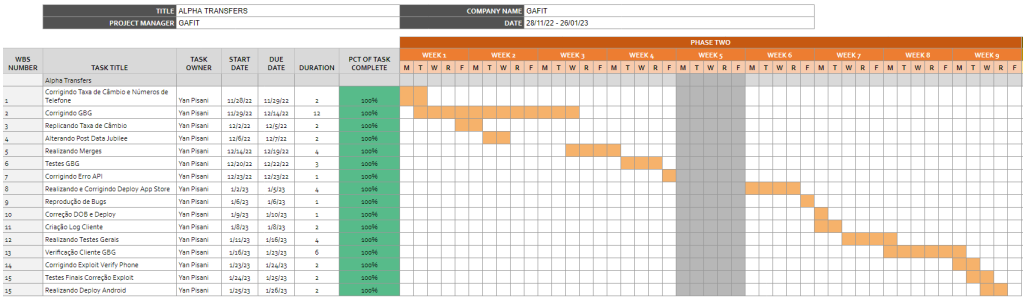
Figura 3.16 – Aplicativo Alpha Tranfers em um Dispositivo *Android*



Fonte: https://gafit.com.br/cases/alpha-transfers

3.2.2.1 Visão Geral das Atividades Realizadas

Figura 3.17 – Gráfico de Gantt do Projeto Alpha Transfers



TITLE	ALPHA TRANSFERS	COMPANY NAME	GAFIT
PROJECT MANAGER	GAFIT	DATE	28/11/22 - 29/01/23

WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	PCT OF TASK COMPLETE
	Alpha Transfers					
1	Corrigindo Taxa de Câmbio e Números de Telefone	Yan Pisani	11/28/22	11/29/22	2	100%
2	Corrigindo GBG	Yan Pisani	11/29/22	12/14/22	12	100%
3	Replicando Taxa de Câmbio	Yan Pisani	12/2/22	12/5/22	2	100%
4	Alterando Post Data Jubilee	Yan Pisani	12/6/22	12/7/22	2	100%
5	Realizando Merges	Yan Pisani	12/14/22	12/19/22	4	100%
6	Testes GBG	Yan Pisani	12/20/22	12/22/22	3	100%
7	Corrigindo Erro API	Yan Pisani	12/23/22	12/23/22	1	100%
8	Realizando e Corrigindo Deploy App Store	Yan Pisani	1/2/23	1/5/23	4	100%
9	Reprodução de Bugs	Yan Pisani	1/6/23	1/6/23	1	100%
10	Correção DOB e Deploy	Yan Pisani	1/9/23	1/10/23	1	100%
11	Criação Log Cliente	Yan Pisani	1/8/23	1/8/23	2	100%
12	Realizando Testes Gerais	Yan Pisani	1/11/23	1/16/23	4	100%
13	Verificação Cliente GBG	Yan Pisani	1/16/23	1/23/23	6	100%
14	Corrigindo Exploit Verify Phone	Yan Pisani	1/23/23	1/24/23	2	100%
15	Testes Finais Correção Exploit	Yan Pisani	1/24/23	1/25/23	2	100%
15	Realizando Deploy Android	Yan Pisani	1/25/23	1/26/23	2	100%

Na Figura 3.17, é apresentado um segundo gráfico de Gantt onde são mostradas todas as atividades no qual o estagiário fez parte durante sua participação no projeto Alpha Transfers. A primeira tarefa do estagiário foi uma correção do valor de taxas de câmbio, onde estava sendo mostrada mais casas decimais em um número em ponto flutuante do que o necessário e alguns números de telefone que estavam obsoletos. A tarefa foi replicada para as outras *branches* do aplicativo. (Linha 1 e 3).

Foram feitas várias tarefas relacionadas às bibliotecas GBG. Desde correções, testes e reprodução de bugs. E também uma tarefa que liberava o cliente para fazer uso do aplicativo após passar pela verificação de identidade. (Linhas 2, 6, 9 e 13). O estagiário também foi requisitado para realizar *merges* e *deploys* dos aplicativos e *back-end* do projeto e testes gerais. (Linhas 5, 8, 12 e 16).

