Definição de Projeto - v1.0

Socifit

I. Tecnologia e frameworks principais:

As tecnologias foram separadas em dois tipos: as tecnologias e frameworks que vão ser utilizadas no app desktop, e as tecnologias que serão utilizadas no backend, que vai ser consumido pelo app desktop.

Desktop

- ### Java

  Será utilizada como linguagem para desenvolver o desktop em conjunto com o Spring Boot como framework do sistema e o JavaFX como framework visual. Escolhida por todos terem uma maior familiaridade com ela e um bom conhecimento prévio.

- ### Spring Boot

  Será utilizado como framework para conexão com o backend, por conta da injeção de dependências e inversão de controle. Escolhido pela simplicidade de uso e ferramentas que adiciona ao Java.

- ### Gradle

  Gerenciador de dependências e pacotes que será utilizado no sistema, escolhido por estar sendo usado em outro projeto pela equipe e por ser um gerenciador de pacotes mais moderno para Java.

- ### JavaFX

  Biblioteca desktop para componentes visuais escolhida por ser amplamente utilizada para aplicações desktop para Java.

Backend

- ### Javascript/Typescript - Node.js

  A linguagem de programação e ambiente de execução utilizados serão o Javascript/Typescript e o Node.js, pela experiência e algumas implementações prontas de alguns participantes, que ajudarão a ter uma boa base para fazer o backend de forma mais consistente.

- ### Nest.js

  Será utilizado como framework de Node.js para que seja facilitada a padronização do projeto do backend e também haja ferramentas que auxiliem na criação do mesmo.

- ### Prisma (ORM)

  É um ORM que será utilizado para fazer a conexão e comunicação com o banco de dados. Será utilizado para facilitar e padronizar a comunicação da aplicação backend com o banco de dados.

- PostgreSQL

  Banco de dados relacional que será utilizado por ser um dos bancos de dados mais utilizados do mercado e também por todos os participantes terem bastante experiência com ele.

II. Ferramentas

As ferramentas utilizadas serão os sistemas escolhidos para ajudar tanto no desenvolvimento quanto na organização, repositório de código e comunicação.

1. Comunicação

- Discord

  Escolhido pela facilidade de organizar a informação em vários canais de interesse.

- Whatsapp

  Utilizado para comunicações mais informais.

2. Repositório e Gerenciamento de tarefas

- Gitlab

  Foi escolhido pela centralização de funcionalidades como criação de atividades, repositório de código e CI/CD mais consolidado dentro do mercado, tudo isso numa só ferramenta.

3. IDE

- IntelliJ (Java)

  IDE escolhida para trabalhar com Java graças a todas as ferramentas dentro dela que dão apoio ao desenvolvimento em Java e tornam muito mais produtivo o desenvolvimento com a linguagem.

- VSCode (Node.js)

  IDE escolhida para trabalhar com Node.js graças à quantidade de extensões e ferramentas voltadas para o desenvolvimento em Javascript/Typescript.

III. Arquitetura e Padrões

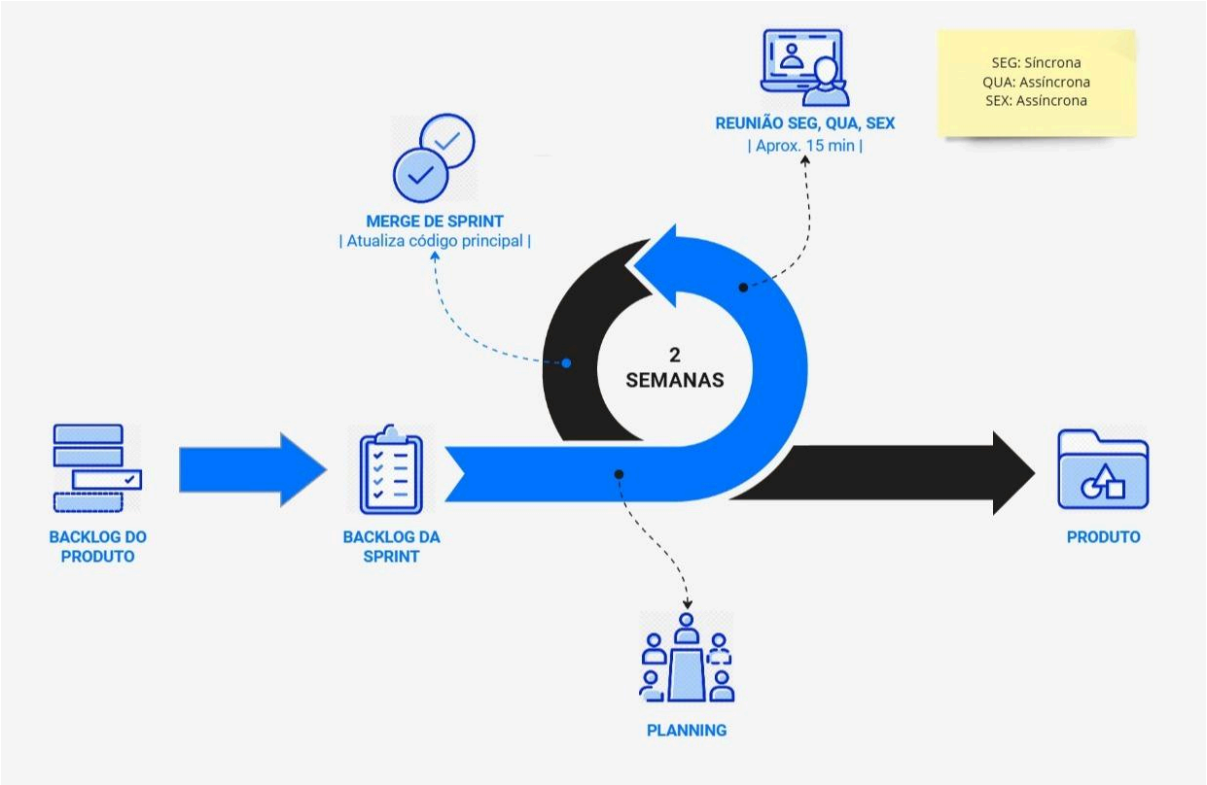
- Desktop

  O padrão de arquitetura escolhido para o desenvolvimento do sistema desktop será o padrão MVVM, porém ele vai ser adaptado inserindo também a camada de serviços que será a camada de comunicação com o backend.

- Backend

  Para o backend, o padrão de arquitetura escolhido foi o Controller-Service-Repository, onde serão criados os endpoints nos controllers para serem utilizados pela aplicação desktop, terá os repositories para acessar o banco de dados e os services para manter as regras de negócio do sistema.

IV. Processo



- Passos que foram definidos sobre como será o processo utilizado pela equipe:

1. O processo que estaremos utilizando será baseado no Scrum, com reuniões síncronas todas as segundas-feiras e reuniões assíncronas todas as quartas-feiras e sextas-feiras. No começo de cada sprint, será feita uma sprint planning para selecionar as histórias do backlog do projeto e trazê-las para o backlog da sprint como tarefas.

2. Antes de virarem tarefas na sprint, as histórias de usuários serão validadas com os stakeholders mais próximos da equipe e será feito o detalhamento delas para que sejam geradas tarefas e possam ser desenvolvidas durante a sprint.

3. No final da sprint, será feita uma validação do que foi desenvolvido como artefato para os stakeholders. Depois de validado, será separado o que vai ser aceito como artefato do sistema e o que vai precisar ser corrigido. O que precisar ser corrigido entrará como uma tarefa para a próxima sprint. E assim vamos construindo incrementalmente nosso software durante a disciplina de projeto.