

CHALLENGE 2024

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - 1º ANO

CRONOGRAMA

FIAP

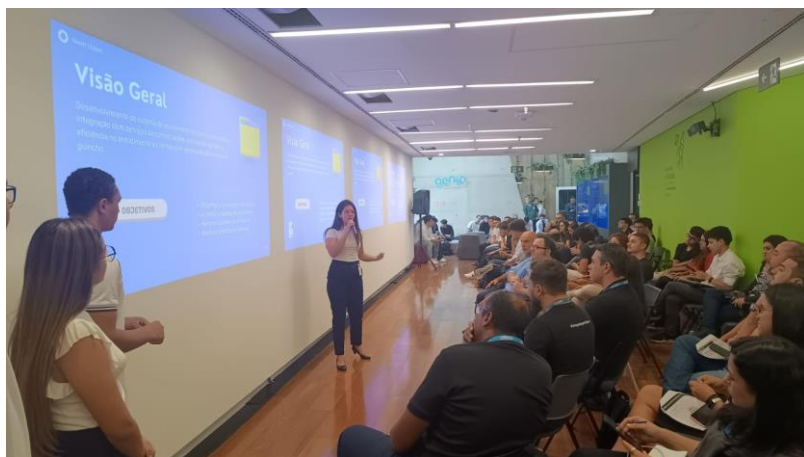
CRONOGRAMA - 2º SEMESTRE



DATA	EVENTO	STEAKHOLDER
26/08/2024	Mentoria Porto Seguro – Presencial, Unidade Paulista (Turmas manhã 10h – 12h)	PORTO
30/08/2024	Mentoria Porto Seguro – Presencial, Unidade Paulista (Turmas noite 19h – 21h)	PORTO
06/09/2024	Apresentação Online Banca de Professores	Professores
16/09/2024	ENTREGA DA SPRINT 3	ALUNOS
25/09/2024	Prazo limite para a postagem das notas da Sprint 3	PROFESSORES
De 26/09/2024 Até 02/10/2024	Feedback das entregas SPRINT 3 (Durante a aula com o professor)	PROFESSORES
18/09/2024	Apresentação Banca de Professores - Filtragem	Professores
A definir	Apresentação Banca Final Porto Seguro	Porto
26/10/2024	NEXT	FIAP
04/11/2024	ENTREGA DA SPRINT 4	ALUNOS
13/11/2024	Prazo limite para a postagem das notas da Sprint 4	PROFESSORES
De 13/11/2024 Até 15/11/2024	Feedback das entregas SPRINT 4 (Durante a aula com o professor)	PROFESSORES

- **A apresentação para a banca de professores (06/09) e a mentoria presencial da Porto Seguro (26/08 Manhã e 30/08 Noite)** será para orientações e direcionamentos no desenvolvimento dos projetos, se não participar, não conseguirá ter precisão para ficar entre os melhores projetos e ir para o NEXT.
- **A segunda apresentação para a banca de professores (18/09)** será para avaliar os projetos aptos a realizarem a apresentação final para a **Porto Seguro na sede da aceleradora Oxigênio, em horário comercial, ou seja, é uma apresentação que quem quiser tentar a chance de ir para o NEXT e ficar entre os três primeiros, essa apresentação é obrigatória. Serão selecionados até 16 grupos para a apresentação.**

- **A apresentação final para a Porto Seguro** será para avaliar os melhores projetos que participarão do evento NEXT e serão premiados na competição. **Nesse dia só irá participar quem foi selecionado entre os melhores projetos na banca de professores que ocorreu no dia 18/09.**



3º ENTREGAS

FIAP

3) DIAGRAMA DE SEQUÊNCIA (25 pontos)

Nível de implementação e que reflita o projeto do seu grupo para o challenge, tem que ter no mínimo cinco.