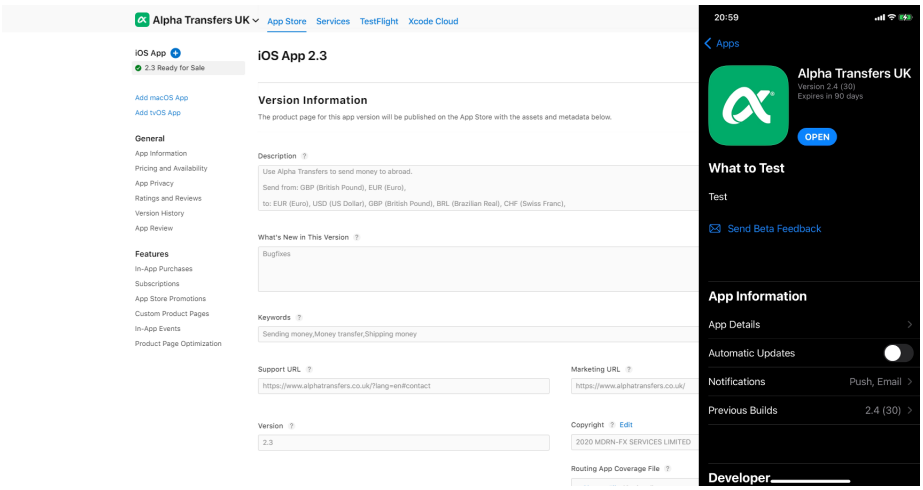
Foi também corrigido um erro de comunicação com a API do cliente, que havia sido atualizada, portanto nossas requisições não estavam enviando todas as informações necessárias, e algumas de forma incorreta (como a data de nascimento do usuário). (Linhas 7 e 10). Também foi realizada a criação de um *log* de cliente que registra todas as interações do usuário com o aplicativo. (Linha 11). Por fim, foi realizada a correção da falha de segurança que permitiu que um usuário realizasse milhares de requisições, e também foram realizados testes dessa correção. (Linhas 14 e 15).

3.2.2.2 Atividades Realizadas

O problema principal do projeto estava relacionado à problemas de compatibilidade devido a um conjunto de bibliotecas para verificação de identidade e detecção de vivacidade da empresa GB Group plc (GBG). A correção foi feita ao reinstalar e reimplementar corretamente a biblioteca. Feita a correção do GBG, foi necessário a correção de algumas telas do iOS que não estavam sendo percorridas corretamente. Durante essas correções, foi necessário que o estagiário trabalhasse juntamente com o cliente, representante do grupo Alpha Transfers, e com os aprendizados do projeto anterior, o estagiário pôde ter iniciativa ao conversar com o cliente, responder as suas dúvidas e acalmá-lo, pois a instalação do GBG foi um problema que levou uma quantidade elevada de tempo para ser resolvida. Por fim, foi pedido ao estagiário para que fosse realizada o merge das mudanças para a *branch* principal.

Após isso, o estagiário ficou responsável pela realização dos testes e *deploy* da aplicação na frente iOS, além de ter um contato mais próximo com o cliente para ser notificado cajo haja problemas, *bugs*, e informar sempre que um *deploy* está pronto para ser realizado ou uma versão nova para ser testada. A empresa Apple possui um sistema próprio para que seja realizado a implantação do aplicativo (App Store Connect). Também incluído um sistema de distribuição de testes para usuários (Testflight), sistema pelo qual o estagiário ficou responsável por manusear para o aplicativo da Alpha Transfers (Figura 3.18).

