INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
CAMPUS CAJAZEIRAS
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

RELATO DE EXPERIÊNCIA PROFISSIONAL EM DESENVOLVIMENTO DE SOLUÇÕES DE SOFTWARE NA EMPRESA CWI

RAILSON MATEUS DE SOUSA SILVA

Cajazeiras
2025

RAILSON MATEUS DE SOUSA SILVA

RELATO DE EXPERIÊNCIA PROFISSIONAL EM DESENVOLVIMENTO DE SOLUÇÕES DE SOFTWARE NA EMPRESA CWI

Trabalho de Conclusão de Curso apresentado junto ao Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia da Paraíba - Campus Cajazeiras, como requisito à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador

Prof. Me. Francisco Paulo de Freitas Neto.

Cajazeiras
2025

IFPB / Campus Cajazeiras
Coordenação de Biblioteca
Biblioteca Prof. Ribamar da Silva
Catalogação na fonte: Cícero Luciano Félix CRB-15/750

S586r Silva, Railson Mateus de Sousa.

Relato de experiência profissional em desenvolvimento de soluções de software na empresa CWI / Railson Mateus de Sousa Silva. – 2025.
27f. : il.

Trabalho de Conclusão de Curso (Tecnólogo em Análise e Desenvolvimento de Sistemas) - Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, Cajazeiras, 2025.

Orientador: Prof. Me. Francisco Paulo de Freitas Neto.

1. Engenharia de software. 2. Desenvolvimento de software. 3. Tecnologia da informação. 4. Relato de experiência. I. Instituto Federal de Educação, Ciência e Tecnologia da Paraíba. II. Título.

IFPB/CZ CDU: 004.42(043.2)

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA

RAILSON MATEUS DE SOUSA SILVA

RELATO DE EXPERIÊNCIA PROFISSIONAL NA EMPRESA CWI NO DESENVOLVIMENTO DE SOLUÇÕES DE SOFTWARE

Trabalho de Conclusão de Curso apresentado junto ao Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia da Paraíba - Campus Cajazeiras, como requisito à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador

Prof. Me. Francisco Paulo de Freitas Neto

Aprovada em: **09 de setembro de 2025.**

Prof. Me. Francisco Paulo de Freitas Neto - Orientador

Prof. Me. Janderson Ferreira Dutra - Avaliador

IFPB - Campus Cajazeiras

Profa. Dra. Eva Maria Campos Pereira - Avaliadora

IFPB - Campus Cajazeiras

Documento assinado eletronicamente por:

- **Francisco Paulo de Freitas Neto**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 10/09/2025 16:15:34.
- **Janderson Ferreira Dutra**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 10/09/2025 18:28:14.
- **Eva Maria Campos Pereira**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 11/09/2025 10:21:26.

Este documento foi emitido pelo SUAP em 10/09/2025. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código 764398
Verificador: 49c0653abe
Código de Autenticação:

AGRADECIMENTOS

Agradeço em primeiro lugar, agradeço a Deus por me guiar e me fortalecer nos momentos mais desafiadores, por me conceder dons e por me fazer acreditar em mim mesmo.

Aos meus familiares e amigos, sou imensamente grato por seu constante apoio e por acreditarem em meu potencial. Sem o encorajamento e o suporte de vocês, eu não teria chegado até aqui.

Quero expressar minha sincera gratidão ao meu orientador, Paulo Freitas, por seu contínuo incentivo, orientação e por me desafiar a ir além do esperado. Sua expertise e dedicação foram fundamentais para o sucesso deste tcc.

Este trabalho é dedicado a cada um de vocês, cujo apoio, orientação e contribuições foram fundamentais para minha trajetória acadêmica e para o sucesso deste projeto. Minha gratidão é imensa, e espero poder retribuir de alguma forma no futuro.

RESUMO

Este trabalho apresenta um relato da experiência profissional do autor durante o estágio na empresa CWI Software, com atuação na área de desenvolvimento e manutenção de soluções digitais para clientes corporativos. São descritas as responsabilidades assumidas, as tecnologias utilizadas e os desafios enfrentados, bem como os aprendizados adquiridos ao longo do período de estágio. O objetivo é compartilhar a vivência prática no ambiente corporativo e apresentar a relação entre os conhecimentos acadêmicos e as demandas do mundo do trabalho.

Palavras-chave: Estágio. Desenvolvimento de Software. CWI. Tecnologias. Experiência Profissional.

ABSTRACT

This work presents an account of the author's professional experience during an internship at CWI Software, working in the area of development and maintenance of digital solutions for corporate clients. It describes the responsibilities undertaken, the technologies used, and the challenges faced, as well as the lessons learned throughout the internship period. The objective is to share the practical experience in a corporate environment and highlight the connection between academic knowledge and the demands of the professional world.

Keywords:Internship. Software Development. CWI. Technologies. Professional Experience.

LISTA DE FIGURAS

Figura 1 – Diagrama das camadas da Clean Architecture. 20
Figura 2 – Eventos do Scrum 23

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
ADS	Análise e Desenvolvimento de Sistemas
AOT	Ahead-Of-Time
Bloc	Business Logic Component
CI/CD	Continuous Integration / Continuous Delivery
CSS	Cascading Style Sheets
DIP	Dependency Inversion Principle
HTML	HyperText Markup Language
IFPB	Instituto Federal de Educação, Ciência e Tecnologia da Paraíba
ISP	Interface Segregation Principle
LSP	Liskov Substitution Principle
NBR	Norma Brasileira
OCP	Open/Closed Principle
SRP	Single Responsibility Principle
TCC	Trabalho de Conclusão do Curso
UI	User Interfaces

