**FACULDADE DE TECNOLOGIA DE SÃO JOSÉ DOS CAMPOS**

**FATEC PROFESSOR Jessen Vidal**

**GILBERTO SOUZA SANTOS JUNIOR**

**SOFTWARE PARA AUXILIAR NA GESTÃO DE HORAS E LOCAIS DE TRABALHO DE EQUIPES REMOTAS**

São José dos Campos

2022

**GILBERTO SOUZA SANTOS JUNIOR**

**SOFTWARE PARA AUXILIAR NA GESTÃO DE HORAS E LOCAIS DE TRABALHO DE EQUIPES REMOTAS**

Trabalho de Graduação apresentado à Faculdade de Tecnologia de São José dos Campos, como parte dos requisitos necessários para a obtenção do título de Tecnólogo em Banco de Dados.

**Orientadora: Adriana da Silva Jacinto**

São José dos Campos

2022

**Dados Internacionais de Catalogação-na-Publicação (CIP)**

**Divisão de Informação e Documentação**

Santos Júnior, Gilberto Souza

SOFTWARE PARA AUXILIAR NA GESTÃO DE HORAS E  
 LOCAIS DE TRABALHO DE EQUIPES REMOTAS.

São José dos Campos, 2022.

999f. (número total de folhas do TG)

Trabalho de Graduação – Curso de Tecnologia em Banco de Dados.

FATEC de São José dos Campos: Professor Jessen Vidal, 20XX.

Orientador Interno ou Principal: Adriana da Silva Jacinto

1. Palavra-Chave 1. 2. Palavra-Chave 2. 3. Palavra-Chave 3. I. Faculdade de Tecnologia. FATEC de São José dos Campos: Professor Jessen Vidal. Divisão de Informação e Documentação. II. Título

**REFERÊNCIA BIBLIOGRÁFICA**

Santos Júnior, Gilberto Souza. **SOFTWARE PARA AUXILIAR NA GESTÃO DE HORAS E LOCAIS DE TRABALHO DE EQUIPES REMOTAS.** 2022. 999f. Trabalho de Graduação - FATEC de São José dos Campos: Professor Jessen Vidal.

**CESSÃO DE DIREITOS**

NOME(S) DO(S) AUTOR(ES): Gilberto Souza Santos Junior

TÍTULO DO TRABALHO: SOFTWARE PARA AUXILIAR NA GESTÃO DE HORAS E LOCAIS DE TRABALHO DE EQUIPES REMOTAS.

TIPO DO TRABALHO/ANO: Trabalho de Graduação/2022.

É concedida à FATEC de São José dos Campos: Professor Jessen Vidal permissão para reproduzir cópias deste Trabalho e para emprestar ou vender cópias somente para propósitos acadêmicos e científicos. O autor reserva outros direitos de publicação e nenhuma parte deste Trabalho pode ser reproduzida sem a autorização do autor.

|  |  |
| --- | --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Gilberto Souza Santos Júnior  Rua Clarisse Franco Rodrigues, 175 - Resid. São Francisco  12227-858, São José dos Campos - SP |  |

**GILBERTO SOUZA SANTOS JÚNIOR**

**SOFTWARE PARA AUXILIAR NA GESTÃO DE HORAS E LOCAIS DE TRABALHO DE EQUIPES REMOTAS**

Trabalho de Graduação apresentado à Faculdade de Tecnologia de São José dos Campos, como parte dos requisitos necessários para a obtenção do título de Tecnólogo em Banco de Dados.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Titulação, Nome do Componente da Banca - Sigla da Instituição**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Titulação, Nome do Componente da Banca - Sigla da Instituição**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Titulação, Nome do Orientador – Sigla da Instituição Titulação**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Nome do Coorientador (se existir) - Sigla da Instituição**

**\_\_\_\_\_/\_\_\_\_\_/\_\_\_\_\_**

**DATA DA APROVAÇÃO (dia da banca)**

Dedicatória (opcional)

O autor oferece a obra (elemento sem título e sem indicativo numérico), ou presta homenagem a alguém, de forma clara e breve em folha única.

