**UNIVERSIDADE VIRTUAL DO ESTADO DE SÃO PAULO**

ENGENHARIA DE COMPUTAÇÃO

Higor Henrique dos Santos Amorim

Hugo Peres de Freitas

Kelly Viana Moura

Marcus Vinicius Martins Beranger

Vanessa Cristina da Silva

Vitor Hugo Campos Alves

**DESENVOLVIMENTO DE SOFTWARE PARA GAMIFICAÇÃO BASEADO EM MICROSSERVIÇOS**

**Santo André- SP**

**2022**

Higor Henrique dos Santos Amorim

Hugo Peres de Freitas

Kelly Viana Moura

Marcus Vinicius Martins Beranger

Vanessa Cristina da Silva

Vitor Hugo Campos Alves

**DESENVOLVIMENTO DE SOFTWARE PARA GAMIFICAÇÃO BASEADO EM MICROSSERVIÇOS**

Trabalho de Conclusão de Curso apresentado ao curso Engenharia de Computação da Universidade Virtual do Estado de São Paulo como requisito parcial para obtenção do título de Bacharel em Engenharia de Computação.

Orientador(a): Thays de Souza João Luiz

**Santo André- SP**

**2022**

Higor Henrique dos Santos Amorim , Hugo Peres de Freitas,

Kelly Viana Moura, Marcus Vinicius Martins Beranger,

Vanessa Cristina da Silva, Vitor Hugo Campos Alves

**DESENVOLVIMENTO DE SOFTWARE PARA GAMIFICAÇÃO BASEADO EM MICROSSERVIÇOS**

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Engenharia de Computação da Universidade Virtual do Estado de São Paulo como requisito parcial para obtenção do título de Bacharel em Engenharia de Computação.

Aprovado em: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Conceito: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**BANCA EXAMINADORA**

Prof. [THAYS DE SOUZA JOÃO LUIZ]

[Universidade de São Paulo]

Presidente

Prof. [JORGE COSTA SILVA FILHO]

[Universidade Virtual do Estado de São Paulo]

Membro

**RESUMO**

Este trabalho relata o desenvolvimento de um aplicativo que utiliza a gamificação como recurso para fomentar a interação entre alunos da Universidade Virtual do Estado de São Paulo – Univesp. Concebido a partir da cultura DevOps (que integra as equipes de desenvolvimento e operações) e com a arquitetura baseada em microsserviços, a aplicação em outros polos e cursos torna-se viável, já que, serviços independentes são mais fáceis de escalar.

**Palavras-chave:** aplicativo. gamificação. DevOps. microsserviços.

LISTA DE FIGURAS

[Figura 1*-  Design Thinking Etapas do pensamento* 18](#_Toc106532136)

[Figura 2 - A transição da arquitetura de aplicativos monolíticos dos anos 90 para os aplicativos baseados em microsserviços de hoje. 26](#_Toc106532137)

**LISTA DE ABREVIATURAS E SIGLAS**

|  |  |
| --- | --- |
| API | Application Programming Interface |
| AWS | Amazon Web Services |
| ESB | Enterprise Service Bus |
| PwC | PricewaterhouseCoopers Brasil Ltda |
| SOA | Service-Oriented Architecture |
| SQL | Structured Query Language |
| UNIVESP | Universidade Virtual do Estado de São Paulo |
|  |  |

Sumário

[1. INTRODUÇÃO 16](#_Toc106551781)

[1.1. Objetivos Gerais 17](#_Toc106551782)

[1.2. Objetivos Específicos 17](#_Toc106551783)

[2. REVISÃO BIBLIOGRÁFICA 18](#_Toc106551784)

[2.1. Design Thinking 18](#_Toc106551785)

[2.2. Gamificação 19](#_Toc106551786)

[2.2.1. Mas por que gamificação? 20](#_Toc106551787)

[2.2.2. Elementos da gamificação 20](#_Toc106551788)

[2.2.3. Principais benefícios da gamificação 22](#_Toc106551789)

[2.3. Desenvolvimento de aplicativos em Java 24](#_Toc106551790)

[2.4. A Cultura DevOps 24](#_Toc106551791)

[2.5. Controle de versão 25](#_Toc106551792)

[2.6. A ferramenta GIT 25](#_Toc106551793)

[2.7. Introdução ao Docker 25](#_Toc106551794)

[2.8. Banco de dados relacionais e linguagem SQL 25](#_Toc106551795)

[2.9. Microsserviços 25](#_Toc106551796)

[2.9.1. Design Evolucionário 28](#_Toc106551797)

[3. METODOLOGIA 29](#_Toc106551798)

[3.1. Design Thinking - 1ª Etapa: Definir 29](#_Toc106551799)

[3.1.1. Definição do Escopo 29](#_Toc106551800)

[3.1.2. Delimitação do Escopo 30](#_Toc106551801)

[3.2. Design Thinking - 2ª Etapa: Pesquisa 31](#_Toc106551802)

[3.3. Design Thinking - 3ª Etapa: Gerar ideias 32](#_Toc106551803)

[3.4. Design Thinking - 4ª Etapa: Testar protótipos 32](#_Toc106551804)

[3.5. Design Thinking - 5ª Etapa: Selecionar 32](#_Toc106551805)

[3.6. Design Thinking - 6ª Etapa: Implementar 33](#_Toc106551806)