- ☐ Diagramas de sequência que represente a solução completa, então fique à vontade para criar a quantidade.
- ☐ Necessária para representar o sistema, se o professor verificar que não entregou o suficiente para representar todo o sistema, terá desconto de pontos.
- ☐ Não será aceito uso de outras ferramentas, SOMENTE ASTAH.
- ☐ Não será aceito diagrama sem os documentos descritivos.
- ☐ O uso correto dos conceitos deve ser praticado, caso contrário não será aceito

4) PROTÓTIPO DE MÉDIA FIDELIDADE(30 pontos)

Desenvolver um protótipo que já contemple TODAS as funcionalidades da aplicação. Pense que é um protótipo QUASE final do challenge. Nesse protótipo iremos observar erros e acertos das 10 Heurísticas de Nielsen e os Fundamentos de UX Writing. O Protótipo deve ser apresentado em vídeo pitch, explicando cada tópico solicitado. Caso tenha a solução em funcionamento pode ser utilizado em substituição ao Figma.

Se o link não funcionar ou estiver fechado, esse item é automaticamente zerado. Observem todas as 10 Heurísticas no protótipo de vocês.

20 pontos - Heurísticas de Nielsen;

10 pontos – Ux Writing.

Forma de entrega: Todas as entregas devem estar em um único documento no formato PDF.

1) PRODUCT BACKLOG ATUALIZADO (20 pontos)

☐ No Jira ou Trello contendo todas as funcionalidades propostas e ações, todo o histórico deve ser mantido. Devendo conter os requisitos funcionais, não funcionais e regras de negócio. (o link deve ser enviado no PDF, com liberação de acesso para o professor).

2) DIAGRAMA DE ATIVIDADES (25 pontos):

☐ Nível de implementação e que reflita o projeto do seu grupo para o challenge, tem que ter no mínimo cinco diagramas;

☐ Todos devem ser comentados;

☐ Represente a solução completa, então fique à vontade para criar a quantidade

☐ Necessária para representar as atividades do sistema, se o professor verificar que não entregou o suficiente para representar todo o sistema, terá desconto de pontos.

Sprint #3: Base de Dados para IA

Você e sua equipe deverão levantar uma base de dados para o treinamento de uma IA.

▣ Requisitos:

- Dados levantados aderentes ao tema do desafio, mínimo 100 exemplos; **[50 pontos]**
- Total de 100 pontos. Explicação dos dados (fonte ou como foram coletados, o que significam, quantidade e qualidade, rótulos, qual o objetivo que se pretende realizar com eles). **[50 pontos]**

▣ Entrega:

- ▣ Arquivo **.zip** com os dados. Os dados podem ser tabelas em .csv, imagens, áudios, etc. Desde que tenham aderência ao tema;
- ▣ Arquivo **.txt** com as explicações e nomes dos membros do grupo.

▣ Modelo Físico de Dados / Modelo Relacional

- Criar o modelo físico (arquivo .sql obrigatoriamente, com os comandos DDLs).
 - O modelo relacional deve apresentar além das restrições (CONSTRAINTS) chave primária (PRIMARY KEY), chave estrangeira (FOREIGN KEY) e obrigatória (NOT NULL) , pelo menos uma restrição única/exclusiva UNIQUE e uma restrição de verificação/validação CHECK, de acordo regras de negócio da solução proposta. (50 Pontos)
 - Todos os nomes devem estar padronizados, conforme nomenclatura trabalhada nas aulas: nomes de tabelas, nome de colunas e nome de restrições. (30 Pontos)
 - É, obrigatório, utilizar a ferramenta Oracle Data Modeler para construir o MER (modelo relacional). A não utilização dessas ferramentas irá acarretar desconto na nota. (20 Pontos)
- ▣ O arquivo referente ao MER desenvolvido DEVE ser gravado no formato .PDF

Sprint 3: Preparação dos dados para IA

A partir dos dados coletados nas outras disciplinas, a equipe deverá desenvolver um programa em Python, o qual deverá contemplar os seguintes itens:

- [30 pts] definição de funções para organizar, filtrar e armazenar os dados em sequências (listas de listas, lista de tuplas e/ou lista de dicionários) - considere as boas práticas, passagem de parâmetro e retorno de funções;
- [20 pontos] tratamento de erros para inserção, alteração e exclusão dos dados (try, except, else, finally);
- [20 pts] onde houver necessidade, deve haver validação de dados (entrada e processamento).
- [10 pts] menu com submenus para acesso ao CRUD dos dados, **o CRUD tem que estar funcionando, senão irá perder 30 pontos.**

COMPUTATIONAL THINKING USING PYTHON

Sprint 3: Preparação dos dados para IA

Entrega

- [20 pts] Compactar em um arquivo .zip:
 - Código fonte em Python (arquivos .py).
 - Vídeo explicativo de até 3 minutos contendo os nomes dos integrantes, a ideia, a solução proposta, explicação do código fonte e execução do mesmo.

DOMAIN DRIVEN DESIGN USING JAVA (1/2)

- ☐ **Documento PDF [20 pontos] - Os itens abaixo devem constar na documentação:**
 - ☐ Capa com o nome dos integrantes, nome da equipe e nome da solução.
 - ☐ Sumário.
 - ☐ Objetivo e escopo do projeto: descrever a solução proposta pelo grupo de forma objetiva.
 - ☐ Breve descrição das principais funcionalidades da solução.
 - ☐ Protótipo - apresentar as telas do sistema explicando como interagir (pode ser desenvolvida em qualquer ferramenta).
 - ☐ Modelo do banco de dados.
 - ☐ Diagrama de classes atualizada.
 - ☐ Procedimentos para rodar a aplicação.

DOMAIN DRIVEN DESIGN USING JAVA (2/2)

☐ Projeto Java:

- ☐ [20 pontos] Camada Model, com os devidos padrões seguidos em sala de aula.
- ☐ [20 pontos] Implementação de (no mínimo) quatro métodos de relevância, contendo lógica e regras de negócio definidos para o projeto (O grau de dificuldade será considerado na correção deste item).
- ☐ [10 pontos] Implementação de uma classe de teste que instancie objetos e teste os métodos implementados no tópico 2.
- ☐ [10 pontos] Camada com a classe de conexão, o usuário e senha tem que estar no código.
- ☐ [20 pontos] Implementação da camada e classe DAO com pelo menos um método insert, funcionando, **o CRUD tem que estar funcionando, senão irá perder 30 pontos.**

☐ Entrega:

- ☐ Projeto Java compactado (.zip) e entregue no portal do aluno no menu do Challenge.

Para esta etapa, vamos criar as telas, nas resoluções desenvolvidas na prototipação da 2ª entrega.

CONSIDERE TODOS OS ITENS A SEGUIR OBRIGATÓRIOS!

▣ REGRAS

- Desenvolver um projeto em **vite-react + Typescript** com base nas estruturas apresentadas no **FIGMA** e no **HTML** durante as entregas das **Sprints** 1 e 2.
- A utilização do **vite-react + Typescript** para a componentização da aplicação **SPA**, seguindo as melhores práticas discutidas em **sala de aula**.
- Reestruturar o projeto apresentado na **Sprint** 2, em (HTML), transformando-o em uma aplicação de página única (single-page-application), com estrutura semântica e responsiva conforme estabelecido nas entregas anteriores, navegação entre páginas utilizando **rotas**.

■ REGRAS (continuação)

- A aplicação não necessita, neste momento, de integração com uma API. Contudo, se houver elementos interativos de entrada de dados do usuário. Ex: formulários de cadastro, campos de pesquisa etc.
- Estes devem ser incluídos conforme descrito na documentação e nas entregas anteriores.
- O projeto deve ser desenvolvido em conformidade com as normas estabelecidas pelo W3C.
- É estritamente proibido o uso de templates ou exercícios fornecidos durante as aulas. O descumprimento dessa norma resultará automaticamente na atribuição de nota ZERO para a equipe.
- A estilização deve ser realizada utilizando styled-components ou CSS3 Modules.
- Uma das páginas do projeto deverá conter obrigatoriamente os nomes e RM dos integrantes da equipe. Alunos pertencentes a outras turmas devem ser devidamente identificados com seus RM e turma.