**AGRADECIMENTOS**

Na página de agradecimentos o autor dirige palavras de reconhecimento àqueles que contribuíram para a elaboração do trabalho. O conteúdo não deve ultrapassar uma página e por isso, é necessário que ele seja sucinto e objetivo.

O texto deve ser escrito em Times New Roman, Tamanho 12, Alinhamento Justificado, Espaçamento entre linhas de 1,5 linhas e com recuo de parágrafo de 1,25 cm.

Epígrafe (opcional)

“É citada uma sentença escolhida pelo autor (elemento sem título e sem indicativo numérico), que deve guardar coerência com o tema abordado na obra.”

Nome do autor

**RESUMO**

Apresentação concisa dos pontos relevantes do documento deve ser exposta no resumo. No presente caso o resumo será informativo, assim deverá ressaltar o objetivo, a metodologia, os resultados e as conclusões do documento. A ordem desses itens depende do tratamento que cada item recebe no documento original. O resumo deve ser composto por uma sequência de frases concisas, afirmativas e não em enumeração de tópicos. Deve ser escrita em parágrafo único e espaçamento de 1,5 linhas. A primeira frase deve ser significativa, explicando o tema principal do documento. Deve-se usar o verbo na voz ativa e na terceira pessoa do singular. Quanto a sua extensão, o resumo deve possuir de 150 a 500 palavras.

**Palavras-Chave**: Com um mínimo de 3 e no máximo 6 palavras, separadas entre si por ponto e vírgula “;” e finalizadas por ponto. As palavras-chave sãopalavras representativas do conteúdo do documento.

**ABSTRACT**

O abstract é o resumo da obra em língua estrangeira, que basicamente segue o mesmo conceito e as mesmas regras que o texto em português. Recomenda-se que para o texto do abstract o autor traduza a versão do resumo em português e faça, se necessário, os ajustes referentes à conversão dos idiomas. É importante observar que o título e texto NÃO DEVEM estar em itálico.

**Keywords**: Recomenda-se que o autor traduza para o inglês as Palavras-Chave em português e faça, se necessário, os ajustes referentes à conversão dos idiomas.

**LISTA DE FIGURAS**

[Figura 1 - Proposta metodológica 17](#_Toc483917379)

**LISTA** **DE TABELAS**

[Tabela 1 - População de 15 a 24 anos de idade 18](#_Toc483917392)

**LISTA DE ABREVIATURAS E SIGLAS**

ARF Árvore da Realidade Futura

APS *Advanced Planning and Scheduling*

ARA Árvore da Realidade Atual

B2B *Business to Business*

CD Centro de Distribuição

CEPAA *Council on Economic Priorities Accreditation Agency*

**LISTA DE SÍMBOLOS**

dab Distância Euclidiana

O(n) Ordem de um Algoritmo

Sumário

[1. INTRODUÇÃO 15](#_Toc102498809)

[1.1. Objetivos do Trabalho 15](#_Toc102498810)

[1.2. Conteúdo do Trabalho 16](#_Toc102498811)

[2. FUNDAMENTAÇÃO TÉCNICA 17](#_Toc102498812)

[2.1 Desenvolvimento para Dispositivos Móveis 17](#_Toc102498813)

[2.1.1 Android 17](#_Toc102498814)

[2.1.2 IoS 18](#_Toc102498815)

[2.1.3 Desenvolvimento Híbrido 19](#_Toc102498816)

[2. 2 Desenvolvimento de Backend 19](#_Toc102498817)

[2.3 Linguagens e Frameworks 20](#_Toc102498818)

[2.3.1 Java 20](#_Toc102498819)

[2.3.2 Spring 20](#_Toc102498820)

[2.3.3 Flutter 21](#_Toc102498821)