[3.7. Design Thinking - 7ª Etapa: Aprender 33](#_Toc106551807)

[4. DISCUSSÃO 33](#_Toc106551808)

[4.1. Criação do Projeto 33](#_Toc106551809)

[4.2. Definir os microsserviços em grupo; 33](#_Toc106551810)

[4.2.1. Desenvolvimento do aplicativo 34](#_Toc106551811)

[4.2.2. Criação do repositório 34](#_Toc106551812)

[4.2.3. Criação de branches builds/ ou commits com o Git 34](#_Toc106551813)

[4.2.4. Versionando o código- fonte 34](#_Toc106551814)

[4.2.5. Criação de microsserviços com o Docker 34](#_Toc106551815)

[5. CONSIDERAÇÕES FINAIS 34](#_Toc106551816)

[6. APÊNDICES 38](#_Toc106551817)

[6.1. APÊNDICE A – Requisitos Funcionais e não Funcionais 38](#_Toc106551818)

[6.2. APÊNDICE B – Protótipos front-end 41](#_Toc106551819)

[6.3. APÊNDICE C – Repositório 43](#_Toc106551820)

[6.4. APÊNDICE D – Telas Finais 44](#_Toc106551821)

[6.5. APÊNDICE E – Lições aprendidas 46](#_Toc106551822)

# INTRODUÇÃO

O ambiente universitário é sem dúvida nenhuma um organismo onde circula inovação e conhecimento em função da diversidade de assuntos, disciplinas e pessoas que interagem com o meio. Essa também é uma realidade das universidades à distância, contudo, nesse modelo percebemos uma menor sinergia entre alunos, professores e demais profissionais. Com o intuito de minimizar este problema a UNIVESP incluiu em sua grade curricular a entrega de projetos integradores, que além de desenvolver diversas competências profissionais como liderança, relacionamento interpessoal e pensamento criativo ajuda conectar os alunos e tutores, porém, esse modelo não alcança alunos de outros cursos como as licenciaturas, pedagogia etc.

Pensando em minimizar as fronteiras entre a Universidade, polos, cursos e alunos levantamos a seguinte questão: Como engajar a troca de conhecimento dentro da Universidade à distância?

Como alunos da Universidade vivenciamos as dificuldades em conciliar trabalho, estudos e qualidade de vida, entretanto, aprendemos também como a tecnologia pode ajudar a reduzir esses gargalos.

Este trabalho de conclusão de curso propõe a construção de um aplicativo que com uso da gamificação motivará os alunos a participarem da Universidade. A solução foi pensada considerando a variável tempo:

* Dispositivos mobile podem ser acessados a qualquer momento;
* A gamificação será utilizada como um meio para recompensar os esforços dos envolvidos;
* O gamificação servirá como um instrumento para manter o aplicativo vivo e fortalecer a troca de experiências e conhecimento.

E como a tecnologia vive em constante evolução faremos uso das mais recentes práticas como:

* Aplicação da arquitetura de microsserviços;
* Uso de serviço de contêiner e;
* Aplicação de cultura Devops.

# Objetivos Gerais

Desenvolver um aplicativo para gamificação orientado a microsserviços.

# Objetivos Específicos

* Engajar a troca de conhecimento dentro da Universidade;
* Bonificar os esforços dos envolvidos através da Gamificação;
* Apresentar as vantagens e desvantagens no desenvolvimento de sistemas em arquitetura orientada em Microsserviços X Monolítica;
* Apresentar Cultura Devops.

# REVISÃO BIBLIOGRÁFICA

# Design Thinking

Design Thinking ou projeto centrado nas pessoas (PINHEIRO; ALT, 2017 p.13) é o processo que transforma uma solicitação em um produto ou serviço e compreende sete etapas (AMBROSE, G; HARRIS, P., 2022, p.11):

Figura 1*-* Design Thinking Etapas do pensamento

Diagrama

Descrição gerada automaticamente

Fonte: AMBROSE, G; HARRIS, P., 2022

* Definir (Briefing): Tem como principais objetivos identificar as necessidades dos clientes e definir os limites dos projetos;
* Pesquisa: Etapa para coleta de informações para despertar o processo criativo, definição de restrições, identificação de barreiras etc.;
* Geração de ideias: Etapa de geração de ideias que inclui o entendimento do temor valor, aplicação de diversas técnicas como brainstorming, esboço de ideias e apresentação de ideias;
* Testar protótipos: Etapa cuja finalidade é aprimorar soluções e criar protótipos para testar a viabilidade da entrega. Um protótipo pode ser um esboço, um modelo, uma maquete etc.;
* Selecionar: Etapa de escolha da entrega que será desenvolvida, do formato e materiais que serão aplicados e da viabilidade das escolhas;
* Implementar: Momento de validação com responsáveis e clientes, revisão de especificações etc.;
* Aprender: Fase para coleta de feedback, de identificação de melhorias e de aprendizado dos estágios que aconteceram durante o processo de desenvolvimento.

# Gamificação

Ludificação, ou gamificação – do inglês gamification, pode ser definido como o uso de mecânicas e características de jogos para engajar, motivar comportamentos e facilitar o aprendizado de pessoas em situações reais, transformando conteúdos densos em materiais mais acessíveis e mais fáceis de serem compreendidos de forma lúdica.

