* **Entrega 1 (Peso 5,0) — Documentação parcial da arquitetura** conforme os modelos e exemplos anexados a atividade do Google Classroom. Este documento deve conter ao menos:
  + Descrição do escopo do sistema (Luca)
  + Descrição dos requisitos arquiteturais, objetivos e restrições da arquitetura (Pedro paradela)
  + Definição dos padrões arquiteturais adotados (Pedro paradela)
  + Justificativa das decisões arquiteturais adotadas com base nos atributos de qualidade do sistema. Descrição de como a arquitetura de software contribui para os atributos de qualidade importantes do sistema (Pedro paradela)
  + Documentação das visões arquiteturais
    - O modelo conceitual (Daniel e Luca)
    - Ao menos diagramas de casos de usos (Luca e Pedro Paradela), modelo conceitual e diagramas de sequência do sistema (Se o sistema for grande, não precisa fazer os DSS dos CRUDs. Mínimo 1 DSS (diagramas de sequência do sistema) por integrante da equipe)
      * Cadastro de conta (Luca Machado)
      * Atualizar Dados Cadastrais (Luca Machado)
      * Aprovar certificado (Pedro Paradela)
      * Contratar Serviço(Rodrigo)
      * Banir usuário (Luca)
      * Aceitar contratacao (Luca)
      * Avaliar Serviço (Daniel)
      * Envio de Mensagens(Rodrigo)
      * Acessar histórico (Rodrigo)

-Diagrama com a visão geral do sistema (organização), incluindo os estilos arquiteturais utilizados e justificativas de uso - (Pedro Paradela)

* + Diagramas de interação. Pelo menos um diagrama por caso de uso. Escolher a operação do DSS mais complexa. Obs: Descrever o contrato da operação escolhida. (Mínimo 1 diagrama por integrante da equipe)
    - Cadastro de conta (Luca Machado)
    - Atualizar Dados Cadastrais (Luca Machado)
    - CRUD anúncios (Rodrigo)
    - CRUD certificado (Rodrigo)
    - Aprovar certificado (Pedro Paradela)
    - Contratar Serviço(Rodrigo)
    - Banir usuário (Luca)
    - Aceitar contratacao (Luca)
    - Avaliar Serviço (Daniel)
    - Envio de Mensagens(Rodrigo)
    - Acessar histórico (Rodrigo)
  + Diagramas de classe detalhado(Luca)
  + Apresentação(???)

**Servito**

**Documento de Arquitetura de Software**

**Versão <1.0>**

**Histórico da Revisão**

| **Data** | **Versão** | **Descrição** | **Autor** |
| --- | --- | --- | --- |
| 18/05/2025 | 1.0 | Primeira versão contendo os detalhes gerais da arquitetura junto com casos de uso e seus respectivos diagramas de Iteração | Luca Machado, Pedro Paradela, Rodrigo Chimelli, Daniel Molina |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Índice Analítico**

1. Introdução 4

1.1 Finalidade 4

2. Metas e Restrições da Arquitetura 4

3. Suposições e Dependências 4

4. Requisitos Arquiteturalmente Significantes 4

5. Decisões, Restrições e justificativas 4

6. Mecanismos Arquiteturais 5

7. Camadas da Arquitetura 5

8. Visões da Arquitetura 5

9. Qualidade 5

**Documento de Arquitetura de Software**

1. **Introdução**

Este é o documento de arquitetura de software do projeto “servito”, que consiste em uma aplicação web que permite que os usuários pessoas físicas, após se cadastrarem, anunciem e contratem serviços dos mais variados tipos (Reparos domésticos, design gráfico, aulas particulares, etc…). Os usuários poderão fazer buscas dos anúncios pelo tipo, localidade e nome do serviço. O site contará com um sistema de avaliação para os contratantes avaliarem os serviços dos prestadores. Também será possível que os usuários enviem certificados/diplomas para a equipe do site poder avaliá-los e confirmar a veracidade, passando mais credibilidade e confiança ao fazer o anúncio de seus serviços. O site também contará com moderação, que irá banir usuários que desrespeitarem as regras do site.

* 1. **Finalidade**.