[2.3.4 Vue 21](#_Toc102498822)

[2.3.5 SQL 21](#_Toc102498823)

[2.4 IDEs 21](#_Toc102498824)

[2.4.1 IntelliJ Idea Comunity 22](#_Toc102498825)

[2.5 Arquiteturas e Padrões de Desenvolvimento 22](#_Toc102498826)

[2.5.1 MVC 23](#_Toc102498827)

[2.5.2 Arquitetura Cliente-Servidor 23](#_Toc102498828)

[2.6 Banco de Dados 24](#_Toc102498829)

[2.6.1 MySQL/MariaDB 24](#_Toc102498830)

[2.6.2 SQL Lite 24](#_Toc102498831)

[3. DESENVOLVIMENTO 25](#_Toc102498832)

[3.1 Requisitos do Sistema 25](#_Toc102498833)

[3.1.1 Requisitos Funcionais 25](#_Toc102498834)

[3.1.2 Requisitos Não Funcionais 25](#_Toc102498835)

[3.2 Casos e Usos 25](#_Toc102498836)

[3.3 Técnologias Utinlizadas 25](#_Toc102498837)

[3.4 Arquitetura Geral 25](#_Toc102498838)

[3.4.1 Servidor 25](#_Toc102498839)

[3.4.2 App Móvel 25](#_Toc102498840)

[3.5 Diagrama de Classe 25](#_Toc102498841)

[3.6 Modelo Entidade Relacionamento 25](#_Toc102498842)

[3.7 Desenvolvimento do Protótipo 25](#_Toc102498843)

[4. RESULTADOS 26](#_Toc102498844)

[5. CONSIDERAÇÕES FINAIS 27](#_Toc102498845)

[REFERÊNCIAS 28](#_Toc102498846)

# 1. INTRODUÇÃO

A presença cada vez maior da telefonia móvel gera uma necessidade de infraestrutura composta por torres e equipamentos para que as pessoas possam utilizar seus smartphones em qualquer lugar, quer seja numa grande ou pequena cidade, ou mesmo durante o deslocamento por uma estrada.

Estas torres que compõem a infraestrutura do sistema de telefonia móvel costumam ficar em diversos locais, inclusive locais remotos como à beira de estradas, e necessitam de uma vistoria constante para garantir a integridade dos mesmos.

Diversas empresas fazem parte desta atividade e possuem um quadro de técnicos que devem percorrer rotas específicas, com a finalidade de visitar estas torres e fazer uma vistoria das condições das mesmas, atestando necessidades de manutenção ou identificando que as mesmas sofreram atos de vandalismo.

Existe a necessidade de acompanhamanto de quais técnicos trabalharam em cada rota, quais as torres foram visitadas e vistoriadas, assim como o controle das horas trabalhadas. Os profissionais são pagos de acordo com as horas de trabalho e se foi uma rota diária ou um atendimento em caráter de urgência. Este controle, muitas vezes, é feito com auxílio de planilhas em Excel.

## 1.1. Objetivos do Trabalho

O objetivo geral deste trabalho é desenvolver um sistema que auxilie nesta tarefa de controle das atividades prestadas por estes técnicos, a proposta é de um app instalado no smartphone dos técnicos, onde a inserção das informações será feita diretamente neste e exportado para um servidor de backend para consolidação dos dados.

Desta forma, serrá possivel substituir com grande vantagem o uso do Excel, pois por meio de uma aplicação web será possível fazer consultas consolidadas, com recursos como filtragem por rota, por técnico, por carro,...

Também, foi informado pela empresa prestadora deste tipo de serviço de vistoria técnica, que existe a necessidade de envio de uma mensagem de whatsapp para outras empresas como a de segurança das torres, assim como a própria operadora de telefonia móvel (ex.: Vivo, Claro, Tim,...). Para atender a esta necessidade, o próprio app será capaz de exportar estes relatórios padronizados para serem enviados pelo whatsapp.