Criar novas maneiras de engajar os participantes em cursos, palestras, treinamentos ou outras atividades educacionais é sempre um desafio para qualquer empresa ou instituição de ensino. Essa abordagem tenta responder como manter os funcionários interessados durante a dinâmica e trazer resultados verdadeiros, mesmo com as obrigações e atribulações cotidianas, e como manter a atenção de alunos de diferentes faixas etárias nas aulas e conteúdos educacionais.

Em poucas palavras, gamification é uma forma de usar elementos comuns dos jogos em situações que não se restringem ao entretenimento. É utilizar a lógica e metodologias dos games para servir a outros propósitos, como tornar conteúdos complexos em materiais mais acessíveis, facilitando os processos de aprendizado — tudo isso de forma mais dinâmica do que outros métodos.

Nessa estratégia, diversos elementos característicos da maioria dos jogos — como um sistema de pontuação e ranking, um objetivo claro a ser alcançado, recompensas ao completar as missões — podem ser usados para chamar a atenção e engajar pessoas, promover o aprendizado e motivar determinadas ações do público-alvo.

A gamificação pode ser aplicada nos mais diversos segmentos, desde educação, saúde, causas sociais, marketing e até em treinamentos corporativos, atingindo, assim, públicos e objetivos variados.

# Mas por que gamificação?

De acordo com a empresa de pesquisa Newzoo, temos aproximadamente 2.5 bilhões de pessoas que jogam no mundo todo. No Brasil, a estimativa é que 66,3% das pessoas jogam independentemente do tipo de dispositivo e possuem idade média entre 25 e 34 anos, conforme dados da Pesquisa Game Brasil.

Outros estudos mostram que colaboradores se sentem mais motivados quando há elementos de jogos em seus treinamentos, cerca de 83% responderam positivamente. Além disso, 89% se sentem mais produtivos quando seu trabalho está gamificado.

Estes números mostram que jogar faz parte do cotidiano das pessoas e aplicar elementos de jogos em atividades que não são de entretenimento pode ajudar no engajamento e na motivação.

# Elementos da gamificação

Conforme artigo do site Ludos Pro, os principais recursos da gamificação são:

* **Storytelling**

Conhecida como a arte de contar histórias, o storytelling é uma prática que busca desenvolver histórias ou narrativas com objetivo de entreter, educar, preservar a cultura ou incutir valores morais.

Na gamificação, esse recurso é utilizado principalmente para engajar o público-alvo e gerar interesse pelo conteúdo compartilhado. Assim pode-se dizer que ele é usado como estratégia para chamar atenção e tornar a prática mais prazerosa e interessante

* **Avatar**

Avatar é uma representação visual do personagem do jogador que pode ser aderida uma foto personalizada ou mesmo um desenho representativo, assim como algo mais sofisticado como visto nos jogos eletrônicos mais complexos.

Esse elemento está diretamente relacionado ao recurso de storytelling – comentado anteriormente -, pois o personagem criado precisa estar relacionado a ambientalização da história. Nesse cenário, o avatar é uma ferramenta que auxilia na imersão do participante na dinâmica ali proposta.

* **Desafios**

É o desafio que motiva os participantes a se esforçarem para alcançar a meta da atividade. Mas para que ele funcione, são trabalhados três recursos: desafio, conquistas e missões que geram interesse nos participantes para que se mantenham voluntária e ativamente na atividade.

Além disso, se faz necessário equilibrar o grau de dificuldade da prática proposta para que o participante entenda que consegue atingir o que foi apresentado. O oposto também precisa ser considerado, é importante que o profissional não veja o desafio como fácil demais, porque isso vai gerar desânimo e, consequentemente, desistência.

A ideia é que na medida em que o participante vá progredindo, a atividade aumente o nível de dificuldade, para que estimule o interesse e o faça alcançar o objetivo definido estrategicamente no desenvolvimento da atividade.

* **Recompensa**

Outra estratégia muito presente nos jogos, a recompensa é realizada de acordo com cada resposta positiva do jogador. Este também é um recurso para estimular o interesse do participante, por isso precisa ser implementado de maneira totalmente estratégica para cumprir com o seu papel.

Podem ser utilizadas diferentes formas de recompensa: avanço para a próxima etapa; moedas virtuais que possam ser trocadas por outros itens; pontuação; barra de experiência que cresce conforme cada vitória; medalhas de honra; troféus; destaques; feedback positivo, entre outros.

* **Pontos**

Como comentado anteriormente, os pontos são considerados um sistema de recompensa que evidenciam aos participantes seu progresso na dinâmica e aumenta sua motivação com as práticas.

Cada acerto gera determinada quantidade de pontos e conforme a dificuldade for aumentando, maior também pode ser a recompensa, assim o jogador se sente motivado a se desafiar cada vez mais e com isso cumprir a meta final da estratégia de gamificação.

Esse recurso também é usado para gerar um sentimento de competição entre os participantes, principalmente quando trabalhado junto o ranking, que você lerá sobre no próximo tópico.

* **Ranking**

Um sistema de ranking serve para identificar os melhores colocados na atividade, destacando aqueles que mais têm assimilado o conteúdo proposto. Ele serve para alimentar a competição e estimular interesse por parte dos participantes em se esforçar para alcançar melhores colocações.