A finalidade deste documento é fornecer uma representação detalhada e estruturada da arquitetura do sistema “Servito”, fazendo com que as decisões arquiteturais (sobretudo, padrões de projeto) relevantes sejam aqui registradas, bem como todos os diagramas de apoio (Iteração, diagrama de classes, casos de uso etc). Esses registros têm como objetivo atuar como guia para tanto a atual equipe começar desenvolvimento e manter qualidade técnica satisfatória que siga os padrões GRASP como demais boas práticas de programação no geral. Além disso, este documento irá servir como referência para atuais e futuros colaboradores poderem expandir as funcionalidades do sistema e para manutenção no geral do mesmo.

1. **Metas e Restrições da Arquitetura**

A ideia da arquitetura do "Servito" é a de construir uma plataforma de marketplace de serviços que seja confiável, escalável e intuitiva, priorizando a segurança dos dados e a facilidade de uso para prestadores e contratantes de serviços.

Questões que orientam a arquitetura:

* O sistema deve ser capaz de gerenciar um alto número de anúncios e interações simultâneas entre usuários, mantendo a estabilidade e a consistência dos dados.
* A integridade dos sistemas de avaliação e verificação de credenciais deve ser assegurada, sendo parte fundamental para manter a confiança e a reputação da plataforma.
* A arquitetura deve ser robusta e adaptável, com suporte a futuras expansões de funcionalidades, novos serviços e aumento progressivo da base de usuários.
* A funcionalidade de busca deve manter relevância e alto desempenho, mesmo em cenários com grande quantidade de anúncios cadastrados.

Questões críticas que a arquitetura deve resolver:

* **Dependência de Armazenamento de Arquivos (Certificados):** A arquitetura deve prever um mecanismo seguro e eficiente para o upload, armazenamento e acesso aos arquivos de certificados/diplomas, isolando essa funcionalidade para não sobrecarregar outros componentes do sistema.
* **Performance da Busca:** Com o crescimento da base de anúncios, como garantir que a funcionalidade de busca por tipo, localidade e nome do serviço permaneça rápida e relevante?
* **Gerenciamento de Estado e Sessão de Usuários:** Como gerenciar de forma eficiente as sessões dos usuários em uma aplicação web potencialmente distribuída?
* **Comunicação entre Componentes/Módulos:** Definir interfaces claras e eficientes para a comunicação entre os diferentes módulos do sistema (ex: como o módulo de anúncios interage com o módulo de usuários e avaliações).
* **Funcionamento em Condições de Alta Concorrência:** O sistema precisa lidar com múltiplos usuários realizando buscas, cadastrando anúncios, enviando mensagens e avaliando serviços simultaneamente, especialmente em horários de pico.
  1. **Metas**

Questões que Orientam a Arquitetura:

* **Modularidade e Baixo Acoplamento:** Isolar as principais funcionalidades (gestão de usuários, anúncios, busca, avaliações, verificação de credenciais, moderação, sistema de mensagens) em componentes ou módulos bem definidos, permitindo desenvolvimento, manutenção e escalabilidade independentes.
* **Escalabilidade:** A arquitetura deve ser capaz de suportar um aumento significativo no número de usuários, anúncios e transações sem degradação de performance. Isso inclui a capacidade de escalar horizontalmente os componentes do backend.
* **Segurança:** Proteger os dados pessoais dos usuários, as informações de anúncios, as mensagens trocadas e, crucialmente, o processo de verificação de credenciais. A arquitetura deve prever mecanismos robustos de autenticação, autorização e proteção contra vulnerabilidades comuns.
* **Usabilidade e Desempenho:** Oferecer uma interface de usuário responsiva e intuitiva, com tempos de resposta rápidos para as principais operações, como busca de serviços, visualização de anúncios e processo de contratação.
* **Manutenibilidade e Extensibilidade:** A estrutura do código e da arquitetura deve facilitar a correção de bugs, atualizações e a adição de novas funcionalidades no futuro com o mínimo de impacto em componentes existentes.
* **Confiabilidade:** Garantir que o sistema esteja disponível e funcione corretamente, especialmente as funcionalidades críticas como cadastro, busca, contratação e o sistema de avaliações.
* **Integridade dos Dados:** Assegurar a consistência e a validade dos dados armazenados, principalmente em relação aos perfis de usuário, anúncios, avaliações e o status de verificação de credenciais.
  1. **Restrições**
* **R01:** O sistema deve ser acessível em navegadores web, desktops e dispositivos móveis. A camada View do MVC deve ser responsiva.
* **R02:** Todas as informações pessoais devem ser protegidas. A lógica de segurança será implementada no acesso ao Modelo e nos Controladores.
* **R03:** O processo de verificação de credenciais deve ter um fluxo bem definido dentro da arquitetura MVC, sem afetar a performance dos Controladores principais.
* **R04:** Limitações de orçamento e de tempo para desenvolvimento inicial podem influenciar a granularidade dos componentes MVC e a escolha das tecnologias.
* **R05:** A estrutura MVC deve ser flexível para acomodar possíveis futuras expansões.