A inserção dos dados por meio do app mobile também minimizará erros no preenchimento das informações que ocorrem atualmente como, por exemplo, erro na data da execução do serviço.

## 1.2. Conteúdo do Trabalho

O presente trabalho está estruturado em cinco Capítulos, cujo conteúdo é sucintamente apresentado a seguir:

No Capítulo 2 é feita a fundamentação das tecnologias e conceitos necessários para o objetivo geral deste trabalho.

O Capítulo 3 apresenta o desenvolvimento da solução, assim com as dificuldades encontradas.

No Capítulo 4 são apresentados os resultados alcançados.

O Capítulo 5 apresenta as considerações finais a partir da análise dos resultados obtidos, assim como sugestões para trabalhos futuros.

# 2. FUNDAMENTAÇÃO TÉCNICA

## 2.1 Desenvolvimento para Dispositivos Móveis

As plataformas de dispositivos móveis, como Smartphones e tablets, há poucos anos era composta por diversos players, tais como: Android, Blackberry, Ios, Windows Phone, .. Atualmente as 2 principais plataformas de dispositivos móveis são Android e Ios. Sendo assim, qualquer foco de desenvolvimento deve concentrar-se nestas duas opções.

Android e Ios, cada um tem suas particularidades tanto na perspectiva do usuário, quanto na do desenvolvedor. Ambos são implementados sobre um Kernel Linux, possuem diferenças nas linguagens nativas e na arquitetura de implementação. A Figura 1 apresenta uma comparação entre as arquiteturas:

Diagrama, Esquemático

Descrição gerada automaticamente

Fig.1 https://www.cs.columbia.edu/~nieh/pubs/asplos2014\_cider.pdf

Como foco deste trabalho é o desenvolvimento, será abordado sobre o desenvolvimento nativo para cada plataforma, assim como técnicas de desnvolvimento hibrido que atenda a ambas.

### 2.1.1 Android

O desenvolvimento nativo Android necessita ser feito em apenas 2 opções de linguagens: Java ou Kotlin.

Kotlin foi criada em 2016 pela JetBrains no intuito de ser uma linguagem com funcionalidades entçao inexistentes no Java, e também exigir menos linhas de código e ser menos verbosa que o Java. Em 2017, a Google anunciou que o Kotlin seria linguagem oficial de desenvolvimento Android, em interoperalidade com as bibliotecas Java, mas não seria uma substituta, ambas seriam oficialmente suportadas pela plataforma Java. Atualmente, devido a facilidade, o Kotlin tem sido cada vez mais adotado pelos desenvolvedores Android.

Com relação a arquitetura, o sistema Android possui diversas camadas.

A arquitetura do sistema Android conforme a documentação oficial é a seguinte:

Interface gráfica do usuário, Gráfico, Gráfico de mapa de árvore

Descrição gerada automaticamente

Fig.2 https://developer.android.com/guide/platform/images/android-stack\_2x.png

### 2.1.2 IoS

O desenvolvimento nativo Ios (Iphone) necessita ser feito em apenas 2 opções de linguagens: Objetive-C ou Swift, sendo que atualmente a principal linguegem usada para o desenvolvimento é o Swift. O Swift foi anuciada e criada pela Apple em 2014, tendo como objetivo de ser uma linguagem multiplataforma (iOS, macOS, watchOS, tvOS e Linux), podendo ser facilmente agragada a códigos Objetive-C já existentes. Atualmente é a linguagem mais utilizada pelos desenvolvedores para IoS.