SUMÁRIO

1 INTRODUÇÃO 15

1.1 OBJETIVOS 16

1.1.1 Objetivo Geral 16

1.1.2 Objetivos Específicos 16

1.2 APRESENTAÇÃO DA EMPRESA 16

1.3 ORGANIZAÇÃO DO DOCUMENTO 17

2 FUNDAMENTAÇÃO TEÓRICA 18

2.1 Flutter: Um Framework Multiplataforma 18

2.2 Clean Architecture 19

2.3 Gerenciamento de Estado: Bloc e Cubit 20

2.3.1 Comparação entre Bloc e Cubit 21

2.4 Princípios SOLID 21

2.5 Plataformas de Controle de Versão e Integração Contínua 22

2.6 Metodologia Ágil Scrum 23

2.6.1 Papéis no Scrum 23

2.6.2 Eventos no Scrum 23

2.6.3 Artefatos do Scrum 24

3 RELATO DE EXPERIÊNCIA 25

4 CONCLUSÃO 29

REFERÊNCIAS 30

1 INTRODUÇÃO

O avanço da transformação digital tem imposto às organizações a adoção contínua de tecnologias emergentes e práticas ágeis, visando o desenvolvimento de soluções de software escaláveis, robustas e de alta qualidade. Segundo a pesquisa PINTEC ((ABDI), 2022), cerca de 85% das empresas brasileiras de médio e grande porte já utilizam tecnologias digitais avançadas, com destaque para computação em nuvem (73,6%), Internet das Coisas (46,8%), robótica (27,7%) e big data (23,4%). Além disso, aproximadamente 57% dessas empresas adotaram frameworks ágeis como parte de seus esforços de transformação digital (CREATE, 2022).

Nesse cenário, destaca-se a CWI Software, fundada em 1991, uma das principais provedoras de soluções digitais no Brasil, que atualmente conta com mais de 1.500 colaboradores e mantém unidades em São Paulo, Porto Alegre e São Leopoldo, consolidando sua atuação em diversos setores industriais com foco em inovação e excelência técnica. (CWI Software, 2025b)

Este trabalho apresenta um relato de experiência acerca do estágio iniciado em março de 2025 na CWI Software. O objetivo principal do time ao qual o autor está alocado é promover a melhoria contínua do aplicativo em desenvolvimento, por meio da adoção de boas práticas de engenharia de software, aprimoramento arquitetural, legibilidade de código e implementação de novas funcionalidades. Especificamente, o autor atua como desenvolvedor mobile voltado ao frontend, utilizando Flutter e aplicando conceitos de Clean Architecture, Bloc, Cubit e princípios SOLID (ABZAL, 2025).

As atividades são realizadas de forma remota, apoiadas por plataformas como GitLab e Azure DevOps para versionamento e integração contínua. A organização do trabalho segue o framework ágil Scrum, reconhecido como um dos principais modelos de gestão de projetos iterativos, com cerimônias estabelecidas como plannings, reviews e dailies, conforme descrito no Scrum Guide (SCHWABER; SUTHERLAND, 2020a). O estágio tem sido marcado por um percurso de aprendizado acelerado, especialmente diante do primeiro contato com o Flutter, superado com o apoio do padrinho de estágio. Essa vivência tem contribuído para o desenvolvimento técnico e o entendimento dos processos reais de mercado, fortalecendo a formação profissional do autor.

Nesse contexto, o estágio revela-se fundamental para consolidar a relação entre os conhecimentos teóricos adquiridos na graduação e as exigências práticas do mercado de trabalho, fortalecendo habilidades de adaptação, colaboração e resolução de problemas em ambientes corporativos.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Relatar a experiência adquirida durante o estágio de 10 meses na CWI Software, com ênfase nas atividades desempenhadas, nas tecnologias utilizadas, nos desafios enfrentados e nos aprendizados obtidos ao longo desse período.

1.1.2 Objetivos Específicos

1. Apresentar a CWI Software, seu contexto de atuação e o papel desempenhado no mercado de desenvolvimento de software.
2. Descrever as atividades desenvolvidas durante o estágio, com foco na implementação de melhorias e novas funcionalidades no aplicativo.
3. Apresentar as tecnologias e metodologias utilizadas, como Flutter, Clean Architecture, Bloc, Cubit, GitLab, Azure DevOps e Scrum, destacando suas aplicações práticas.
4. Relatar os desafios enfrentados, incluindo o aprendizado inicial de Flutter, e como o acompanhamento do padrinho de estágio contribuiu para o desenvolvimento técnico.
5. Analisar os aprendizados adquiridos e sua importância para a inserção no mercado de trabalho e para o aprimoramento profissional.

1.2 APRESENTAÇÃO DA EMPRESA

A **CWI Software** é uma empresa brasileira de tecnologia da informação, especializada em engenharia de software sob medida, fundada em 1991. Com sedes em Porto Alegre, São Leopoldo (no parque tecnológico Tecnosinos) e São Paulo, a empresa atua tanto no mercado nacional quanto internacional, consolidando-se como referência em soluções digitais para setores como financeiro, varejo, saúde, educação e tecnologia (CWI Software, 2025b).

Com aproximadamente 1.500 colaboradores — os “cwisers” — distribuídos pelo país, a CWI possui cultura organizacional marcada pela valorização da colaboração, excelência técnica, adaptabilidade e transparência em todas as suas ações (CWI Software, 2025b). Sua trajetória inclui a ampliação contínua da estrutura, com destaque para a inauguração da unidade de São Paulo em 1997, a inserção no Tecnosinos em 2002, e a conquista da certificação CMMI, atestando a adoção de boas práticas em processos de desenvolvimento de software (RENNER, 2025).

A empresa oferece uma gama diversificada de serviços, incluindo squads ágeis, outsourcing, sustentação, desenvolvimento de produtos digitais, e-commerces e APIs. Entre os modelos

de atuação, destacam-se as Fábricas de Software, Fábricas de Projetos, Fábricas de Testes e o modelo de Outsourcing, permitindo flexibilidade e customização conforme a demanda do cliente (Tecnosinos, 2025).