# Principais benefícios da gamificação

Este tipo de abordagem vem crescendo e conquistando espaço em diversos lugares, como instituições de ensino, empresas e palestrantes. Diversos são os benefícios de adotar este modelo, tais como:

* **Aumento da motivação do time:** A estratégia desse recurso cria conexão entre os participantes e a atividade realizada, transformando a proposta em algo prazeroso.
* **Desenvolvimento de habilidades:** O ambiente e a proposta construída na gamificação é totalmente personalizada, aumentando as chances do público-alvo aperfeiçoar as soft skills desejadas.
* **Aumento na produtividade**: De acordo com a ciência, a motivação tem forte relação com a produtividade, e a gamificação usa o propósito de metas, calcando as estratégias em alcançar um objetivo final.
* **Otimização de processos:** Através dos relatórios e disponibilização do acompanhamento do progresso dos participantes, é possível facilmente observar pontos de melhoria.
* **Aumento do engajamento em treinamentos:** Este é um dos mais expressivos ganhos que a gamificação proporciona, pois permite trabalhar técnica de storytelling e trabalhar temáticas do trabalho que geralmente não são contempladas em jogos.

Segundo um artigo da FIA (2020) os elementos da gamificação podem ser separados em três categorias principais: dinâmicas, mecânicas e componentes.

De acordo com estudos dos pesquisadores Werbach e Hunter, um ou mais componentes se ligam a uma mecânica, enquanto uma ou mais mecânicas estão relacionadas a uma dinâmica. Este mesmo artigo define os 28 elementos da gamificação, sendo:

1. **Dinâmicas:** Reúnem os elementos mais básicos e estruturais dos games, caracterizando o sistema sob o qual funcionam. Podem ser divididas em:

* **Emoções:** diversão e outros estados emocionais mantêm o interesse dos jogadores;
* **Narrativa:** é a estrutura que dá coerência ao game, seja através de uma história ou uma trajetória de eventos;
* **Progressão:** itens que mostram que o participante está avançando; **Relacionamentos:** interações entre o jogador, seus companheiros e adversários; **Restrições:** regras que impõem as limitações do universo do game.

1. **Mecânicas:** São fatores pensados a partir das regras do jogo, orientando as ações dos participantes para que eles avancem. Dependendo do game, as seguintes mecânicas podem aparecer:

* **Aquisição de recursos para ajudar na trajetória proposta;**
* **Avaliação ou feedback:** notifica sobre o desempenho do jogador;
* **Chance:** resultados são revelados de forma aleatória, a fim de surpreender os participantes;
* **Cooperação e Competição:** ao completar uma tarefa, o jogo mostra se houve desempenho satisfatório, ou não, em relação a outros competidores;
* **Desafios:** atividades que devem ser realizadas pelo participante;
* **Recompensas**: são como presentes recebidos após uma conquista;
* **Transações**: quando há troca, compra ou venda de recursos entres os jogadores;
* **Turnos**: períodos demarcados para que cada participante jogue;
* **Vitória**: a finalização do jogo de modo satisfatório.

1. **Componentes:** Descrevem as aplicações mais específicas e concretas entre os elementos dos jogos. Assim como as mecânicas, nem todas elas estão presentes em um game, pois dependem dos objetivos e dinâmicas estabelecidas. Abaixo, veja uma lista com componentes comuns:

* **Avatar:** representação visual do jogador (pode ser um peão, boneco, etc.); **Bens virtuais:** itens que só existem dentro do jogo, podendo ser conquistados ou comprados;
* **Boss:** grande desafio que marca o final de um ciclo ou etapa dentro do jogo**;  
  Coleções:** grupos de itens semelhantes coletados ao longo do game; **Conquistas:** recompensas adquiridas mediante o sucesso em um conjunto de ações específicas**;**
* **Conteúdos desbloqueáveis:** complementos que podem ser destravados através de alguma atividade ou pontuação;
* **Emblemas e medalhas:** símbolos de conquistas dentro do jogo;
* **Gráfico social:** representação visual da performance de outros jogadores;
* **Missão:** descrição de tarefas que devem ser concluídas pelo participante ou equipe;
* **Níveis:** números que revelam o quão evoluído e experiente é um jogador; **Pontos:** produtos de ações dentro do game;
* **Presentes:** itens ou moedas adicionais;
* **Ranking:** lista com os participantes de melhor desempenho dentro do jogo;
* **Times:** oportunidade de jogar ao lado de um ou mais companheiros para conquistar um mesmo objetivo.

# Desenvolvimento de aplicativos em Java

# A Cultura DevOps

# Controle de versão

# A ferramenta GIT

# Introdução ao Docker

# Banco de dados relacionais e linguagem SQL

# Microsserviços

Mais do que uma nova revolução nas arquiteturas de TI, os microsserviços são uma evolução da arquitetura SOA (para Service Oriented Architecture) de que muito se falava no início dos anos 2000. Um desenvolvimento lógico que resulta de um feedback de longa experiência na implementação da arquitetura SOA, e que vem apagar algumas das suas falhas. Jérôme Mainaud, arquiteto de softwares da Ippon Technologies, dá sua definição de microsserviços:

*"é um estilo de arquitetura de software em que os vários componentes são desenvolvidos na forma de serviços autônomos que rodam com seu próprio processo. arquiteturas onde os componentes escritos na forma de bibliotecas, módulos ou classes são todos implantados no mesmo processo."*

Os microsserviços nasceram, então, como uma nova abordagem na arquitetura e organização do desenvolvimento de software, trazendo a concepção de fragmentar o novo software em pequenos serviços independentes que se comunicam através de APIs.