- **REQUISITOS E CRITÉRIOS DE AVALIAÇÃO**

- **Construção do projeto de acordo com a 2ª entrega (20,0 pontos)**

- **Escolha de Cores e Fontes (10,0 pontos)**

- Coerência das cores de acordo com a paleta apresentada no FIGMA. (5,0 pontos)
 - Consistência no uso das fontes em todo o projeto(Padronização). (5,0 pontos)

- **Atratividade do Design (10,0 pontos)**

- Contextualização entre documentação e entrega, ou seja o que foi apresentado deve ser entregue. (5,0 pontos)
 - Facilidade de navegação, interface amigável. (5,0 pontos)

❑ REQUISITOS E CRITÉRIOS DE AVALIAÇÃO (continuação)

❑ Componentização das páginas e itens reaproveitáveis (40,0 pontos)

❑ Modularidade e Reutilização (20,0 pontos):

- ❑ Identificação e separação clara de componentes, utilizando **vite-react + Typescript** . (7,0 pontos)
- ❑ Facilidade de reutilização de elementos em diferentes partes do projeto. (7,0 pontos)
- ❑ Eficiência na manutenção e atualização de componentes. (6,0 pontos)

❑ Padronização e Consistência (20,0 pontos)

- ❑ Coerência na estruturação e organização dos elementos. (10,0 pontos)
- ❑ Consistência no uso de estilos, classes e IDs. (10,0 pontos)

▣ REQUISITOS E CRITÉRIOS DE AVALIAÇÃO (continuação)

▣ Responsividade das páginas: Desktop, Tablet e Mobile. (20,0 pontos)

▣ Adaptabilidade (10,0 pontos)

- ▣ Funcionalidade e aparência adequada em diferentes dispositivos. (3,3 pontos)
- ▣ Adequação do layout e conteúdo para diferentes tamanhos de tela. (3,3 pontos)
- ▣ Navegabilidade e usabilidade em todas as plataformas(**Desktop, Tablet e Mobile**). (3,4 pontos)

▣ Performance (10,0 pontos)

- ▣ Velocidade de carregamento e fluidez da interface em cada dispositivo. (5,0 pontos)
- ▣ Ausência de problemas de redimensionamento ou sobreposição de elementos. (5,0 pontos)

FRONT-END DESIGN ENGINEERING (6/8)

■ REQUISITOS E CRITÉRIOS DE AVALIAÇÃO (continuação)

■ GIT/GITHUB (10,0 pontos)

■ Colaboração e Uso do Git (7,0 pontos)

- Clareza nos commits com mensagens descritivas. (3,5 pontos)
- Contribuições distribuídas e evidências de trabalho em equipe, projeto com no mínimo 10 commits e participação de todos os integrantes. (3,5 pontos)

■ README.MD (3,0 pontos)

- O grupo deve criar um arquivo README, formatado de acordo com MD(MarkDown), apresentando todas as informações pertinentes para manipular o sistema. **(3,0 pontos)**

■ Vídeo (5,0 pontos)

■ Gravação de vídeo (5,0 pontos)

- Grupo deverá gravar um vídeo de no máximo 3 minutos apresentando os recursos do projeto, telas, layout, o mesmo poderá ser disponibilizado via link e hospedado por exemplo Youtube. (5,0 pontos)

☐ **REQUISITOS E CRITÉRIOS DE AVALIAÇÃO** (continuação)

☐ **Entrega dentro dos padrões solicitados no documento (5,0 pontos)**

☐ **Qualidade da Entrega (5,0 pontos)**

- ☐ Boas práticas de desenvolvimento. (2,5 pontos)
- ☐ Ausência de erros significativos ou falhas na implementação. (2,5 pontos)

☐ **Disciplinas para integração**

- ☐ Não há.