A CWI também desenvolve programas internos para formação e captação de talentos. Entre eles, o *CWI Crescer*, lançado em 2009, direcionado à capacitação de estudantes de TI por meio de treinamentos práticos em formato progressivo; e mais recentemente o *CWI Reset*, voltado para profissionais em transição de carreira (CWI Software, 2025a).

Na prática, a CWI tem se destacado pelo compromisso com a transformação digital de seus clientes, entregando soluções estratégicas baseadas em tecnologia de ponta e adaptadas às particularidades de cada setor. Esse foco na qualidade e inovação técnica sustenta sua reputação como uma das principais empresas de engenharia de software do Brasil.

1.3 ORGANIZAÇÃO DO DOCUMENTO

Este trabalho está organizado em cinco capítulos, estruturados de forma a apresentar de maneira clara e objetiva a experiência adquirida durante o estágio na CWI Software.

No Capítulo 2, é exposta a fundamentação teórica, abordando conceitos, metodologias e tecnologias utilizadas durante o estágio, como Flutter, Clean Architecture, Bloc, Cubit, princípios SOLID, GitLab, Azure DevOps e Scrum.

O Capítulo 3 descreve o relato de experiências, detalhando as atividades desempenhadas pelo autor, os desafios enfrentados, os aprendizados obtidos e a relação entre os conhecimentos acadêmicos e as práticas adotadas no ambiente corporativo.

Por fim, o Capítulo 4 apresenta as considerações finais sobre o estágio, destacando as competências desenvolvidas, os impactos na formação profissional e as perspectivas para a continuidade da carreira.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os conceitos, metodologias, tecnologias e práticas utilizadas durante o estágio na CWI Software, fundamentais para a compreensão do contexto e da execução das atividades desenvolvidas.

2.1 FLUTTER: UM FRAMEWORK MULTIPLATAFORMA

O **Flutter** é um framework de interface de usuário (UI) open-source desenvolvido pelo Google, lançado em maio de 2017, que permite a construção de aplicações multiplataforma (mobile, web e desktop) a partir de uma única base de código (AUNG et al., 2024). Ele utiliza a linguagem Dart[1], também criada pelo Google, que é compilada diretamente para código nativo por meio de compilação *Ahead-Of-Time* (AOT), proporcionando elevado desempenho e menor latência, em comparação com outras soluções híbridas que dependem de bridges entre linguagens (VääNäNEN, 2025).

A renderização de interface no Flutter é realizada por meio do motor gráfico Skia, com cada widget sendo desenhado diretamente pelo framework, não dependendo dos componentes nativos da plataforma, o que assegura desempenho fluido e experiência visual consistente em 60 fps em dispositivos compatíveis. Adicionalmente, o recurso de *Hot Reload* permite visualizar alterações de código em tempo real sem reiniciar o aplicativo ou perder seu estado atual, acelerando drasticamente o ciclo de desenvolvimento (JEDYNASTY, 2025).

Outra funcionalidade marcante do Flutter é o recurso *Hot Reload*, que possibilita visualizar alterações de código quase instantaneamente em um aplicativo em execução, sem perda do estado atual ou necessidade de recompilação completa. Esse recurso acelera significativamente o ciclo de desenvolvimento, favorecendo a experimentação e a correção eficiente de bugs (RADADIYA, 2025).

Dentre os principais benefícios do uso do Flutter em projetos corporativos, destacam-se:

- **Agilidade no desenvolvimento**: unificação da base de código para múltiplas plataformas reduz o esforço de desenvolvimento e manutenção (JEDYNASTY, 2025).
- **Performance comparável ao nativo**: renderização própria e compilação AOT garantem fluidez e baixo overhead em tempo de execução (TOLLIN; LIDEKRANS, 2023).

[1] Dart é uma linguagem de programação criada pelo Google em 2011, com foco em produtividade de desenvolvedores e desempenho. É uma linguagem orientada a objetos, com sintaxe semelhante a C, e pode ser compilada tanto para JavaScript quanto para código nativo através de AOT (Ahead-of-Time) e JIT (Just-in-Time).

- **Consistência visual e comportamental**: o uso de widgets padronizados (Material e Cupertino) assegura uniformidade nas interfaces entre plataformas (EFENDI, 2025).
- **Ecossistema ativo e evolução contínua**: sua natureza open-source combinada com o apoio do Google fomentam um amplo repositório de pacotes, plugins e atualizações constantes (EFENDI, 2025).

O Algoritmo 1 apresenta um exemplo clássico de *Hello World* em Flutter, ilustrando a estrutura mínima de um aplicativo:

Algoritmo 1 – Hello World em Flutter

```dart
import 'package:flutter/material.dart';

void main() {
  runApp(const HelloWorldApp());
}

class HelloWorldApp extends StatelessWidget {
  const HelloWorldApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Hello, World',
      home: Scaffold(
        body: Center(
          child: Text(
            'Hello, World!',
            style: Theme.of(context).textTheme.headlineSmall,
          ),
        ),
      ),
    );
  }
}
```

Fonte: Elaborado pelo autor

Dessa forma, o Flutter apresenta-se como solução estratégica em projetos que demandam escalabilidade, alta performance, rápida entrega e manutenção contínua, alinhando produtividade à qualidade e experiência do usuário final.

2.2 CLEAN ARCHITECTURE

A **Clean Architecture**, proposta por Robert C. Martin (conhecido como Uncle Bob), organiza o sistema em camadas concêntricas que separam regras de negócio, casos de uso,

adaptadores de interface e infraestrutura externa, promovendo independência de frameworks, testabilidade e manutenção facilitada (MARTIN, 2018).