Figura 3.18 – App Store Connect e Testflight

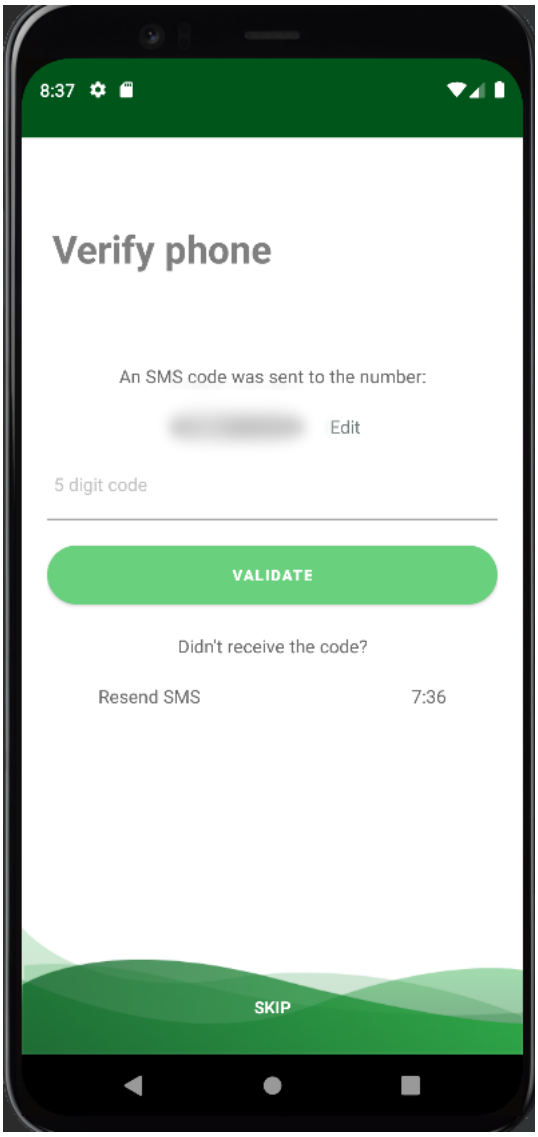


Outra tarefa realizada pelo o estagiário foi a liberação do cliente que passasse na verificação do GBG, para que ele tivesse acesso às funcionalidades dos aplicativos. No banco de dados do sistema, havia uma coluna que marcava se o usuário foi ou não verificado, e era uma coluna que tinha que ser alterada manualmente, pois o sistema de verificação de identidade e detecção de vivacidade não estava funcional. Quando uma verificação era realizada, era recebido um relatório que informava se a verificação validada ou não, portanto, era necessário utilizar essa informação para validar o cliente automaticamente. Pela API RESTful do GBG, era possível encontrar as informações do relatório, e com essas informações, foi possível criar um método para verificação, que ao receber o resultado do relatório, o valor de verificação do usuário era alterado para válido, e ele ficava livre para fazer uso de todas as funcionalidades dos aplicativos.

Houve também um incidente envolvendo a página de registro de usuário (a página requer que o usuário adicione seu número de telefone, para que receba um SMS com um código de verificação, para continuar o registro). O estagiário e seu colega de equipe foram informados pelo cliente sobre um grande volume de requisições em muito pouco tempo (aproximadamente três mil requisições, de um usuário). O cliente não estava ciente de alguma situação na qual seria necessária uma quantidade elevada de requisições (por exemplo, uma empresa de auditoria). Portanto, foi de suma importância que os desenvolvedores buscassem uma solução para impedir esses requerimentos. Na página de verificação do telefone (Figura 3.19), o usuário pode editar o telefone, e há um timer de três minutos, e quando chega a zero, o usuário pode receber um novo código. No entanto, foi descoberto que o consumidor se utilizou de um *exploit* na página, onde, caso o telefone seja editado, um novo código é enviado.