▣ ENTREGA

- ▣ O grupo deverá **compactar e entregar** o repositório do projeto em formato .ZIP, seguindo as diretrizes do .gitignore para garantir a exclusão da pasta NODE_MODULES e/ou .NEXT. O não cumprimento desta norma resultará em um arquivo excedendo 50MB, ultrapassando o limite do portal da FIAP, acarretando na **PERDA** de **CINQUENTA PONTOS** para todos os integrantes do grupo.
- ▣ O aluno encarregado é responsável por revisar o documento antes da entrega, garantindo a ausência de falhas ou equívocos. Recomenda-se realizar testes em múltiplas máquinas, se necessário.
- ▣ A entrega deve ser feita por apenas um aluno do grupo. Caso ocorra a entrega por mais de um aluno, será **descontado UM ponto do grupo para cada entrega adicional**.

▣ Local de entrega

- ▣ Portal do Auno

4º ENTREGAS

FIAP

1) Apresentação para venda da solução (20 pontos):

Elaborar o material de apresentação final da sua solução para o cliente, com identidade visual, nome da startup, solução. Neste documento deve ter: Problemas que serão solucionados, suas funcionalidades, aplicabilidade no mercado, diferenciais dos concorrentes, quem são seus concorrentes. Método de aplicação, como será mantido, qual a inovação que está atuando. Formato PDF. Deverá seguir regras de padronização em todo o texto (tamanho, letra, parágrafo, concordâncias).

2) Plano de negócios completo (30 pontos):

Para validar da incubação da solução) deve conter todos os tópicos apresentados pelo modelo do professor em sala. Documentação, Diagramas (caso de uso, atividades e sequência) com as correções das últimas Sprints.

3) Precificação (25 pontos):

Apresentar cálculos referentes os custos que sua equipe teve e irá ter para entregar a solução funcionando e a permanência dela em perfeitas condições de uso (inclua horas dedicadas, tipos de níveis de profissionais dedicados, materiais, inteligência, com custos diretos e indiretos;). Precisamos entender qual o custo, investimentos e a viabilidade da sua solução. Deve ser entregue todo o racional de cálculo.

4) SLA (25 pontos):

Elabore um Acordo de Nível de Serviço (SLA) para garantir a qualidade e a confiabilidade da sua solução oferecida. Deve conter: quais são os componentes que terão no seu contrato, especificação dos níveis de desempenho esperados, definição das métricas utilizadas para medir o desempenho do serviço, responsabilidades de manutenção, suporte, e fornecimento de informações relevantes e os métodos de monitoramento.

Forma de entrega: Todas as entregas devem estar em um único documento no formato PDF.

Sprint #4: Modelos de IA

Nesta etapa, você e seu grupo usarão os dados do Sprint 3 para criar modelos de inteligência artificial.

▣ Requisitos:

- ▣ Usando os dados levantados, crie um modelo de aprendizado de máquina. O modelo pode ser de classificação, regressão e/ou agrupamento; **[70 pontos]**
- ▣ O modelo treinado deve ser disponibilizado em uma API REST para ser consumido pela sua aplicação. **[30 pontos]**

▣ Entrega:

- ▣ Arquivo **.ipynb** com os modelos de aprendizado de máquina, discussão dos resultados; em *markdown* e todo o código em Python implementado;
- ▣ Arquivo do modelo treinado (várias extensões são possíveis como .joblib, .pickle, .h5, etc.);
- ▣ Arquivo de configuração da API REST (por exemplo, se em Node-RED, o .json).