As arquiteturas de microsserviços facilitam muito a escalabilidade e agilizam o desenvolvimento de novos aplicativo, ativando a inovação e acelerando o tempo de introdução de recursos no mercado. Essa nova forma de arquitetura e organização veio como uma alternativa à arquitetura monolítica.

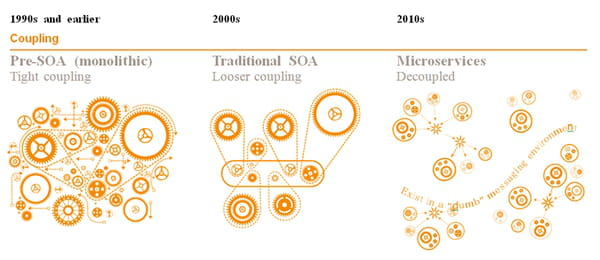
Conforme matéria publicada no site da AWS:

*“Com as arquiteturas monolíticas, todos os processos são altamente acoplados e executam como um único serviço. Isso significa que se um processo do aplicativo apresentar um pico de demanda, toda a arquitetura deverá ser escalada. A complexidade da adição ou do aprimoramento de recursos de aplicativos monolíticos aumenta com o crescimento da base de código. Essa complexidade limita a experimentação e dificulta a implementação de novas ideias. As arquiteturas monolíticas aumentam o risco de disponibilidade de aplicativos, pois muitos processos dependentes e altamente acoplados aumentam o impacto da falha de um único processo.”*

No caminho oposto, agora no viés da arquitetura de microsserviços, um aplicativo é criado como componentes independentes que executam cada processo do aplicativo como um serviço, através de interfaces de APIs leves. Como são executados de maneira independente, têm a característica de poder ser atualizado, implantado e escalado para atender a demanda de funções específicas do aplicativo.

A ambição é a mesma da SOA, ou seja, transformar aplicações monolíticas, por definição funcionalmente complexas e volumosas, em serviços simples que se comunicam entre si. Para Romain Level, consultor da Xebia, o microsserviço difere do SOA principalmente em dois pontos: “os pesados ​​e complexos ESBs de SOA são substituídos por barramentos de mensagens simples, sem complexidade de negócios e tecnicamente robustos"

Figura 2 - A transição da arquitetura de aplicativos monolíticos dos anos 90 para os aplicativos baseados em microsserviços de hoje.



Fonte: PWC (2015)

Conforme artigo publicado no site da AWS, as principais características dos microsserviços são:

* **Autônomos:** Cada serviço do componente de uma arquitetura de microsserviços pode ser desenvolvido, implantado, operado e escalado sem afetar o funcionamento de outros serviços. Os serviços não precisam compartilhar nenhum código ou implementação com os outros serviços. Todas as comunicações entre componentes individuais ocorrem por meio de APIs bem definidas;
* **Especializados:** Cada serviço é projetado para ter um conjunto de recursos e é dedicado à solução de um problema específico. Se os desenvolvedores acrescentarem mais código a um serviço ao longo do tempo, aumentando sua complexidade, ele poderá ser dividido em serviços menores.

E ainda contam com os benefícios de:

* **Agilidade:** Os microsserviços promovem uma organização de equipes pequenas e independentes que são proprietárias de seus serviços. As equipes atuam dentro de um contexto pequeno e claramente compreendido e têm autonomia para trabalhar de forma mais independente e rápida. O resultado é a aceleração dos ciclos de desenvolvimento. Você obtém benefícios significativos do throughput agregado da organização;
* **Escalabilidade flexível:** Os microsserviços permitem que cada serviço seja escalado de forma independente para atender à demanda do recurso de aplicativo oferecido por esse serviço. Isso permite que as equipes dimensionem corretamente as necessidades de infraestrutura, meçam com precisão o custo de um recurso e mantenham a disponibilidade quando um serviço experimenta um pico de demanda;
* **Fácil implantação:** Os microsserviços permitem a integração e a entrega contínuas, o que facilita o teste de novas ideias e sua reversão caso algo não funcione corretamente. O baixo custo de falha permite a experimentação, facilita a atualização do código e acelera o tempo de introdução de novos recursos no mercado;
* **Liberdade tecnológica:** As arquiteturas de microsserviços não seguem uma abordagem generalista. As equipes são livres para escolher a melhor ferramenta para resolver problemas específicos. O resultado é que as equipes que criam microsserviços podem optar pela melhor ferramenta para cada tarefa;
* **Código reutilizável:** A divisão do software em módulos pequenos e bem definidos permite que as equipes usem funções para várias finalidades. Um serviço criado para uma determinada função pode ser usado como componente básico para outro recurso. Isso permite que os aplicativos sejam reutilizados, pois os desenvolvedores podem criar recursos sem precisar escrever código;
* **Resiliência:** A independência do serviço aumenta a resistência a falhas do aplicativo. Em uma arquitetura monolítica, a falha de um único componente poderá causar a falha de todo o aplicativo. Com os microsserviços, os aplicativos lidam com a falha total do serviço degradando a funcionalidade, sem interromper todo o aplicativo.