A correção precisou ser feita tanto na frente iOS quanto na frente Android; o estagiário ficou responsável pela frente Android. O desenvolvedor criou um contador no qual era incrementado toda vez que o usuário requisitava um novo código e editava seu número. Este contador multiplicava o tempo da requisição, e enquanto o timer estava ativo, o botão de editar o telefone é desativado. Ao mesmo tempo, foi feita uma correção no *back-end* e também no *front-end* para não permitir dois telefones idênticos.

Figura 3.19 – Tela para Verificação de Telefone



4 CONCLUSÃO

4.1 O Aprendizado

Mesmo com uma bagagem de conhecimento substancial vinda da Universidade, o estagiário teve oportunidade de aprender ainda mais com suas vivências no estágio. Ao ser chamado para o estágio, o aluno tinha apenas conhecimento de linguagens como C++ e Java que foram ensinadas durante seu período de aprendizado na Universidade – e como a principal ferramenta utilizada na GAFIT era a *framework* Laravel, foi necessário que o estagiário aprendesse também a linguagem na qual a ferramenta é baseada, o PHP (enquanto juntamente aprendia como a *framework* funcionava). As vantagens de ter um foco maior para ensinamentos em uma linguagem como C++, é o fortalecimento da base de conhecimento do aluno, e facilita o aprendizado de novas linguagens – como neste caso, a linguagem PHP. Além disso, o estagiário também pôde aprender novas linguagens como JavaScript, utilizada para deixar as páginas nos sistemas WEB mais interativas, Objective-C e Swift para a construção de aplicativos para o sistema operacional iOS, e melhorar seus conhecimentos já adquiridos em HTML/CSS, utilizadas para construir e estilizar as páginas WEB e Java, para criação de aplicativos para o sistema operacional Android.

Outrossim, o estagiário pôde aplicar e aperfeiçoar seus conhecimentos relacionados à bancos de dados (BD), principalmente manuseamento de BDs já existentes – conhecimentos que vieram diretamente de matérias na Universidade como Introdução à Bancos de Dados e Sistemas Gerenciadores de Bancos de Dados. Também pôde-se aplicar os aprendizados de matérias como Engenharia de Software, principalmente devido ao fato que existia uma conexão direta com o cliente, portanto era de suma importância conhecimentos como de gerência de projetos e testes de software.

Ademais, a conexão direta com o cliente trouxe o estagiário para fora de sua zona de conforto, pois foi necessário que o mesmo aperfeiçoasse suas habilidades sociais para conseguir conversar e discutir claramente com o cliente, além de entender seu ponto de vista, ter paciência com quaisquer dúvidas e preocupações, e atender o cliente de uma maneira que o mesmo saia satisfeito após a interação. Além disso, o estagiário pôde melhorar conjuntamente sua aptidão para trabalhar em equipe, necessária para trabalhar em projetos com seus colegas de trabalho.

4.2 Conclusões Gerais

Diante do exposto, pode-se observar que o estagiário pôde aplicar conhecimentos adquiridos durante o curso, além de aperfeiçoar habilidades já existentes e aprender novas, e principalmente ter um contato direto com clientes em projetos. O curso de Ciência da Computação possibilitou que o estagiário pudesse cumprir suas tarefas confortavelmente, e que ele tivesse flexibilidade para realizar diversos tipos, incluindo tarefas relacionadas à *front-end*, *back-end* e também bancos de dados.

Uma das principais dificuldades enfrentadas pelo estagiário foi o *multitasking*, um requisito durante o período do estágio. Os desenvolvedores precisam atuar em vários projetos diferentes enquanto simultaneamente atendem os clientes – respondem suas dúvidas e preocupações, realizam alterações no projeto, mudanças de prazo – e isto impõe pressão no desenvolvedor. Outra dificuldade enfrentada foi que a introdução do estagiário à empresa teve um foco maior no estudo de documentações das ferramentas utilizadas e vídeo-aulas, e não tanto nos processos da empresa e padrões de projeto.