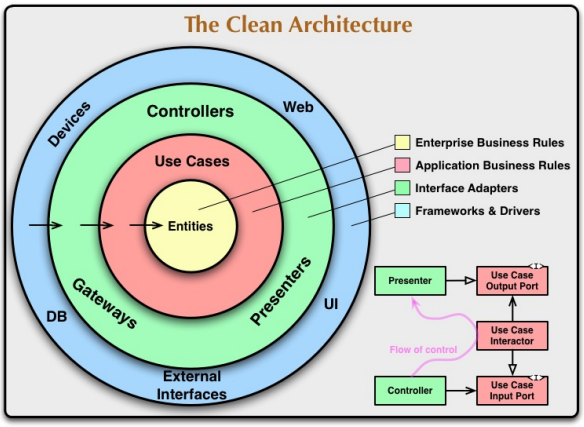

Figura 1 – Diagrama das camadas da Clean Architecture. Fonte: (MARTIN, 2012)

No centro, a camada **Entities** representa as regras de negócio fundamentais e independentes de contexto. Ao redor, a camada de **Use Cases** define os comportamentos específicos da aplicação. As camadas mais externas, como **Interface Adapters** e **Frameworks & Drivers**, lidam com detalhes externos como interface, banco de dados e integrações (MORIN, 2025).

A essência dessa arquitetura está na *Regra de Dependência*: as dependências devem apontar apenas para dentro, evitando acoplamento de regras centrais com detalhes externos ou frameworks instáveis (NDepend Team, 2025).

Esta abordagem traz diversas vantagens para sistemas corporativos, conforme aponta (MARTIN, 2018):

- **Manutenibilidade aprimorada**: a separação clara de responsabilidades permite mudanças localizadas sem impactar todo o sistema.
- **Elevada testabilidade**: regras de negócio podem ser testadas isoladamente, sem dependência de elementos externos como **user interfaces (UI)** ou banco de dados .
- **Flexibilidade tecnológica**: alterações em frameworks ou infraestrutura não afetam a lógica central, que se mantém intacta.
- **Organização de código sustentável**: cada camada tem responsabilidade bem definida, facilitando o desenvolvimento e onboarding de novos membros da equipe .

2.3 GERENCIAMENTO DE ESTADO: BLOC E CUBIT

No desenvolvimento de aplicativos Flutter, o gerenciamento de estado é um componente essencial para orquestrar a comunicação entre a interface do usuário e a lógica de negócios.

Nesse contexto, destacam-se duas abordagens oferecidas pela biblioteca *flutter_bloc*: o padrão **Bloc** (Business Logic Component) e sua versão simplificada, o **Cubit**.

O **Bloc** segue uma abordagem baseada em eventos e estados: o usuário ou sistema dispara um evento, que é processado pelo Bloc, resultando em um novo estado emitido para a interface. Isso promove previsibilidade, testabilidade e facilidade de manutenção, especialmente em aplicações complexas com múltiplas interações de usuário e fluxos de dados encadeados (OKTAY, 2023; RANGEL, 2023; BIBEROGLU, 2023).

Por outro lado, o **Cubit** oferece uma alternativa mais direta, sem a necessidade de trabalhar explicitamente com eventos. Nesse modelo, métodos são invocados diretamente para emitir novos estados, reduzindo o código boilerplate e acelerando o desenvolvimento em cenários de menor complexidade (MARQUES, 2023; STACKMARKET, 2024).

2.3.1 Comparação entre Bloc e Cubit

De acordo com a literatura recente, as principais diferenças podem ser resumidas da seguinte forma:

- O **Cubit** promove simplicidade e clareza, facilitando a separação das responsabilidades e tornando o código mais legível e testável (PAVITHRAN, 2023).
- O **Bloc** destaca-se por sua robustez, permitindo lidar com lógicas mais complexas e oferecendo maior controle sobre o fluxo de eventos (BIBEROGLU, 2023).
- A escolha entre os dois depende do escopo do projeto: enquanto o Cubit é recomendado para casos mais simples ou prototipagem rápida, o Bloc proporciona arquiteturas mais estruturadas e rastreáveis (STACKMARKET, 2024).

2.4 PRINCÍPIOS SOLID

Os princípios **SOLID**, introduzidos por Robert C. Martin (também conhecido como *Uncle Bob*), representam um conjunto de boas práticas voltadas para o desenvolvimento de software orientado a objetos, com o objetivo de tornar o código mais modular, reutilizável, flexível e de fácil manutenção (MARTIN, 2003). O acrônimo **SOLID** engloba cinco princípios fundamentais:

- **S — Single Responsibility Principle (SRP):** Cada classe ou módulo deve ter apenas uma responsabilidade, ou seja, deve ser responsável por uma única funcionalidade do sistema. Isso reduz o acoplamento e facilita a manutenção do código.

- **O — Open/Closed Principle (OCP):** Os módulos devem estar “abertos” para extensão, mas “fechados” para modificação. Novas funcionalidades devem ser adicionadas sem alterar o comportamento já existente.
- **L — Liskov Substitution Principle (LSP):** Objetos de uma classe derivada devem poder substituir objetos da classe base sem que o comportamento do sistema seja alterado. Esse princípio assegura a consistência e a previsibilidade do sistema.
- **I — Interface Segregation Principle (ISP):** Interfaces devem ser específicas e coesas, evitando que os clientes sejam forçados a implementar métodos que não utilizam.
- **D — Dependency Inversion Principle (DIP):** Os módulos de alto nível não devem depender diretamente de módulos de baixo nível, mas sim de abstrações. Isso promove o desacoplamento e facilita a integração de diferentes componentes do sistema.

2.5 PLATAFORMAS DE CONTROLE DE VERSÃO E INTEGRAÇÃO CONTÍNUA

O desenvolvimento de software moderno demanda o uso de ferramentas que viabilizem colaboração, rastreabilidade e automação de processos. Entre as soluções mais utilizadas nesse contexto destacam-se o **GitLab** e o **Azure DevOps**, plataformas que integram recursos de controle de versão, gerenciamento de projetos e integração e entrega contínua (*CI/CD*).

O **GitLab** é uma plataforma de desenvolvimento colaborativo baseada no sistema de controle de versão Git, criada em 2011. Além de hospedar repositórios, o GitLab fornece um ambiente completo para automação de *pipelines*, revisões de código, monitoramento de qualidade e gerenciamento de fluxos de trabalho. Sua principal característica é possibilitar a integração de todas as etapas do ciclo de vida do software em um único sistema, seguindo os princípios da abordagem *DevOps* (GitLab Inc., 2024).