As equipes que criam microsserviços geralmente preferem uma abordagem diferente aos padrões. Em vez de usar um conjunto de padrões definidos escritos em algum lugar no papel, eles preferem a ideia de produzir ferramentas úteis que outros desenvolvedores possam usar para resolver problemas semelhantes aos que estão enfrentando. Essas ferramentas geralmente são colhidas de implementações e compartilhadas com um grupo mais amplo, às vezes, mas não exclusivamente usando um modelo interno de código aberto. Agora que o git e o github se tornaram o sistema de controle de versão de fato preferido, as práticas de código aberto estão se tornando cada vez mais comuns internamente.

A Netflix é um bom exemplo de organização que segue essa filosofia. Compartilhar código útil e, acima de tudo, testado em batalha como bibliotecas incentiva outros desenvolvedores a resolverem problemas semelhantes de maneiras semelhantes, mas deixa a porta aberta para escolher uma abordagem diferente, se necessário. As bibliotecas compartilhadas tendem a se concentrar em problemas comuns de armazenamento de dados, comunicação entre processos e, como discutiremos mais adiante, automação de infraestrutura.

# Design Evolucionário

Os praticantes de microsserviços geralmente vêm de um background de design evolucionário e veem a decomposição de serviços como uma ferramenta adicional para permitir que os desenvolvedores de aplicativos controlem as mudanças em seus aplicativos sem desacelerar as mudanças. Controle de mudanças não significa necessariamente redução de mudanças - com as atitudes e ferramentas certas, você pode fazer mudanças frequentes, rápidas e bem controladas no software.

Sempre que você tenta quebrar um sistema de software em componentes, você se depara com a decisão de como dividir as partes - quais são os princípios sobre os quais decidimos dividir nossa aplicação? A propriedade chave de um componente é a noção de substituição independente e capacidade de atualização - o que implica procurar pontos onde podemos imaginar reescrever um componente sem afetar seus colaboradores. De fato, muitos grupos de microsserviços levam isso adiante, esperando explicitamente que muitos serviços sejam descartados em vez de evoluídos a longo prazo.

# METODOLOGIA

Sabemos que o Trabalho de Conclusão de Curso propicia o aprofundamento do conhecimento adquirido no curso, através da integração de conteúdo e disciplinas (Silva, 2020, p. 13), também entendemos como uma oportunidade para trocar conhecimento e buscar novos desafios.

Este Trabalho de Conclusão de Curso utilizou duas abordagens:

* Design Thinking: condução da entrega do trabalho de conclusão de curso;
* DevOps: definição do Ciclo de vida do desenvolvimento do software e criação de microsserviços.

# Design Thinking - 1ª Etapa: Definir

A partir da definição do grupo, foi feita uma reunião para entender os desejos individuais de cada integrante, neste momento conclui-se que o tema de interesse comum transitava no assunto desenvolvimento de softwares.

# Definição do Escopo

Para chegar ao escopo deste trabalho de conclusão de curso foram levantados os bônus e ônus da modalidade de ensino à distância e vimos como uma oportunidade, a criação de um aplicativo que, além de incentivar a troca de conhecimento teria a gamificação como componente de estímulo para manter a solução viva.

Decidiu-se por incluir como alicerce da entrega dois assuntos atuais:

* Arquitetura baseada em Microsserviços: arquitetura orientada a serviços (NEWMAN, 2022, p. 32);
* Cultura DevOps: metodologia de desenvolvimento de software que se concentra nos times de desenvolvimento e operações e em como eles podem trabalhar juntos de uma forma mais produtiva (Freeman, 2021, p.18).

# Delimitação do Escopo

Optou-se em uma entrega destinada à escalabilidade, portanto, a base de dados deste trabalho será enxuta, com dados suficientes para implementar as funcionalidades do aplicativo proposto.

O aplicativo para gamificação será concebido a partir da arquitetura de microsserviços que consiste no desmembramento de uma aplicação em aplicações menores e independentes que se comunicam por meio de protocolos abertos. Neste trabalho a aplicação será dividida nos seguintes microsserviços:

* Microsserviço Polo;
* Microsserviço Alunos;
* Microsserviço Disciplina;
* Microsserviço Base de Conhecimento: Respostas referentes à disciplina.

Os microsserviços se comunicarão por meio da interface requisição-resposta. Segundo Newman (2022, p. 190) nesse estilo de comunicação um microsserviço envia requisição e espera até receber uma resposta conforme exemplo:

*Figura 3- O microsserviço: Tabela envia uma requisição para Estoque solicitando os níveis de estoque.*

*Diagrama, Forma, Polígono

Descrição gerada automaticamente com confiança média*

*Fonte NEWMAN (2022)*

O desenvolvimento do software será aplicado a partir da criação de pipelines (conjunto de práticas) que segundo (MONTEIRO et al, 2021 apud DESTRO; FRANÇA, 2019) pode variar de acordo com o projeto, já que cada sistema suporta tecnologias distintas e possui ferramentas específicas.

Os estágios do ciclo de vida do desenvolvimento de software (Freeman, 2021, p.18) que aparecerão no decorrer deste trabalho são:

* Planejamento;
* Design;
* Codificação;
* Implementação.

Não serão contempladas neste trabalho as etapas de Testes e Manutenção. Essa decisão se baseou pela necessidade de se aprofundar nas tecnologias de gestão de ambientes isolados para construção da arquitetura, para cumpri-se o cronograma das entregas (gestão do tempo) e pelo fato de ter-se visto este conteúdo na graduação.