Para tornar os processos da companhia mais eficazes e ter uma diminuição no acúmulo de funções de seus desenvolvedores sugere-se a contratação de profissionais cuja função seja atuar diretamente com os clientes (ou apenas designar integrantes já existentes), buscar a criação de um departamento com foco em *Quality Assurance* (QA) para que os produtos sejam entregues com a qualidade exigida aos clientes e, além disso, também ter um foco especial na introdução de novos integrantes aos processos da empresa, além dos estudos.

Entretanto, o fato da empresa ser pequena, e a utilização de uma ferramenta como o Gather, para a criação de um escritório virtual, proporcionou uma conexão mais pessoal com seus colegas de trabalho, e também o próprio gerente da empresa, deixando o estagiário mais seguro para sanar suas dúvidas gerais, discutir sobre outros projetos, e até dar ideias sobre novos, além de serem bem flexíveis, bem compreensivos com novos e velhos integrantes. Um ambiente familiar no qual o estagiário pôde trabalhar com conforto e à vontade, onde pôde aplicar diretamente seus conhecimentos adquiridos ao longo de sua carreira acadêmica, e adicionado ainda mais para sua bagagem.

REFERÊNCIAS

APPLE INC. **Programming with Objective-C**. [S.l.], 2014. Disponível em: <https://developer.apple.com/library/archive/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/Introduction/Introduction.html#//apple_ref/doc/uid/TP40011210>.

APPLE INC. **Swift**. [S.l.], 2014. 1 p. Disponível em: <https://carlosicaza.com/swiftbooks/SwiftLanguage.pdf>.

APPLE INC. **The Swift Programming Language**. [S.l.], 2022. Disponível em: <https://docs.swift.org/swift-book/>.

APPLE INC. **Xcode Documentation**. [S.l.], 2023. Disponível em: <https://developer.apple.com/documentation/xcode>.

CHAGAS, C. E. das; BARUQUE, C. B.; BARUQUE, L. B. **Java Básico e Orientação a Objeto**. 1. ed. [S.l.: s.n.], 2010.

CLICKUP. **Why Should You Use ClickUp?** [S.l.], 2020. Disponível em: <https://clickup.com/blog/why-clickup/>.

CLICKUP. **What makes ClickUp different from other Project Management products?** [S.l.], 2023. Disponível em: <https://help.clickup.com/hc/en-us/articles/6311918951447-What-makes-ClickUp-different-from-other-Project-Management-products-#:~:text=ClickUp%20empowers%20you%20to%20manage,%2C%20spreadsheets%2C%20and%20mind%20maps.>

DAVID J ANDERSON SCHOOL OF MANAGEMENT. **Benefits of Kanban**. [S.l.], 2021. Disponível em: <https://djaa.com/wp-content/uploads/2021/06/Benefits-of-Kanban-3.pdf>.

DOCKER INC. **Docker Docs - Docker storage drivers**. [S.l.], 2023. Disponível em: <https://docs.docker.com/storage/storagedriver/select-storage-driver/>.

DOCKER INC. **Docker Docs - Overview**. [S.l.], 2023. Disponível em: <https://docs.docker.com/get-started/overview/>.

FITRIA, T. N. Creating sensation of learning in classroom: Using 'gather town' platform video game-style for virtual classroom. **Education and Human Development Journal**, v. 6, n. 2, p. 30–43, 2021. Disponível em: <https://journal2.unusa.ac.id/index.php/EHDJ/article/view/2106/1497>.

FLANAGAN, D. **JavaScript: The Definitive Guide**. 7. ed. [S.l.: s.n.], 2020.

GITLAB INC. **DevOps Tools**. [S.l.], 2023. Disponível em: <https://about.gitlab.com/handbook/marketing/brand-and-product-marketing/product-and-solution-marketing/devops-tools/>.

GITLAB INC. **What is GitLab?** [S.l.], 2023. Disponível em: <https://about.gitlab.com>.

GITLAB INC. **Why GitLab?** [S.l.], 2023. Disponível em: <https://about.gitlab.com/why-gitlab/>.