1. **Suposições e Dependências**

*[Liste as suposições e dependências que dirigem as decisões arquiteturais. Isto pode incluir áreas sensíveis ou críticas, dependências e interfaces com sistemas legado, a habilidade e experiência da equipe, a disponibilidade de recursos importantes, e assim por diante ]*

1. **Requisitos Arquiteturalmente Significantes**

*[Insira uma referência ou link para os requisitos que exploram aspectos relevantes da arquitetura.]*

1. **Decisões, Restrições e justificativas**

*[Liste as decisões tomadas em relação às abordagens arquiteturais. Estas decisões servirão como guias para definir as partes arquiteturalmente significantes do sistema. Justifique cada decisão ou restrição para que os desenvolvedores compreendam a importância de construir o sistema dentro desse contexto. Isto pode incluir uma lista de “Faça” ou “Não Faça” para guiar os desenvolvedores no desenvolvimento do sistema]*

* Decision or constraint and justification
* Decision or constraint and justification

1. **Mecanismos Arquiteturais**

*[Liste os mecanismos arquiteturais, como mecanismos de persistência, comunicação e tratamento de erros, por exemplo, e descreva ocorrente estado de cada um. Inicialmente, cada mecanismo pode ser somente um nome e uma breve descrição. Eles evoluirão até que o mecanismo se torne um padrão ou uma colaboração de elementos de projeto que possam ser aplicados diretamente em algum aspecto do projeto.]*

**Mecanismo Arquitetural 1**

*[Descreva a finalidade, os atributos e funções do mecanismo arquitetural.]*

**Mecanismo Arquitetural 2**

*[Descreva a finalidade, os atributos e funções do mecanismo arquitetural.]*

1. **Camadas da Arquitetura**

*[Descreva os padrões de arquitetura utilizados e como a arquitetura se manterá consistente e uniforme. Isto pode ser uma simples referência para um conhecido padrão arquitetural, como o padrão de divisão em camadas e uma descrição de como os componentes do sistema podem ser colocados juntos.]*

1. **Visões da Arquitetura**

*[Descreva as visões arquiteturais usadas para descrever a arquitetura. Isto ilustra as diferentes perspectivas disponíveis para rever e documentar as decisões arquiteturais.]*

Visões Recomendadas:

* **Lógica:** Descreva a estrutura e comportamento de porções arquiteturalmente significantes do sistema. Isto deve incluir a estrutura de pacotes, interfaces críticas, importantes classes e subsistemas e as relações entre estes elementos. Isto também inclui visões físicas e lógicas dos dados persistentes.
* **Operacional:** Descreva os nós físicos do sistema e os processos, threads e componentes que rodam em cada um desses nós. Esta visão não é necessária se o sistema roda num único processo e num único thread.
* **Casos de Uso:** Uma lista ou diagrama dos casos de uso que contém requisitos arquiteturalmente relevantes.

1. **Qualidade**

*[Uma descrição de como a arquitetura do software contribui para todos os recursos (exceto a funcionalidade) do sistema: extensibilidade, confiabilidade, portabilidade e assim por diante. Se essas características possuírem significado especial, como implicações de segurança, garantia ou privacidade, elas deverão ser delineadas claramente.]*