■ **SCRIPT DDL E DML - CRIAÇÃO DA ESTRUTURA E DADOS PARA POPULAR AS TABELAS PARA OS TESTES DA APLICAÇÃO**

- Arquivo com as instruções DDL (scripts da 3a sprint corrigido) e DMLs referente a criação e carga de dados para testes.
- Cada tabela deve ser preenchida com no mínimo 10 linhas. As tabelas associativas, devem ser preenchidas com no mínimo 20 linhas.
- A massa de dados, deve ser composta por dados válidos, ou seja, não devem ser inseridos: xxxx,11111,teste, ou similar. Trabalhar com dados fictícios mas coerentes. (20 pontos)

▣ **SCRIPT DQL - CONSULTAR AS TABELAS PARA OS TESTES DA APLICAÇÃO**

Implementar a criação de relatórios na aplicação, deverá ser implementado os seguintes relatórios:

- Relatório utilizando classificação de dados, a escolha da tabela é decisão do grupo.[10 pontos]
 - Relatório utilizando alguma função do tipo numérica simples.[10 pontos]
 - Relatório utilizando alguma função de grupo. [15 pontos]
 - Relatório utilizando sub consulta. [15 pontos]
 - Relatório utilizando junção de tabelas [30 pontos]
 - Pelo menos 2 relatórios de cada versão solicitada.
 - O arquivo DEVE ser gravado no formato .SQL.
- ▣ Inserir no início do arquivo, em forma de comentário, o nome e RM de cada componente do grupo (OBRIGATÓRIO).

Sprint 4: Conectando ao Banco de dados

Tendo como referência o entregável da sprint 3, a equipe deverá realizar a integração com o banco de dados, desenvolvendo um CRUD em Python. Para tanto, os requisitos são os seguintes:

- ▣ [10 pontos] Estrutura de menus e submenus para acesso às opções do CRUD (inserir, alterar, excluir e consultar);
- ▣ [40 pontos] Realizar ao menos um exemplo de cada operação do CRUD e disponibilizar uma opção, para exportar os dados dessas consulta para um arquivo JSON. **Se o CRUD tem que estar funcionando, senão irá perder 30 pontos.**
- ▣ [20 pontos] Desenvolvimento e/ou consumo de uma API externa pública.
- ▣ [10 pontos] As informações colhidas e/ou alteradas pelo programa desenvolvido devem refletir no sistema web (front-end) e vice-versa.

COMPUTATIONAL THINKING USING PYTHON



Sprint 4: Conectando ao Banco de dados

Entrega:

- [20 pts] Compactar em um arquivo .zip:
 - Código fonte em Python (arquivos .py).
 - Vídeo explicativo de até 3 minutos contendo os nomes dos integrantes, a ideia, a solução proposta, explicação do código fonte e execução, e conformidade com o front-end.

DOMAIN DRIVEN DESIGN USING JAVA (1/3)

☐ Documento PDF Atualizado (10 pontos)

- ☐ Capa com o nome dos integrantes, nome da equipe e nome da solução.
- ☐ Objetivo e escopo do projeto: descrever a solução proposta pelo grupo de forma objetiva.
- ☐ Breve descrição das principais funcionalidades da solução.
- ☐ Tabela dos endpoints (API Restful) contendo as URIs, verbo HTTP e Códigos de status de respostas para a requisição.
- ☐ Protótipo - Prints das telas implementadas.
- ☐ Modelo do banco de dados.
- ☐ Diagrama de classes atualizada.

DOMAIN DRIVEN DESIGN USING JAVA (2/3)

☐ Projeto Java Finalizado:

- ☐ (10 pontos) Camada Model com todas as classes necessárias, de acordo com o banco de dados.
- ☐ (30 pontos) Camada DAO e Service (validações), com todas as funcionalidades necessárias para o Front-End do projeto, **o CRUD tem que estar funcionando, senão irá perder 30 pontos.**
- ☐ (30 pontos) API Restful com todos os endpoints necessários para o funcionamento do front-end do projeto.
- ☐ (20 pontos) – Seguir todas as boas práticas trabalhadas em sala de aula, como regras de nomenclatura, tratamento de exceções, padrões de projetos etc.