GOOGLE DEVELOPERS. **Android Studio User Guide**. [S.l.], 2022. Disponível em: <https://developer.android.com/studio/intro>.

HASHICORP. **Introduction to Vagrant**. [S.l.], 2023. Disponível em: <https://developer.hashicorp.com/vagrant/intro>.

HASHICORP. **Vagrant Documentation - Vagrantfile**. [S.l.], 2023. Disponível em: <https://developer.hashicorp.com/vagrant/docs/vagrantfile>.

HAWB, S. **Mastering Laravel Eloquent ORM**. [s.n.], 2020. Disponível em: <https://dev.to/xenoxdev/mastering-laravel-eloquent-orm-the-eloquent-journey-part-1-1571>.

JETBRAINS. **PhpStorm - Quick Start Guide**. [S.l.], 2022. Disponível em: <https://www.jetbrains.com/help/phpstorm/quick-start-guide-phpstorm.html>.

JETBRAINS. **PhpStorm Features**. [S.l.], 2023. Disponível em: <https://www.jetbrains.com/phpstorm/features/>.

KANBAN UNIVERSITY. **O Guia Oficial do Método Kanban**. [S.l.], 2021. Disponível em: <https://kanban.university/wp-content/uploads/2021/04/The-Official-Kanban-Guide_Portuguese_A4.pdf>.

LARAVEL. **Laravel Documentation**. [S.l.], 2022. Disponível em: <https://laravel.com/docs/9.x/readme>.

MICROSOFT. **Comparing WSL Versions**. [S.l.], 2022. Disponível em: <https://learn.microsoft.com/en-us/windows/wsl/compare-versions#whats-new-in-wsl-2>.

MICROSOFT. **Frequently Asked Questions about Windows Subsystem for Linux**. [S.l.], 2022. Disponível em: <https://learn.microsoft.com/en-us/windows/wsl/faq>.

MOZILLA FOUNDATION. **CSS: Cascading Style Sheets - MDN Web Docs**. [S.l.], 2023. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/CSS>.

MOZILLA FOUNDATION. **HTML: HyperText Markup Language - MDN Web Docs**. [S.l.], 2023. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/HTML>.

MOZILLA FOUNDATION. **JavaScript - MDN Web Docs**. [S.l.], 2023. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>.

MOZILLA FOUNDATION. **What is JavaScript - MDN Web Docs**. [S.l.], 2023. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction#what_is_javascript>.

MUDRAKOLA, S. **What is ClickUp?** [s.n.], 2022. Disponível em: <https://petri.com/what-is-clickup/>.

MYSQL. **MySQL Workbench Repository**. [S.l.], 2023. Disponível em: <https://github.com/mysql/mysql-workbench>.

ORACLE. **MySQL 8.0 Reference Manual - Including MySQL NDB Cluster 8.0**. [S.l.], 2023. 4 - 6 p. Disponível em: <https://downloads.mysql.com/docs/refman-8.0-en.pdf>.

ORACLE. **MySQL Workbench**. [S.l.], 2023. 1 p. Disponível em: <https://downloads.mysql.com/docs/workbench-en.pdf>.

SCHILDT, H. **Java - A Beginner's Guide**. 8. ed. [S.l.: s.n.], 2019. 14-15 p.

SCHWABER, K.; SUTHERLAND, J. **The Scrum Guide - The Definitive Guide to Scrum: The Rules of the Game**. [s.n.], 2020. Disponível em: <https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf>.

THE PHP GROUP. **PHP Documentation**. [S.l.], 2023. Disponível em: <https://www.php.net/manual/en/>.

THOMSON, L.; WELLING, L. **PHP and MYSQL Web Development**. 5. ed. [S.l.: s.n.], 2017. 4-6 p.

W3C. **HTML and CSS - W3C**. [S.l.], 2010. Disponível em: <https://web.archive.org/web/20101129081921/https://www.w3.org/standards/webdesign/htmlcss#whatcss>.