# RELATÓRIO DO PROJETO INTERDISCIPLINAR: CaféZen

**Curso:** Desenvolvimento de Software Multiplataforma (DSM) **Disciplina:** Computação em Nuvem I **Semestre:** 5º Semestre


### Integrantes do Grupo:

- Danilo Benedetti
- Gustavo Moreira
- Thiago Resende
- Wilton Monteiro


---

## 1. Resumo e Objetivo do Projeto

O **CaféZen** é uma solução tecnológica desenvolvida com o objetivo de auxiliar pessoas a encontrarem um equilíbrio saudável entre o consumo de café e o bem-estar pessoal em uma rotina acelerada. O projeto ataca o problema da gestão do estresse e do consumo de estimulantes, oferecendo uma ferramenta que monitora hábitos e fornece *insights* baseados em dados.

A aplicação funciona integrando três pilares principais:

1. **Monitoramento de Hábitos:** Coleta de dados sobre consumo de café, qualidade do sono e estilo de vida.
2. **Predição de Estresse:** Utilização de Inteligência Artificial para classificar o nível de estresse do usuário.
3. **Recomendações Personalizadas:** Indicação do consumo ideal para promover saúde e produtividade sustentável.

## 2. Tecnologias Utilizadas

Para alcançar os objetivos propostos, o projeto foi estruturado em uma arquitetura de microsserviços, utilizando as seguintes tecnologias:

### Frontend (Aplicação Mobile)

- **Flutter:** Framework utilizado para o desenvolvimento do aplicativo mobile multiplataforma (Android/iOS), garantindo uma interface fluida e responsiva.
- **Bibliotecas:** Uso de `http` para comunicação com a API e `fl_chart` para visualização gráfica dos dados e resultados.

### Backend (API RESTful)

- **Node.js & Express:** Responsáveis pela lógica de negócios, gerenciamento de rotas e integração entre o banco de dados e o modelo de IA.
- **Autenticação:** Implementação de segurança com `jsonwebtoken` (JWT) para sessões e `bcrypt` para criptografia de senhas.

### Banco de Dados

- **MySQL:** Sistema gerenciador de banco de dados relacional escolhido para armazenar informações de usuários e registros de formulários.
- **Sequelize:** ORM (Object-Relational Mapper) utilizado para facilitar a manipulação dos dados e estruturação das tabelas via código JavaScript.

### Inteligência Artificial e Machine Learning

- **Python & Scikit-learn:** Linguagem e biblioteca utilizadas para o pré-processamento de dados e treinamento do modelo.
- **SVM (Support Vector Machine):** Algoritmo de classificação escolhido após testes comparativos (superando Random Forest e Regressão Logística), atingindo **100% de acurácia** no conjunto de testes sintéticos.
- **Joblib:** Utilizado para a serialização e persistência do modelo treinado (`modelo_estresse.joblib`) e do normalizador (`scaler_estresse.joblib`).

### Infraestrutura e DevOps

- **Docker & Docker Compose:** Utilizados para a containerização da aplicação. O `docker-compose` orquestra os serviços de Backend e Banco de Dados, garantindo que o ambiente de execução seja replicável e isolado, atendendo aos requisitos da disciplina de Computação em Nuvem.

## 3. Desenvolvimento e Resultados Alcançados

O desenvolvimento seguiu o fluxo de pipeline KDD (Knowledge Discovery in Databases) para a criação do modelo de IA. Os dados foram pré-processados para tratamento de valores nulos e normalização, garantindo que o modelo SVM recebesse entradas padronizadas.

A integração foi realizada através da API em Node.js, que invoca scripts Python (`predict.py`) via processos filhos (`child_process`), permitindo que a predição ocorra em tempo real assim que o usuário envia seus dados pelo aplicativo.

Como resultado, entregamos um sistema completo onde:

1. O usuário se cadastra e faz login no App.
2. Preenche formulários sobre sua rotina.

3. Recebe instantaneamente uma classificação de estresse (Baixo, Médio ou Alto) e recomendações práticas.

# 4. Instruções de Instalação e Execução

O ambiente foi configurado para ser executado de forma simples utilizando Docker, eliminando a necessidade de configurações complexas de ambiente local para o backend e banco de dados.

## Pré-requisitos

- Git
- Docker
- Docker Compose

## Passo a Passo

**1- Clonar o Repositório:**

```bash
git clone https://github.com/FatecFranca/DSM-P5-G04-2025-2.git
cd DSM-P5-G04-2025-2
```

**2- Configurar Variáveis de Ambiente:**

- Localize o arquivo `.env.example` na raiz do projeto.
- Faça uma cópia deste arquivo e renomeie para `.env`.
- As configurações padrão já são compatíveis com o ambiente Docker, não sendo necessária alteração imediata para testes locais.

**3- Iniciar os Serviços (Backend + Banco de Dados):** Execute o comando abaixo para construir e subir os contêineres:

```bash
sudo docker-compose up --build
```

*A API estará disponível em `http://localhost` (ou na porta configurada).*

**4 . Executar o Frontend (Mobile):**

- Navegue até a pasta do front: `cd front`
- Instale as dependências: `flutter pub get`
- Execute o emulador ou conecte um dispositivo físico e rode: `flutter run`