# Design Thinking - 2ª Etapa: Pesquisa

Nesta etapa, focou-se no levantamento dos requisitos funcionais relacionados ao comportamento do aplicativo. Segundo Filho (W.D.P. P., 2019 p.10) geralmente as características de softwares são classificadas em duas categorias:

* **Requisitos Funcionais:** representam os comportamentos que um programa deve apresentar diante de certas ações de seus usuários;
* **Requisitos não funcionais:** quantificam determinados aspectos do comportamento.

Para Filho (2019 p.10)  o levantamento e a documentação dos requisitos dos produtos de software quando bem-feitos, têm maiores chances de serem corretamente entendidos pelos desenvolvedores.

Em reunião definiu-se as principais funcionalidades do aplicativo (APÊNDICE A).

# Design Thinking - 3ª Etapa: Gerar ideias

Esta etapa foi a mais trabalhosa pois, foi preciso optar entre as centenas de frameworks e ferramentas disponíveis no mercado conforme salientado por Ambrose (2022,p.20):

Nesta etapa, a equipe de design também precisa escolher um entre os inúmeros movimentos e paradigmas da arte e do design – um briefing pode ser interpretado sob um ponto de vista modernista, abstrato, construtivista ou desconstrutivista, por exemplo.

Optou-se pelo uso de tecnologias Open Source por serem ferramentas que podem ser acessadas livremente por qualquer usuário, suportadas por uma ampla comunidade de desenvolvedores e pesquisadores.

Então, as tecnologias escolhidas são:

* **Docker:** Criação e administração de ambientes isolados;
* **Figma:** Construção do protótipo das telas do aplicativo front-end;
* **GitHub:** Hospedagem de código-fonte e controle de versão usando o Git;
* **IntelliJ IDEA:** Codificação;
* **Swagger:** Documentação da API;
* **MySQL:** Sistema Gerenciador de Banco de Dados Relacional.
* **Visual Studio Code:** Levantamento de requisitos funcionais e não funcionais.

# Design Thinking - 4ª Etapa: Testar protótipos

Nesta etapa iniciou-se o processo de criação das telas de front-end (APÊNDICE B) e do repositório (APÊNDICE C).

# Design Thinking - 5ª Etapa: Selecionar

Segundo Ambrose e Harris (2022 p.24) está é a etapa de validação das decisões, nesta comparamos a entrega do projeto à sua finalidade. As entregas foram refinadas e os desvios foram ajustados. Realizaram-se os seguintes tipos de ajustes:

* Ajuste tema;
* Ajuste da metodologia;
* Inclusão na fundamentação teórica da pesquisa referente a Banco de Dados relacionais e linguagem SQL dentro do microsserviços;
* Inclusão na fundamentação teórica da pesquisa referente Desenvolvimento de aplicativos em JAVA dentro do microsserviços;
* Inclusão da entrega referente à documentação da API;
* Refizemos o protótipo front-end para melhorar a experiência do usuário.

# Design Thinking - 6ª Etapa: Implementar

Nesta etapa foi construído o diagrama da arquitetura desenhada , a codificação do software desenvolvido e a documentação do mesmo.

# Design Thinking - 7ª Etapa: Aprender

Nesta etapa consta a análise e registro das lições aprendidas durante o desenvolvimento deste trabalho de conclusão de curso (APÊNDICE D). Segundo Xavier (2018, p.251) Lições aprendidas ou Lessons learned é a aprendizagem obtida no processo de realização de um projeto.

# DISCUSSÃO

# Criação do Projeto

Na fase de planejamento, a primeira etapa do ciclo de vida de desenvolvimento de um software, foram definidos a topologia de implantação de microsserviços, suas instâncias, o método de comunicação entre as instâncias e os bancos de dados.

# Definir os microsserviços em grupo;

* Requisitos funcionais e não funcionais/ Algoritmo do game (linguagem natural)

# Desenvolvimento do aplicativo

# Criação do repositório

<https://github.com/univesp-tcc-s4-g5>

# Criação de branches builds/ ou commits com o Git

# Versionando o código- fonte

# Criação de microsserviços com o Docker

# CONSIDERAÇÕES FINAIS

Através da pesquisa e das discussões internas do grupo pode-se evidenciar que a perspectiva do desenvolvimento a partir de microsserviços permite maior flexibilidade de dividir as aplicações com tecnologias que se adaptem melhor a elas seja com um banco de dados relacional ou não relacional, com backends independentes cuja modificação não impacte na aplicação como um todo. Conforme reforçado no apêndice E “Lições Aprendidas” algumas entregas foram repensadas durante o desenvolvimento do projeto:

1. Diferenciação entre o protótipo e a versão implementada: a complexidade da construção de microsserviços só foi identificada durante o desenvolvimento do projeto, portanto, algumas funcionalidades foram dispensadas na versão final;
2. Na delimitação foi informado que a etapa de testes não seria realizada em função de outras necessidades, no decorrer do desenvolvimento percebemos que mesma é fundamental e por isso aparece com grande ênfase no pipeline da esteira DevOps;
3. Não pensamos na escalabilidade quando definimos o uso de um único repositório ao invés de fazer um por microserviços;
4. Durante o desenvolvimento do projeto soube-se do Github pages, ferramenta que possibilita o compartilhamento da página sem custos de hospedagem;
5. O aprofundamento do desenvolvimento e infraestrutura em nuvem, recurso que possibilita economia em mão de obra e infraestrutura não foi aprofundado neste trabalho.