O desenvolvimento de apps Ios conforme documentação da Apple é baseada na arquitetura Cooca que permite o uso de uma metodologia única quer seja para Singles Pages ou Mutlti Pages Applications e usa o padrão MVC. A arquitetura padrão é esta:

Diagrama

Descrição gerada automaticamente

Fig.3 https://developer.apple.com/library/archive/documentation/General/Conceptual/MOSXAppProgrammingGuide/Art/multi-window-app\_2x.png

### 2.1.3 Desenvolvimento Híbrido

Apesar de desnveolver em linguagens nativas dos sistemas Android e Ios possa garantir maior desempenho das aplicaçõs, ter que desenvolver uma aplicação para Android usando Java ou Kotlin, e ter que reescrever a mesma aplicação em Objetive-C ou Swift para que funcione nos Iphones/Ios, devemos concordar que é muito trabalhoso e um desperdício de tempo, para resolver esta questão surgiram diversos linguagens / frameworks / plataformas que permitam o desenvolvimento hibrido usando apenas uma única linguagem de programação. Isto é o que chamamos de desenvolvimento hibrido.

Algumas opções para o desenvolvimento hibrido são Ionic, React Native, Flutter. Destas, o Flutter tem crescido muito em popularidade, por esta razão, neste trabalho será adotado o mesmo.

## 2. 2 Desenvolvimento de Backend

Normalmente muitos sistemas trabalham numa arquitetura Cliente-Servidor, isto permite por exemplo que se tenha diversos clientes e alguns processamentos centralizados num único servidor. Pela característica do problema a ser resolvido por este trabalho acadêmico, esta foi a arquiteura adotada.

No servidor podem ser processados os consolidadores de informações geradas pelos diversos clientes e os relatórios destes dados, assim como as alterações nas características das rotas que as equipes técnicas percorrerão podem ser atualizadas somente no servidor, e os apps nos smartphones (ondes estarão as aplicações clientes) já obterão os dados atualizados.

## 2.3 Linguagens e Frameworks

Para o desenvolvimente de qualquer sistema, software ou aplicativo, existem diversas linguágens de programação e frameworks disponíveis, cada um atende uma necessidade específica, assim como tem vantagens, desvantagens e facilidades para cada tipo de cenário em que será necessário utilizar.

Neste trabalho foram escolhidos as seguintes linguuagens e frameworks: Java, Spring, Flutter, Vue e SQL. Nos tópicos seguintes serão descritas cada uma, assim como qual será o uso e qual razão de cada escolha.

### 2.3.1 Java

Desenvolvida pela equipe chefiada por James Gosling na Sun Microsystems em 1991, e posteriormente adquirida pela Oracle em 2008. Tem como principais características em ser orientada à objetos e não ser compilada diretamente para um sistema nativa, mas sim em ser compilada para um bytecode que é interpretado por uma máquina virtual (Java Virtual Machine, ou JVM), isto garante que um mesmo código possa ser executado em diferentes sistemas operacionais, bastando para isto que no mesmo tenha sido instalado um JVM específica para o SO, e esta JVM que irá interpretar o bytecode, traduzindo-o para um código nativo que o SO possa entender e executar.

### 2.3.2 Spring

O Spring é um ecossistema modular que possui diversos frameworks, cada um com uma funcionalidade, que juntos tem o objetivo de trazer facilidade ao desenvolvimento Java, assim, dentro do Spring temos framework para conexão com o banco de dados, para montar uma aplicação MVC, e diversos outros. Assim, podemos usar apenas os módulos necessários ao desenvolvimento da aplicação, e se houver necessidade de nova funcionalidade futura, pode-se adicionar posteriormente os módulos que forem necessários à esta nova funcionalidade.

Alguns módulos do ecossistema Spring são: Spring Boot, Spring Framework, Spring Data, Spring Security, Spring Web Services, e muito outros. O Spring Framework será a base do desenvolvimento do backend deste trabalho, ele possui a seguinte arquitetura:

Interface gráfica do usuário

Descrição gerada automaticamente

Fig.4 Spring MVC: Domine o principal framework web Java

### 2.3.3 Flutter

O Flutter é um kit para o desenvolvimento para interface de usuário criado pela Google em 2015 e baseado na linguagem Dart que permite a criação de aplicativos compilados nativamente para o sistema Android. Permite facilmente desenvolver aplicativos que rodem no Android ou no Iphone.