Já o **Azure DevOps**, desenvolvido pela Microsoft, é um conjunto de serviços voltados para a gestão do ciclo de desenvolvimento, oferecendo funcionalidades como controle de versão, gerenciamento de tarefas, integração contínua e entrega automatizada. Sua principal vantagem é a capacidade de integrar equipes e ferramentas em um ambiente corporativo, permitindo maior alinhamento entre desenvolvedores, analistas e operações. Além disso, o Azure DevOps suporta metodologias ágeis, como Scrum e Kanban, facilitando a organização do trabalho e a priorização de entregas (Microsoft Corporation, 2024).

Ambas as ferramentas desempenham um papel essencial no contexto atual de desenvolvimento de software, permitindo maior produtividade, rastreabilidade e qualidade nas entregas, além de possibilitar uma colaboração mais eficiente entre equipes distribuídas.

2.6 METODOLOGIA ÁGIL SCRUM

O **Scrum** é um framework ágil para gerenciamento e desenvolvimento de produtos complexos, proposto inicialmente por (SUTHERLAND, 2010) e amplamente utilizado no desenvolvimento de software. Sua principal característica é promover a entrega incremental de valor, permitindo que equipes trabalhem de forma iterativa, colaborativa e adaptável às mudanças nos requisitos do projeto.

Diferente de metodologias tradicionais, como o modelo em cascata, o Scrum enfatiza a *auto-organização* das equipes e a melhoria contínua. Ele é estruturado em torno de três pilares fundamentais: *transparência*, *inspeção* e *adaptação*, que orientam todos os processos do framework (SCHWABER; SUTHERLAND, 2020b).

2.6.1 Papéis no Scrum

O Scrum define três papéis principais:

- **Product Owner**: responsável por maximizar o valor do produto e gerenciar o *Product Backlog*, priorizando funcionalidades de acordo com o retorno esperado.
- **Scrum Master**: atua como facilitador, garantindo que o time compreenda e aplique corretamente os princípios do Scrum, removendo impedimentos e promovendo a melhoria contínua.
- **Time de Desenvolvimento**: equipe multidisciplinar encarregada de transformar os itens do *Product Backlog* em incrementos de produto potencialmente entregáveis a cada *Sprint*.

2.6.2 Eventos no Scrum

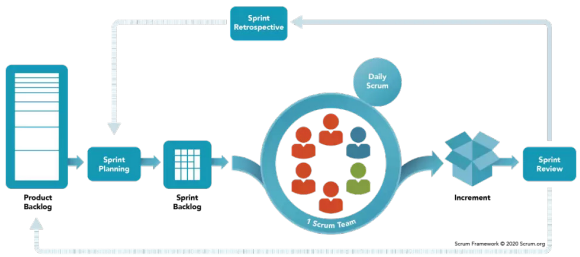

Figura 2 – Eventos do Scrum. Fonte: (PM3, 2023)

O framework é organizado em eventos de tempo fixo, conhecidos como *time-boxes*:

- **Sprint**: ciclo de desenvolvimento que dura, geralmente, entre uma e quatro semanas, no qual o time desenvolve funcionalidades priorizadas.

- **Sprint Planning**: reunião para definir o objetivo da *Sprint* e planejar as tarefas a serem executadas.
- **Daily Scrum**: reunião diária de curta duração (até 15 minutos) para alinhar o progresso e identificar impedimentos.
- **Sprint Review**: apresentação do incremento desenvolvido ao final da *Sprint* para stakeholders.
- **Sprint Retrospective**: encontro destinado a refletir sobre o que funcionou bem, o que pode ser melhorado e quais ações serão tomadas na próxima *Sprint*.

2.6.3 Artefatos do Scrum

O Scrum define três artefatos principais para gerenciar o trabalho:

- **Product Backlog**: lista priorizada de funcionalidades, requisitos e melhorias a serem implementadas no produto.
- **Sprint Backlog**: conjunto de itens selecionados do *Product Backlog* para serem desenvolvidos durante a *Sprint*.
- **Incremento**: versão potencialmente utilizável do produto ao final de cada *Sprint*.

O Scrum tem se consolidado como uma das metodologias ágeis mais adotadas mundialmente devido à sua simplicidade, flexibilidade e foco na entrega de valor contínuo. Além disso, pesquisas recentes destacam sua eficácia no aumento da produtividade, redução de riscos e melhoria da comunicação entre equipes distribuídas (HOSSAIN et al., 2009).

3 RELATO DE EXPERIÊNCIA

O ingresso no estágio na CWI Software ocorreu por meio do *Programa CWI Crescer*, uma iniciativa de capacitação profissional promovida pela empresa, destinada a pessoas a partir de 18 anos que possuam vínculo acadêmico, seja em curso técnico ou superior, na área de tecnologia. O programa tem como objetivo oferecer uma formação prática e intensiva, desenvolvendo competências técnicas e comportamentais que preparem os participantes para atuar em projetos reais e, eventualmente, integrar o quadro de colaboradores da empresa.

O programa foi estruturado em duas etapas principais: o **Level 1**, voltado para a formação inicial, e o **Level 2**, direcionado para o aprofundamento técnico e aplicação prática por meio de projetos.

A **Level 1**, teve duração de dois meses e foi voltada para a construção de uma base sólida em desenvolvimento web. Durante esse período, foram abordados conceitos fundamentais como JavaScript, HTML, CSS, orientação a objetos e bancos de dados, sempre com foco nas tecnologias e metodologias mais modernas utilizadas no mercado.