☐ **Obs.: Não é necessária uma classe de teste, desde que o projeto implemente API Restful.**

DOMAIN DRIVEN DESIGN USING JAVA (3/3)



- ☐ **Entrega:**

- ☐ **Link do repositório do GitHub com o Código Java e Documento PDF.**

Chegou a hora de finalizarmos nosso projeto.

Para esta etapa, vamos considerar as regras da 3ª entrega **MAIS** as regras abaixo

CONSIDERE TODOS OS ITENS A SEGUIR OBRIGATÓRIOS!

▣ REGRAS

- ▣ Para completar nossa implementação, é crucial a integração de uma *API* em nosso projeto. Essa etapa marca a convergência de todos os elementos do projeto, assegurando que o *frontend* esteja perfeitamente alinhado com o *backend*.

▣ REGRAS (continuação)

- ▣ A *API*, desenvolvida na disciplina de **Domain Drive Design Using Java**, será responsável por enviar e receber dados entre o *backend* e o *frontend*. Ela trará todos os dados coletados no *backend* para o *frontend*, possibilitando também o envio de dados do *frontend* para o *backend*, viabilizando seu armazenamento no banco de dados, tal como abordado na disciplina de **Building Relational Database**.
- ▣ É essencial que as informações capturadas ou alteradas pelo sistema desenvolvido na disciplina de **Computational Thinking Using Python** sejam refletidas no sistema *web*, e vice-versa, para garantir uma integração plena.
- ▣ O projeto final deverá ser estruturado utilizando o **vite-react + Typescript + NEXT.JS** para o roteamento de páginas e outras adaptações necessárias.
- ▣ Além disso, é fundamental realizar o **deploy** do projeto na plataforma **Vercel**, onde será disponibilizada uma **URL** de acesso para a avaliação.

▣ REQUISITOS E CRITÉRIOS DE AVALIAÇÃO

▣ Construção do projeto de acordo com a 3ª entrega (20 pontos)

▣ Estrutura e Organização do Código (7 pontos)

- ▣ Projeto bem estruturado e nomenclatura de arquivos organizados de maneira clara. (2,0 pontos)
- ▣ Uso consistente de nomes para variáveis, funções e componentes. (2,0 pontos)
- ▣ Código modular e reutilizável. (3,0 pontos)

▣ Estilo e Design (7 pontos)

- ▣ Escolha de cores, fontes e elementos visuais alinhados com o propósito do projeto. (3,5 pontos)
- ▣ Adaptação e responsividade do design em diferentes dispositivos. (3,5 pontos)

▣ Funcionalidades Implementadas (6 pontos)

- ▣ Implementação bem-sucedida das funcionalidades descritas no escopo do projeto. (3,0 pontos)
- ▣ Ausência de bugs críticos ou erros que impactem a usabilidade. (3,0 pontos)

▣ REQUISITOS E CRITÉRIOS DE AVALIAÇÃO (continuação)

▣ Criação de Rotas e Navegação com NEXT.JS (20 pontos)

▣ Correta Implementação de Rotas (8 pontos)

- ▣ Definição adequada e funcional de rotas utilizando **TYPESCRIPT + NEXT.JS**. (4,0 pontos)
- ▣ Navegação fluida entre páginas. (4,0 pontos)

▣ Gerenciamento de Estado da Aplicação (7 pontos)

- ▣ Utilização eficiente do estado da aplicação para a troca de informações entre as páginas. (3,5 pontos)
- ▣ Manuseio correto de parâmetros de rota, quando aplicável. (3,5 pontos)

▣ Performance e Otimização (5 pontos)

- ▣ Desempenho satisfatório na transição entre rotas. (2,5 pontos)
- ▣ Utilização de técnicas para otimização do carregamento das páginas. (2,5 pontos)

▣ REQUISITOS E CRITÉRIOS DE AVALIAÇÃO (continuação)