### 2.3.4 Vue

Vue, se pronuncia com View, é um framewrok progressivo baseado em Javascript para criação de interfaces de usuário, foi projetado desde sua concepção para ser incremental, sua biblioteca principal é focada na camada visual, sendo assim fácil de integrar com outras bibliotecas ou projetos existentes. O Vue foi adotado neste trabalho pela sua simplicidade se comparado a outros frameworks de frontend como o React ou Angular, e devido ao mesmo já ser abordado em matéria lecionada durante o curso na Fatec

### 2.3.5 SQL

O SQL (Structure Query Language) é a linguagem criada na década de 70 pela IBM para realizar as operações como inserção, modificação, consulta e exclusão de dados em um Banco de Dados. Podemos agrupar os seus comandos em 4 grupos: DML, DDL, DCL e TCL.

* Data Manipulation Language (DML) é o conjunto de instruções para consulta e modificação dos dados: SELECT, INSERT, DELETE, UPDATE, MERGE
* Data Definition Language (DDL) é o conjunto de comandos prar criar ou modificar estruturas de armazenamento, manipulação de dados ou visualização de dados, como tabelas, procedures, triggers e views: CREATE, ALTER, DROP.
* Data Control Language (DCL) é o conjunto de comandos para gerenciar as permissões, como de leitura ou modificação de dados, de criação de tabelas, ..., são: GRANT, REVOKE, DENY.
* Transactional Control Language (TCL) é o conjunto de instruções para gerenciar mudanças feitas pelas instruções DMLe permite que as declarações sejam agrupadas em declarações lógicas, são: COMMIT e ROLLBACK

O SQL evoluiu desde sua criação e foi adotado por diversas empresas, cada uma usando um dialeto, o que levou a uma padronização pela American National Standards Institute (ANSI), assim surgiu o padrão com as devidas revisões:  SQL-92, SQL3, SQL2003, ... Isto não impede que cada SGBD (será abordado no tópico 2.6) desenvolva sua variação personalizada da linguagem SQL padrão.

## 2.4 IDEs

Antes do surgimento dos IDEs, os programadores escreviam em editores de textos, salvavam, executavam compiladores, tentavam executar, anotavam os erros, e corrigiam os códigos nos editores de testos, e repetiam este processo até o código estivesse correto, isto era muito trabalho. Para melhorar este processo, surgiram os IDEs (Integrated Development Environment). Os precursores na utilização de um ambiente integrado de desenvolvimento foram o Turbo Pascal da Borland (1983) e o Visual Basic da Microsoft (1991), de lá pra cá surgiram diversos IDEs, alguns específicas para uma linguagem de programação, outros suportando o desenvolvimento em diversas linguagens.

Ainda é possível usar este processo, e digitar os códigos num editor de texto, compilar e executar, mas é muito mais prático fazer tudo isto através de um IDE. Atualmente, os IDEs reunem editores de código, depurador, compilador, além de diversas outras funcionalidades integradas, plugins e terminais acoplados que facilitam muito o desenvolvimento. Alguns exemplos são Atom, Sublime Text, Visual Studio Code, Eclipse, IntelliJ, ...

Existem também os que são apenas editores de código, não uma IDE completa, mas que possuem recursos como autocompletar palavras chaves de linguagens e alguns outros recursos, entre eles, identificar erros na sintaxe. Um exemplo é o Notepad++

IDEs e Editores mais populares conforme pesquisa do Stack Overflow

Gráfico, Gráfico de funil

Descrição gerada automaticamente

Fig.5 https://www.alura.com.br/artigos/assets/o-que-e-uma-ide/imagem4.png

### 2.4.1 IntelliJ Idea Comunity

O IntelliJ Idea Community possui suporte à diversas linguagens e frameworks, como Java, Kotlin, Scala, Python, Rust, Dart, HTML, CSS, processadores CSS como SASS, Swing, Android, Maven, ... Para Java possui depurador, compilador, decompilador, cobertura de testes.