O aprendizado foi organizado em módulos, cada um concluído com um *projeto prático desafiador*, no qual era necessário aplicar os conhecimentos adquiridos. Essa estrutura exigiu dedicação e favoreceu o desenvolvimento de habilidades essenciais, como resolução de problemas, pensamento lógico e criatividade, consolidando conceitos vistos em disciplinas do curso como Programação Orientada a Objetos (POO) e Lógica de Programação, que forneceram a base necessária para compreender estruturas, fluxos de controle e modularização de código. Essa fase representou um período de intensa evolução, marcado por crescimento técnico, amadurecimento pessoal e aprimoramento profissional.

A **Level 2**, foi mais extensa e aprofundada, passando por diferentes etapas de aprendizado. Inicialmente, houve uma breve revisão e adaptação dos conceitos de JavaScript e TypeScript, fundamentais para o desenvolvimento moderno de aplicações.

Em seguida, o foco passou para o **React**, explorando a criação de interfaces dinâmicas, componentes reutilizáveis e boas práticas de desenvolvimento front-end. Para consolidar o aprendizado, foram propostos projetos práticos simulando cenários reais do mercado, permitindo aplicar conceitos de arquitetura e testes em projetos concretos, integrando o conhecimento adquirido durante a disciplina de Arquitetura e Padrões de Projeto de Software, que abordou a organização de sistemas e boas práticas de design.

Posteriormente, avançamos para o desenvolvimento *backend* utilizando **Java**[1] **com Spring Boot**[2]. A adaptação à sintaxe foi mais tranquila por já possuir conhecimento prévio da linguagem. Nessa fase, os desafios foram mais complexos, envolvendo a implementação de *APIs*, integração com **PostgreSQL** [3] para persistência de dados e utilização de **Docker**[4] para conteinerização. O aprendizado teórico adquirido na disciplina de Bancos de Dados foi fundamental para projetar consultas eficientes, normalizar tabelas e garantir a integridade e consistência dos dados. Essas experiências contribuíram para o aprimoramento de habilidades técnicas avançadas, como integração de sistemas e padronização de ambientes.

Durante o Level 2, todos os módulos foram concluídos com **projetos em equipe**, reforçando a importância do trabalho colaborativo, da troca de conhecimento e da comunicação clara entre os integrantes. O ponto culminante da fase foi o **projeto final**, considerado o "TCC" do Crescer. Cada equipe recebeu um tema definido pela CWI e contou com o suporte de um *Product Owner*, designer e mentor técnico para validação das ideias de implementação.

Um dos requisitos do desafio era utilizar ao menos uma tecnologia não abordada durante o programa, o que estimulou o aprendizado autônomo e a capacidade de pesquisa independente. Durante o projeto final, atuei como *desenvolvedor full stack*, participando tanto da construção do *backend* quanto do *frontend*. Ao término, apresentamos o projeto para a banca avaliadora e, após essa etapa, recebi a confirmação da minha seleção para o estágio.

Com o término do Crescer, fui alocado para um time responsável pelo desenvolvimento e manutenção de sistemas e aplicativos corporativos para uma rede hospitalar. A recepção na empresa foi bastante acolhedora, evidenciando o cuidado da CWI com o bem-estar físico e mental de seus colaboradores.

Nesse processo de integração, a empresa adota um **sistema de padrinhos**, no qual cada estagiário recebe um profissional experiente para orientá-lo durante todo o período de adaptação e aprendizado. O padrinho desempenha um papel fundamental, auxiliando na compreensão dos processos internos, esclarecendo dúvidas técnicas, revisando códigos e acompanhando o desenvolvimento do estagiário, garantindo evolução contínua e alinhada às boas práticas da equipe.

---

1 Java é uma linguagem de programação orientada a objetos, criada pela Sun Microsystems em 1995, conhecida por sua portabilidade e robustez.

2 Spring Boot é um framework open-source para desenvolvimento de aplicações Java, que simplifica a configuração e execução de aplicativos, especialmente microserviços.

3 PostgreSQL é um sistema de gerenciamento de banco de dados relacional open-source, conhecido por sua robustez, conformidade com SQL e suporte a extensões avançadas.

4 Docker é uma plataforma de código aberto para conteinerização de aplicações, permitindo que softwares sejam empacotados com todas as suas dependências e executados de forma consistente em diferentes ambientes.

Após a integração na equipe, o foco principal foi a adaptação ao ambiente de trabalho e a aquisição de conhecimentos sobre o **Flutter**, tecnologia com a qual ainda não havia tido contato. Durante essa fase inicial, não atuei diretamente no projeto, o que representou um desafio, pois o ritmo do Crescer era mais acelerado. Entretanto, essa desaceleração planejada foi essencial para consolidar os fundamentos necessários antes de iniciar contribuições no desenvolvimento do produto.

Posteriormente, passei a atuar efetivamente no *aplicativo*, com foco na refatoração de módulos existentes e na implementação de melhorias seguindo os padrões estabelecidos pela equipe. A arquitetura do projeto é baseada na **Clean Architecture**, com **Bloc** e **Cubit** utilizados para o gerenciamento de estado, além da aplicação dos princípios **SOLID**, garantindo um código organizado, legível e de fácil manutenção. Conceitos estudados na disciplina de Arquitetura e Padrões de Projeto de Software ajudaram na compreensão das responsabilidades de cada camada, separação de preocupações e modularização.

Cada entrega passava por um processo de *code reviews* conduzido pelo padrinho, que fornecia orientações valiosas para aprimorar a qualidade do código e aprofundar o entendimento da estrutura do sistema. Essa prática reforçou a importância do feedback contínuo e da atenção a detalhes na programação corporativa, algo discutido em Engenharia de Software, disciplina que enfatiza processos, boas práticas e controle de qualidade no desenvolvimento de sistemas.

Além do desenvolvimento técnico, participei ativamente das cerimônias do **Scrum**, como *plannings*, *reviews* e *dailies*, fundamentais para alinhar prioridades, acompanhar o andamento das tarefas e definir metas de cada sprint. Essa experiência permitiu aprimorar habilidades comportamentais, como trabalho em equipe, organização, comunicação assertiva e gestão de tempo, essenciais para um ambiente corporativo.