**REFERÊNCIAS**

AMBROSE, G.; HARRIS, P. **Design thinking. (Design básico)**. São Paulo: Grupo A, 2010. 9788577808267. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788577808267/>. Acesso em: 07 Mai 2022.

AWS. **O que são Microserviços?** Disponível em: <<https://aws.amazon.com/pt/microservices/#:~:text=Microsservi%C3%A7os%20s%C3%A3o%20uma%20abordagem%20arquitet%C3%B4nica,pertencem%20a%20pequenas%20equipes%20autossuficientes>>. Acesso em: 01 Mai 2022.

CLAPAUD, Alain. **O conceito de arquitetura de Microserviços**. JDN. Disponível em: <<https://www.journaldunet.com/solutions/cloud-computing/1166432-microservices-est-ce-realiste/1167035-le-concept-des-microservices>>. Acesso em: 30 Abr 2022.

[CRIANDO MICROSSERVIÇOS **– 2ª Edição: Projetando sistemas com componentes menores e mais especializados**](https://www.amazon.com.br/Criando-Microsservi%C3%A7os-Projetando-componentes-especializados-ebook/dp/B09WF2MVT8/ref=sr_1_3?qid=1651801083&refinements=p_27%3ASam+Newman&s=digital-text&sr=1-3&text=Sam+Newman) por Sam Newman | Vendido por: Amazon Serviços de Varejo do Brasil Ltda | 24 mar 2022.

FARDO, M. L. **A GAMIFICAÇÃO APLICADA EM AMBIENTES DE APRENDIZAGEM.** CINTED-UFRGS Novas Tecnologias na Educação. V. 11 Nº 1, julho, 2013 Disponível em: <<http://seer.ufrgs.br/renote/article/viewFile/41629/26409>>. Acesso em 02 mai 2022.

FIA: Fundação Instituto de Administração. Gamificação. O que é, vantagens e como implementar. Disponível em: <<https://fia.com.br/blog/gamificacao/>>. Acesso em: 30 Abr. 2022.

FILHO, W.D.P. P. **Engenharia de Software - Produtos - Vol.1**. São Paulo,Grupo GEN, 2019. 9788521636724. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788521636724/>. Acesso em: 29 Abr 2022.

FREEMAN, E. **DevOps Para Leigos**. São Paulo,Editora Alta Books, 2021. 9788550816661. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788550816661/>. Acesso em: 08 Mai 2022.

LUDOSPRO. Gamificação: O que é e quais os benefícios na aprendizagem? Disponível em: <<https://www.ludospro.com.br/blog/o-que-e-gamificacao>>. Acesso em 06 Mai 2022.

MARTINFOWLER. Microserviços. Disponível em: <<https://martinfowler.com/articles/microservices.html>>. Acesso em: 01 Mai 2022.

MONTEIRO, E.R.; CERQUEIRA, M.V.B.; SERPA, M.D.S.; AL., E. **DevOps**. São Paulo,Grupo A, 2021. 9786556901725. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9786556901725/>. Acesso em: 29 Apr 2022.

PINHEIRO, Tennyson; ALT, Luis. **Design Thinking Brasil: Empatia, colaboração e experimentação para pessoas, negócios e sociedade.** Rio de Janeiro: Alta Books, 2017.

RED HAT. O que é Open Source?. Disponível em: <<https://www.redhat.com/pt-br/topics/open-source/what-is-open-source>>. Acesso em 01 Mai 2022.

SILVA, D.F. D. **Manual prático para elaboração de trabalhos de conclusão de curso**. São Paulo: Editora Blucher, 2020. 9786555500028. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9786555500028/>. Acesso em: 05 Mai 2022.

WERBACH, Kevin. **(Re)Defining Gamification: A Process Approach**. In: Persuasive technology: 9th International Conference, PERSUASIVE 2014, Padua, Italy, May 21- 23, 2014.

XAVIER, C.M.D. S. **Gerenciamento de projetos: como definir e controlar o escopo do projeto**. São Paulo,Editora Saraiva, 2018. 9788553131204. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788553131204/>. Acesso em: 30 Abr 2022.

ZICHERMANN, Gabe; CUNNINGHAM, Christopher. **Gamification b y Design. Implementing Game Mecha nics in Web and Mobile Apps.** Canada : O’Reilly Media,2011.

# APÊNDICES

# APÊNDICE A – Requisitos Funcionais e não Funcionais

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

# APÊNDICE B – Protótipos front-end

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

Interface gráfica do usuário

Descrição gerada automaticamenteTexto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteUma imagem contendo Forma

Descrição gerada automaticamente

# APÊNDICE C – Repositório

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

# APÊNDICE D – Telas Finais

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo, Teams

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo, Word

Descrição gerada automaticamente Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário

Descrição gerada automaticamente

# APÊNDICE E – Lições aprendidas

1. A complexidade tem um custo por isso, deixamos a solução final mais enxuta que o protótipo;
2. Não mapeamos a necessidade dos requisitos de segurança e disponibilidade de infraestrutura;
3. Nas próximas implementações criar um repositório por microsserviços para facilitar a visualização dos serviços que sofreram alterações.