Por suportar as linguagens e frameworks que serão usadas neste trabalho, pelos recursos e pela popularidade (que traz facilidade em encontrar comunidades online para buscar ajuda, caso seja necessário), por estas razões foi a principal IDE escolhida.

## 2.5 Arquiteturas e Padrões de Desenvolvimento

Resumidamente, definem como as partes de um software são organizadas, o comportamento (propriedades externamente visiseis por outras partes) e as responsabilidades de cada parte em relação ao software por completo.

### 2.5.1 MVC

O MVC é uma sigla (Model-View-Controller), ou seja, 3 camadas: visão, controle e modelo. O padrão MVC foi descrito originalmente em 1979 pelo cientista da computação Trygve Mikkjel Heyerdahl Reenskaug no artigo "Applications Programming in Smalltalk-80: How to use Model-View-Controller”, sendo um dos padrões mais utilizados na atualidade no desenvolvimento de softwares.

Vantagens em se utilizar o MVC:

* Reduz o acomplamento: Um baixo acoplamanto, uma parte do código (classe por exemplo) deve conhecer apenas a interface de outra classe para poder utilizá-la, nunca deve conhecer componentes internos destes outra parte(isto seria um alto acomplamanto) pois isto dificulta em muito a manutenabilidade do código.
* Aumenta a coesão: Uma alta coesão é quando cada parte (classe) possui um único e bem definido propósito, isto facilita a manutenabilidade do código. O que deve ser evitado é a baixa coesão, quando cada parte possui multiplas atribuições.

As 3 camadas do MVC possuem as seguintes responsabilidades:

* View
* Controller
* Model

Diagrama

Descrição gerada automaticamente

Fig.6 https://arquivo.devmedia.com.br/artigos/Higor\_Medeiros/IntroducaoMVC/IntroducaoMVC02.jpg

### 2.5.2 Arquitetura Cliente-Servidor

#### 2.5.3 HTTP



Resumo a partir da referência: <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Status>

#### 2.5.4 Web Service

#### 2.5.5 REST

#### 2.5.6 JSON

## 2.6 Banco de Dados

### 2.6.1 MySQL/MariaDB

### 2.6.2 SQL Lite

# 3. DESENVOLVIMENTO

## 3.1 Requisitos do Sistema

### 3.1.1 Requisitos Funcionais

### 3.1.2 Requisitos Não Funcionais

## 3.2 Casos e Usos

## 3.3 Técnologias Utinlizadas

## 3.4 Arquitetura Geral

### 3.4.1 Servidor

### 3.4.2 App Móvel

(Estrutura de diretórios da aplicação) / (códigosimportantes)

## 3.5 Diagrama de Classe

## 3.6 Modelo Entidade Relacionamento

## 3.7 Desenvolvimento do Protótipo

# 4. RESULTADOS

# 5. CONSIDERAÇÕES FINAIS

# REFERÊNCIAS

Documentação <https://developer.apple.com/>

Documentação <https://developer.android.com/>

Documentação https://br.vuejs.org/

Documentação <https://flutter.dev/development>

Documentação <https://spring.io/>

Artigo <https://www.cs.columbia.edu/~nieh/pubs/asplos2014_cider.pdf>

Artigo <https://www.alura.com.br/artigos/o-que-e-uma-ide>

Artigo https://www.devmedia.com.br/introducao-ao-padrao-mvc/29308

Livros

Use a cabeça!: Java Capa comum – 16 novembro 2007

Edição Português por Bert Bates (Autor), Kathy Sierra (Autor)

Use a cabeça!: desenvolvendo para Android Capa comum – 20 setembro 2019

por David Griffiths (Autor), Dawn Griffiths (Autor)

Head First Kotlin: A Brain-Friendly Guide Capa comum – Ilustrado, 19 março 2019

Edição Inglês por Dawn Griffiths (Autor), David Griffiths (Autor)

Head First Swift

by Jon Manning, Paris Buttfield-Addison

Released November 2021

AGENDA 21. **Conferência da Nações Unidas sobre Meio Ambiente e Desenvolvimento**. Disponível em http://www.mma.gov.br/sitio/index.php?ido=conteudo.monta&idEstrutura=18 Acesso em: 12/10/2010.