▣ Consumo das APIs (25 pontos)

▣ Integração Adequada com APIs (10 pontos)

- ▣ Correto uso das chamadas de API conforme a documentação fornecida. (5,0 pontos)
- ▣ Tratamento de erros e respostas inesperadas. (5,0 pontos)

▣ Funcionalidades Implementadas com APIs (8 pontos)

- ▣ Integração bem-sucedida das funcionalidades que dependem das APIs. (4,0 pontos)
- ▣ Manipulação correta dos dados obtidos das requisições. (4,0 pontos)

▣ Segurança e Gerenciamento de Dados (7 pontos)

- ▣ Garantia de segurança nas transações de dados entre o frontend e o backend. (3,5 pontos)
- ▣ Adequado tratamento e armazenamento dos dados obtidos das APIs. (3,5 pontos)

FRONT-END DESIGN ENGINEERING (6/9)

■ REQUISITOS E CRITÉRIOS DE AVALIAÇÃO (continuação)

■ GIT/GITHUB (10,0 pontos)

■ Colaboração e Uso do Git (7,0 pontos)

- Clareza nos commits com mensagens descritivas. (3,5 pontos)
- Contribuições distribuídas e evidências de trabalho em equipe, projeto com no mínimo 10 commits e participação de todos os integrantes. (3,5 pontos)

■ README.MD (3,0 pontos)

- O grupo deve criar um arquivo README, formatado de acordo com MD(MarkDown), apresentando todas as informações pertinentes para manipular o sistema. **(3,0 pontos)**

■ Vídeo (5,0 pontos)

■ Gravação de vídeo (5,0 pontos)

- Grupo deverá gravar um vídeo de no máximo 3 minutos apresentando os recursos do projeto, telas, layout, o mesmo poderá ser disponibilizado via link e hospedado por exemplo Youtube. (5,0 pontos)

▣ REQUISITOS E CRITÉRIOS DE AVALIAÇÃO (continuação)

▣ Deploy do Projeto para a Plataforma Vercel (15 pontos)

▣ Sucesso no Deploy (8 pontos):

- ▣ Efetivação bem-sucedida do deploy na plataforma Vercel. (4,0 pontos)
- ▣ Disponibilização de uma URL funcional para acesso ao projeto. (4,0 pontos)

▣ Configuração e Otimização (7 pontos):




- ▣ Configuração adequada das variáveis de ambiente, se necessárias. (3,5 pontos)
- ▣ Otimização para um carregamento rápido e eficiente do projeto. (3,5 pontos)

▣ Entrega Dentro dos Padrões Solicitados no Documento (5 pontos)

▣ Atendimento às Especificações (5 pontos):

- ▣ Cumprimento dos requisitos e funcionalidades estabelecidas no documento. (2,5 pontos)
- ▣ Conformidade com as diretrizes e padrões exigidos. (2,5 pontos)

DISCIPLINAS PARA INTEGRAÇÃO

-  Building Relational Database
-  Computational Thinking Using Python (opcional)
-  Domain Drive Design

■ ENTREGA

- O grupo deverá **compactar e entregar** o repositório do projeto em formato .ZIP, seguindo as diretrizes do .gitignore para garantir a exclusão da pasta NODE_MODULES e/ou .NEXT. O não cumprimento desta norma resultará em um arquivo excedendo 50MB, ultrapassando o limite do portal da FIAP, acarretando na **PERDA** de **CINQUENTA PONTOS** para todos os integrantes do grupo.
- O aluno encarregado é responsável por revisar o documento antes da entrega, garantindo a ausência de falhas ou equívocos. Recomenda-se realizar testes em múltiplas máquinas, se necessário.
- A entrega deve ser feita por apenas um aluno do grupo. Caso ocorra a entrega por mais de um aluno, será **descontado UM ponto do grupo para cada entrega adicional**.

■ Local de entrega

- Portal do Auno

DÚVIDAS?!

OBRIGADO!!