Um dos maiores desafios enfrentados foi o primeiro contato com o **Flutter**, tecnologias e padrões que não faziam parte do meu repertório anterior. Apesar das dificuldades iniciais, o suporte constante do padrinho e da equipe foi essencial para adquirir confiança e desenvolver uma base sólida para lidar com tarefas cada vez mais complexas. Esse processo evidenciou a importância da resiliência, da paciência e da capacidade de aprender com erros, competências fundamentais no dia a dia de desenvolvimento de software.

Durante o estágio, pude aprimorar significativamente meus conhecimentos técnicos, aplicando princípios **SOLID**, utilizando **Bloc** e **Cubit** para o gerenciamento de estado e trabalhando com padronização de código em um ambiente corporativo real. Além disso, desenvolvi soft skills essenciais, como comunicação com colegas de diferentes níveis de experiência, liderança situacional e organização de tarefas via metodologias ágeis. A experiência consolidou minha capacidade de integrar conhecimentos teóricos adquiridos na graduação com a prática em

projetos reais, proporcionando uma visão mais ampla das demandas do mercado de tecnologia.

Por fim, algumas percepções e dicas que considero valiosas para futuros estagiários:

- **Aproveitar feedbacks** de padrinhos e colegas para evolução constante.
- **Estudar documentação oficial** antes de pedir ajuda, estimulando autonomia.
- **Registrar aprendizados** diários, boas práticas e padrões adotados pelo time.
- **Participar ativamente das cerimônias ágeis**, pois ajudam a entender prioridades e integração da equipe.
- **Manter a resiliência frente a novos desafios** e tecnologias desconhecidas, enxergando cada dificuldade como oportunidade de aprendizado.

Essa experiência foi determinante para o meu crescimento profissional, oferecendo uma compreensão prática das tecnologias modernas, da dinâmica de equipes ágeis e da aplicação de boas práticas de desenvolvimento, consolidando a base para uma carreira sólida na área de tecnologia.

4 CONCLUSÃO

O estágio na CWI Software tem representado uma etapa de extrema relevância para o meu desenvolvimento profissional e pessoal. Inicialmente, as atividades concentraram-se no estudo aprofundado de *Flutter*, *Clean Architecture*, *Bloc*, *Cubit* e dos princípios *SOLID*, conhecimentos fundamentais para compreender a estrutura do projeto e consolidar a base técnica necessária para atuar de forma mais eficiente e assertiva.

Atualmente, estou envolvido na refatoração do primeiro módulo do aplicativo, aplicando os conceitos adquiridos e participando ativamente das cerimônias do time, como *plannings*, *reviews* e *dailies*. Essa experiência tem sido essencial para integrar teoria e prática, permitindo não apenas a consolidação dos conhecimentos técnicos, mas também o desenvolvimento de competências comportamentais relevantes, como trabalho em equipe, comunicação eficaz e organização das tarefas no contexto de metodologias ágeis.

Além disso, o estágio proporcionou a oportunidade de perceber a relação direta entre o aprendizado acadêmico e o trabalho prático, mostrando como os conteúdos estudados no curso de Análise e Desenvolvimento de Sistemas, como programação, lógica, bancos de dados e engenharia de software, fundamentam as atividades realizadas no dia a dia corporativo. Essa conexão reforçou a importância de manter uma base sólida de conhecimentos e evidenciou que a prática profissional amplia, complementa e aprofunda o aprendizado teórico.

Ainda que esta seja apenas a fase inicial da minha trajetória profissional, o estágio tem proporcionado uma compreensão mais ampla acerca das boas práticas de desenvolvimento, da importância da padronização de código e do funcionamento de equipes ágeis em projetos de grande porte. Ademais, a vivência prática reforça a necessidade do aprendizado contínuo, evidenciando que o aprimoramento constante das competências técnicas e comportamentais é indispensável para acompanhar as demandas do mercado e evoluir como desenvolvedor.

REFERÊNCIAS

(ABDI), A. B. de D. I. **PINTEC aponta que 85% das empresas utilizaram tecnologias digitais avançadas em 2022**. 2022. Acessado em 01 de setembro de 2025. Disponível em: <https://www.abdi.com.br/pintec-aponta-que-85-das-empresas-utilizou-tecnologias-digitais-avancadas-em-2022/>.

ABZAL, S. Solid principles in clean architecture with flutter. **Medium**, jun 2025. Acesso em: 30 ago. 2025. Disponível em: <https://medium.com/@serikbay.a04/solid-principles-in-clean-architecture-with-flutter-07e511d10ae0>.

AUNG, S. T.; FUNABIKI, N.; AUNG, L. H.; KINARI, S. A.; MENTARI, M.; WAI, K. H. A study of learning environment for initiating flutter app development using docker. **Information**, v. 15, n. 4, 2024. ISSN 2078-2489. Disponível em: <https://www.mdpi.com/2078-2489/15/4/191>.

BIBEROGLU, Y. **BLoC vs Cubit: Unraveling the Secrets of State Management in Flutter**. 2023. <https://yusufbiberoglu.medium.com/bloc-vs-cubit-unraveling-the-secrets-of-state-management-in-flutter-f0de7268ee38>. Acesso em: 30 ago. 2025.

CREATE, M. **22 Digital Transformation Statistics for 2022**. 2022. Acessado em 01 de setembro de 2025. Disponível em: <https://moduscreate.com/blog/22-digital-transformation-statistics-for-2022/>.

CWI Software. **Crescer CWI – Programa de impulsionamento profissional**. 2025. Acesso em: 30 ago. 2025. Disponível em: <https://cwicrescer.cwi.com.br/>.

____. **Sobre a CWI Software**. 2025. <https://cwi.com.br/sobre/>. Acesso em: 30 ago. 2025. Disponível em: <https://www.cwi.com.br/>.