ALVES, J. M. **Proposta de um Modelo Híbrido de Gestão da Produção**: **aplicação na indústria aeronáutica. 2001.** 236 f. Tese (Doutorado em Engenharia Mecânica) - Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, Campinas, 2001.

ALVES FILHO, A. G.; CERRA, A. L.; MAIA, J. L. ; SACOMANO NETO, M. e BONADIO, P. V. G. Pressupostos da Gestão da Cadeia de Suprimentos: Evidências de Estudos sobre a Indústria Automobilística. **G&P – Gestão & Produção.** Vol. 11, n. 3, p. 275-288, Set.-Dez. 2004.

ANGERHOFER, B. J. e ANGELIDES, M. C. *A model and a performance measurement system for collaborative supply chains.* **Science Direct - Decision Support Systems**, Vol. 42, p. 283-301, 2006.

BALLOU, R. H. **Gerenciamento da Cadeia de Suprimentos**. São Paulo: Artmed, 2005.

SANTOS, R. F. **Proposta de um sistema híbrido de Contabilidade Gerencial: Estudo de Caso na Empresa Siber do Brasil S.A. 2005.** 168 f. Dissertação (Mestrado em Ciência no Curso de Engenharia Aeronáutica e Mecânica, Área de Produção) - ITA - Instituto Tecnológico de Aeronáutica, São José dos Campos, 2005.

SANTOS, R. S. e ALVES, J. M. Proposta de um Modelo de Gestão da Cadeia de Suprimentos com o Apoio da Teoria das Restrições, VMI e B2B. In: ENCONTRO NACIONAL DE ENGENHARIA DE PRODUÇÃO, 2009, Salvador. **Anais...** Salvador, 2009. 12 f.

ZILIO, S. D. Modeling and verification of parallel processes. In: CASSEZ, Franck et al (Ed.). **Mobile processes:** a commented bibliography. New York: Springer-Verlag, 2001. p. 206-222. (Lectures Notes in Computer Science, v. 2067).

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR** 5462: 1994: confiabilidade e mantenabilidade: terminologia. Rio de Janeiro, 1994.

EMBRAPA. Unidade de Apoio, Pesquisa e Desenvolvimento de Instrumentação Agropecuária (São Carlos, SP). Paulo Estevão Cruvinel. **Medidor digital multissensor de temperatura para solos.** BR n. PI 8903105-9. 26 jun. 1989, 30 maio 1995.

MICROSOFT. **Project for windows 95:** project planning software. Version 4.1: [S.l.]: Microsoft Corporation, 1995. Conjunto de programas. 1 CD-ROM.

ALLISON, D.O.; MINECK, R.E. **Aerodynamic characteristics and pressure distributions for an executive-jet baseline airfoil section**. Washington, DC: NASA, 1993. 25 p. (NASA TM-4529).

MARINHO, P. A pesquisa em ciências humanas. Petrópolis: Vozes, 1980 apud MARCONI, M. A.; LAKATOS, E. M. Técnicas de pesquisa. São Paulo: Atlas, 1982.

As referências acima são das fontes:

Amarelo: Internet

Verde: Dissertação ou Tese de Mestrado e Doutorado

Azul Claro: Artigo publicado em periódico

Magenta: Livro

Azul Escuro: Congresso

**Vermelho:** Capítulo de livro

**Cinza:** Normas técnicas

**Roxo:** Patentes

**Verde Escuro:** Programa de computador

**Marrom:** Relatório técnico

**AZUL Petróleo:** Exemplo de referência com apud