EFENDI, F. **The Advantages of Flutter: Revolutionizing Cross-Platform App Development**. 2025. Acesso em: 31 ago. 2025. Disponível em: <https://fluttermasterylibrary.com/1/1/1/2/>.

GitLab Inc. **GitLab: The Complete DevOps Platform**. 2024. Acesso em: 30 ago. 2025. Disponível em: <https://about.gitlab.com/>.

HOSSAIN, E.; BABAR, M. A.; PAIK, H. Scrum practices in global software development: A research framework. In: IEEE. **2009 IEEE International Conference on Global Software Engineering**. [S.l.], 2009. p. 175–184.

JEDYNASTY, R. **Flutter Hot Reload Feature: Benefits & Performance Explained**. 2025. <https://www.miquido.com/blog/flutter-hot-reload/>. Acesso em: 31 ago. 2025.

MARQUES, D. **Flutter: quando usar o Bloc ou Cubit**. 2023. <https://medium.com/@ddanielddiniz/bloc-vs-cubit-flutter-cfabe13040d4>. Acesso em: 30 ago. 2025.

MARTIN, R. C. **Agile Software Development: Principles, Patterns, and Practices**. [S.l.]: Prentice Hall, 2003. ISBN 978-0135974445.

____. **The Clean Architecture**. 2012. <https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html>. Acesso em: 11 set. 2025.

____. **Clean Architecture: A Craftsman's Guide to Software Structure and Design**. 2018. Robert C. Martin Series.

Microsoft Corporation. **Azure DevOps Services: Automate Software Delivery**. 2024. Acesso em: 30 ago. 2025. Disponível em: <https://azure.microsoft.com/pt-br/services/devops/>.

MORIN, T. **Understand the Theory Behind Clean Architecture**. 2025. Acesso em: 31 ago. 2025. Disponível em: <https://blog.scalablebackend.com/understand-the-theory-behind-clean-architecture>.

NDepend Team. **Clean Architecture: The Rescue of Domain-Centric Design**. 2025. Acesso em: 31 ago. 2025. Disponível em: <https://blog.ndepend.com/introduction-clean-architecture/>.

OKTAY, Y. **State Management in Flutter: Writing Clean Code with BLoC and Cubit**. 2023. <https://medium.com/@yunusoktaydev/state-management-in-flutter-writing-clean-code-with-bloc-and-cubit-f9d6d803ef10>. Acesso em: 30 ago. 2025.

PAVITHRAN. **Effective State Management with Flutter Cubit**. 2023. <https://pavithran07.medium.com/effective-state-management-with-flutter-cubit-24bd68a50aed>. Acesso em: 30 ago. 2025.

PM3. **Scrum: o que é, como funciona e como aplicar**. 2023. Acessado em 01 de setembro de 2025. Disponível em: <https://pm3.com.br/blog/scrum-o-que-e-como-funciona-e-como-aplicar/>.

RADADIYA, S. M. R. S. **The Impact of Flutter Development**. 2025. <https://www.irjmets.com/uploadedfiles/paper//issue_3_march_2025/70270/final/fin_irjmets1743146075.pdf>. Acesso em: 31 ago. 2025.

RANGEL, M. **Gerência de estado BLoC e Cubit no Flutter**. 2023. <https://www.dio.me/articles/gerenciando-estado-com-bloc-e-cubit-no-flutter>. Acesso em: 30 ago. 2025.

RENNER, M. **CWI atinge nível 3 do CMMI**. 2025. Acesso em: 30 ago. 2025. Disponível em: <https://www.baguete.com.br/noticias/cwi-e-cmmi-nivel-3>.

SCHWABER, K.; SUTHERLAND, J. **The Scrum Guide – The Definitive Guide to Scrum: The Rules of the Game**. 2020. Acessado em 01 de setembro de 2025. Disponível em: <https://scrumguides.org/scrum-guide.html>.

____. **The Scrum Guide: The Definitive Guide to Scrum**. Scrum.org, 2020. Acesso em: 30 ago. 2025. Disponível em: <https://scrumguides.org/scrum-guide.html>.

STACKMARKET. **A Guide to State Management in Flutter: BLoC vs. Cubit**. 2024. <https://stackmarket.io/a-guide-to-state-management-in-flutter-bloc-vs-cubit/>. Acesso em: 30 ago. 2025.

SUTHERLAND, J. **Scrum: A Pocket Guide**. Creative Commons, 2010. Acesso em: 30 ago. 2025. Disponível em: <https://www.scrumguides.org/>.

Tecnosinos. **CWI Software**. 2025. Acesso em: 30 ago. 2025. Disponível em: <https://www.tecnosinos.com.br/company/cwi/>.

TOLLIN, G.; LIDEKRANS, M. React native vs. flutter: A performance comparison between cross-platform mobile application development frameworks. **Master's Thesis, Linköping University**, 2023. Acesso em: 31 ago. 2025.

VääNäNEN, T. **Flutter in cross platform development: Tools, performance and Dart integration**. 2025. Acesso em: 31 ago. 2025. Disponível em: <https://oulurepo.oulu.fi/bitstream/handle/10024/54358/nbnfioulu-202502261849.pdf>.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
Campus Cajazeiras - Código INEP: 25008978
Rua José Antônio da Silva, 300, Jardim Oásis, CEP 58.900-000, Cajazeiras (PB)
CNPJ: 10.783.898/0005-07 - Telefone: (83) 3532-4100

Documento Digitalizado Ostensivo (Público)

Entrega do TCC

Assunto:	Entrega do TCC
Assinado por:	Railson Silva
Tipo do Documento:	Anexo
Situação:	Finalizado
Nível de Acesso:	Ostensivo (Público)
Tipo do Conferência:	Cópia Simples

Documento assinado eletronicamente por:

- **Railson Mateus de Sousa Silva**, **ALUNO (202012010031) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 16/09/2025 13:13:33.

Este documento foi armazenado no SUAP em 16/09/2025. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 1610833
Código de Autenticação: 99187bd2fd